From Visual Representation to Object Discovery and Back

Giorgos Tolias

Habilitation Thesis

Czech Technical University in Prague

February 2022

Abstract

This thesis summarizes the author's post-PhD work that has a major focus on instance-level recognition tasks, such as instance-level search primarily, but also instance-level recognition. Visual representation is at the core of most computer vision tasks. This is addressed in the first part of the manuscript with crafting and learning of visual representations and similarity measures. A number of different representation approaches are summarized under the same framework provided by a match kernel formulation, while the interplay between local and global representation is highlighted. Given the representation, efficient estimation of similarity with respect to a large number of examples enables visual search applications. This is covered in the second part especially in the form of query expansion, which goes beyond pairwise similarity and treats a collection of examples as a whole. The last part discusses object discovery as an outcome of exploiting visual similarity and search within a large unordered collection of examples. Its result is then used to improve other components, namely representation and search, therefore, forming a circle and pronouncing the synergy between different contributions in this thesis.

Acknowledgements

This manuscript carries the name of the candidate, myself, but the work presented is, and could only be, the outcome of team collaboration. Therefore, the whole manuscript is written in first plural, and the only part in first singular is here, where I cannot thank enough all the wonderful researchers I collaborated with over the years and in different teams in Athens at NTUA, in Rennes at Inria, and in Prague at CTU.

Special thanks goes to professor Stefanos Kollias for trusting me to start a PhD. Yannis Avrithis had an instrumental influence on my development with his guidance, through his passion for science, and by being a great mentor and a positive role model. Words can hardly explain how grateful I am to him. He managed to make the advisor-advisee relationship one of the highlights of my PhD times. Even though the joint work with Yannis Kalantidis is not included in this manuscript, one cannot not thank their research bother for being there during deadlines, rejections, learning, exploring, trips, fun, and backing-up each other. As of today, I am extremely lucky to be collaborating with him again.

Even though my stay at Inria was relatively short as a PhD intern and a post-doc, the interaction with Herve Jegou was intense and had a great impact on me as well; I learned a lot from him and I thank him for that. I thank Andrei Bursuc, a very inspiring colleague, and Teddy Furon for our collaboration. The joint work with Ahmet Iscen started at Inria and continued over my years at CTU. It has been a pleasure to work with him and observe his great development since then.

One of the major reasons for my move to Prague was to work with Ondrej Chum whom I cannot thank enough for being a great and supportive mentor and for giving me critical opportunities as a member of his team. I thank Jiri Matas for his support and advice during the recent years regarding research, teaching, and career development. The journey at CTU has been wonderful with student collaborators or advisees on board including Arun Mukundan, Filip Radenovic, Tomas Jenicek, Oriane Simeoni, Nikolaos Ypsilantis and Yash Patel. I have learned a lot through the interaction with them. Now, a new journey just began with Pavel, Juliette, Nikos, Vladan, Monish, and Zakaria and I am looking forward to it.

Last but not least I owe a lot to all the loved ones, not colleagues, that have been by my side and keep trying to understand what it means to do research in computer vision.

Dedicated to everyone who has helped me and to everyone whom I can help.

Contents

1 Introduction

Machine-extracted representation of visual content in images and videos is used in computer vision to help estimate human-perceived visual similarity. This is an essential part of many computer vision methods and applications. Visual similarity is a relationship that holds between two perceptual objects, *i.e*. the perceptual resemblance of objects to one another. Human perceptual visual similarity, i.e. the human judgment of whether two scenes or objects are similar or not, is thought to incorporate multiple different levels of visual analysis such as shapes, objects, textures, brightness, layout, color, and motion. The ability to estimate visual similarity allows to transfer and extend existing knowledge. When things appear to be similar to each other, humans tend to group them and also think they have similar properties, purpose, and functionality. In addition to humans, the same holds for machines; automatic estimation of visual similarity enables ways to infer about previously unseen objects.

In the field of computer vision, and in particular in the task of visual similarity estimation, *visual representations* are extracted to represent the visual content. Then, visual representation is compared across two images or videos to estimate the degree of *visual similarity*. The representation and the matching process should be able to capture a variety of visual cues according to the target application. It is the target application, along with the underlined visual cues or the corresponding level of semantics, that define what is a similar object and what is not. Common examples include visual content corresponding to the same physical entities, *e.g*. two views of the same particular man-made object or natural location, to objects from the same animal or plant species, *e.g*. two different "Barn owls", or to objects from the same generic object category, *e.g*. two different cameras. The case of matching different instances of the same physical entities is the major scope of the work presented in this manuscript, which is called *instance-level recognition*. Major challenges include large changes in viewpoint and illumination, object occlusion and severe background clutter, such as in the case of small objects.

The visual representation and the accompanying matching process for similarity estimation can be either hand-crafted, also termed simply as crafted, or learned. Crafted models consist of hand-designed processes that integrate prior domain knowledge in order to handle the aforementioned challenges. Learned models heavily rely on a set of existing examples with a known degree of resemblance to learn the desired invariance. Nevertheless, the design of learned models plays an important role as well. Widely used learned models, such as Convolutional Neural Networks (CNN), are training data hungry. At the same time, obtaining labeled training data is very costly, especially in the task of instance-level recognition, where each indistinguishable object of the visual word forms its own class. There is an apparent necessity for automated methods to obtain labeled data or for learning methods that exploit unlabeled data additionally to the labeled ones.

Other than comparing two images, it is important to compare a query image with all elements of a large image collection and rank them according to the estimated visual similarity, from the most similar to the least. This is the goal in the task of *visual search*, also termed as instance-level search or retrieval when the goal is to retrieve particular objects. Visual search with query-by-example allows users to obtain content-based access to large image and video collections and enables their exploration. It provides ways to identify elements of identical or similar visual content in large unorganized collections. The need for such access became essential after the size explosion of public-online and personal multimedia collections. Some of the domains of professional applications are copyright protection of visual content, trademark (logo) identification, art exhibit recognition, street-to-online-shop retail product matching, *etc*. Search at a very large scale makes the efficiency of the matching to be the most essential property. Compactness of the representation is directly affecting the search efficiency while also participating in a trade-off with effectiveness; compact representations make it even more challenging to handle background clutter and small objects. Therefore, visual search becomes a challenging task with many real-world applications.

Comparing all images within a large collection, *i.e*. by using each one as a search query, is an essential first step that with further processing constitutes a way to discover and establish links between similar elements. In this way, connectivity graphs are formed that facilitate organization and visual navigation of large, initially unordered, collections. Moreover, the result of such a *visual discovery* process is a valuable input to other computer vision and machine learning methods that involve training from similar examples.

This manuscript summarizes our work on the following four interconnected tasks:

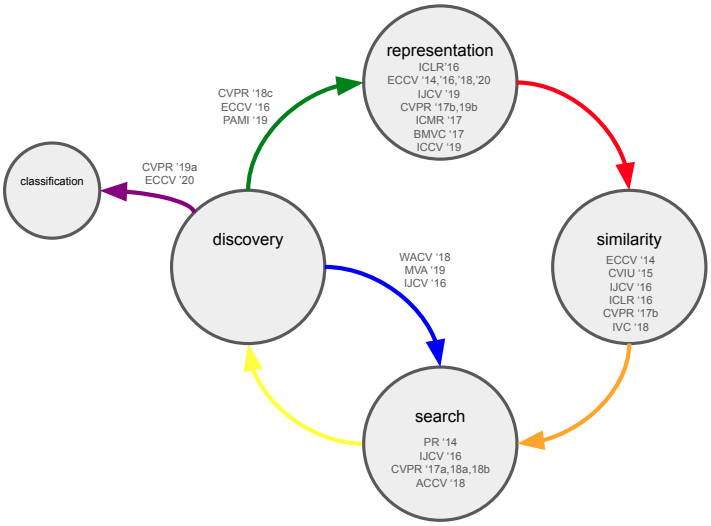


Figure 1: Main four axes of the work presented in this manuscript, their interconnection, and our contributions as published in international conferences and journals.

- *representation* that is designed or learned to represent whole images, image regions, or objects depicted in them
- *similarity* that is estimated by a matching process that uses the obtained representation
- *search* to efficiently and effectively estimate similarity of each element of a large image collection with respect to a query image
- *object discovery* by exploring visual similarity among all elements of a collection.

These four different tasks, are essentially different stages of a cyclical process, as shown in Figure 1, to improve visual representation without any human supervision in the loop; some sort of self-supervised learning which is a research direction that is attracting a lot of attention. Visual similarity is estimated on top of the obtained visual representation. Visual search properly indexes the representation and efficiently estimates the visual similarity for a large set of elements. Visual discovery exploits visual search to efficiently obtain the required similarities within a collection and discover similar and repeating entities, objects, or objects parts within the collection. Such a discovery process is performed automatically, and without any human supervision in the loop. A circle is formed by using the result of discovery as input to improve visual search or representation learning approaches that will learn an improved visual representation that will consequently improve the following stages too.

The rest of the manuscript is organized as follows. Sections 2, 3, and 4 present our contributions to visual representation/similarity, search, and object discovery respectively. The relevant publications are not discussed in a single piece but the respective contributions may appear in multiple sections according to the topic. Section 5 presents datasets and relevant benchmarks that are part of our contributions. Finally, a summary is provided in Section 6.

2 Visual representation and similarity

In the context of this manuscript visual similarity is estimated between a pair of examples. An example is an image region, an image, or a set of images. An example is assumed to be composed by a set of elements. In the case of an image region, the corresponding elements are its pixels. In the case of an image, the corresponding elements are either its pixels or a set of regions. In the case of image sets, obviously, the corresponding elements are the individual images.

In the following, no matter the context, a pair is denoted by examples $X \in \mathcal{X}$ and $Y \in \mathcal{X}$, where $\mathcal{X}$ is the space of all possible examples. Equivalently, examples are seen as sets $X = \{x\}$ and $Y = \{y\}$, with $x, y \in \mathcal{X}_{el}$, where $\mathcal{X}_{el}$ is the space of all set elements.

Two ways are considered for estimating the pairwise similarity

$$S : \mathcal{X} \times \mathcal{X} \to \mathbb{R}, \tag{1}$$

whose value is typically normalized in $[-1, 1]$ or additionally restricted to be positive in $[0, 1]$.

The first way is achieved by what is referred to as *single vector representation*, or *global descriptor* in the case of images as examples, *i.e.* the whole image is described with a single vector. It involves a representation function and standard similarity measure in the representation space:

$$f : \quad \mathcal{X} \to \mathbb{R}^D \tag{2}$$

$$S_D : \quad \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R} \tag{3}$$

$$S_g(X, Y) = \quad S_D(f(X), f(Y)), \tag{4}$$

where $f : \mathcal{X} \to \mathbb{R}^D$ is the representation function that maps each example to a D-dimensional representation space, *i.e.* a vector space. Vector $f(X)$ is called the representation of X, its embedding, or descriptor, equivalently. Such embedding is often ℓ_2-normalized, *i.e.* $f(\cdot)$ is a mapping on the unit hypersphere. Function S_D is a common similarity measure, or an increasing function of a common distance measure. For example, $S_D(X, Y) = f(X)^\top f(Y)$ for dot product, equivalent to cosine similarity for ℓ_2-normalized embeddings.

The second way is achieved by using a *match kernel* over all elements of both sets. This is given by

$$S_m(X, Y) = \gamma(X)\gamma(Y) \sum_{x \in X} \sum_{y \in Y} s(x, y), \tag{5}$$

where $s : \mathcal{X}_{el} \times \mathcal{X}_{el} \to \mathbb{R}$ is a similarity function that acts on a pair of set elements. Normalization factors $\gamma(\cdot)$ ensure unit self-similarity, *i.e.* $\gamma(X) = \sqrt{\sum_{x \in X} \sum_{z \in X} s(x, z)}$. For simplicity, normalization factors are often skipped and the match kernel is denoted as

$$S_m(X, Y) \propto \sum_{x \in X} \sum_{y \in Y} s(x, y). \tag{6}$$

In the following, methods of both families, *i.e.* single vector representation and match kernels, are discussed. Emphasis is given in cases where there is an (approximate) equivalence between the two approaches, while the proposed design choices to make the equivalence possible are analyzed.

2.1 From local descriptors to image similarity

Assume that set X corresponds to a set of local features, also called keypoints, [58, 57], or image regions [14], that are used to represent the image. Each feature $x \in X$ is accompanied by a local descriptor $\mathbf{x}_a \in \mathbb{R}^d$ capturing its appearance, and geometric properties such as 2D position in the image given by $x_x \in \mathbb{R}^+$ and $x_y \in \mathbb{R}^+$, scale $x_s \in \mathbb{R}^+$, and dominant orientation $x_o \in [0, 2\pi]$.

A common example are local features and descriptors, such as SIFT [56], used in traditional computer vision methods. Other detectors fit a local affine frame [18] at each feature but such as case is not considered in the following. Deep learning methods are also used perform local feature detection [3] or refinement [60].

The example, that almost dominates nowadays, is that of deep local features obtained with a Fully Convolutional Network (FCN) which maps an input image of size $W \times H \times 3$ to a 3D activation tensor of size $w \times h \times d$. In that case, this 3D tensor is seen as a dense set of local descriptors, with $w \times h$ local descriptors in total. The corresponding image region or local feature corresponding to each local descriptor is the region formed by the receptive field of the activations. Position is further used to describe the local features, and scale which varies only if input images at different resolutions are used, while dominant orientation is typically not applicable in this setup. This setup resembles dense-SIFT [98, 55] where SIFT descriptors are extracted from overlapping image regions that are sampled with a uniform step at different image resolutions. If a detector is involved, then a subset of the $w \times h$ positions from the 3D tensor are attained.

In the following, we discuss the model design choices to either estimate visual similarity with match kernels or to generate single vector representations or both when there is an equivalence. The discussed representation and similarity models are applicable to both traditional and deep local features. However, each method was typically proposed in one context or the other. When it comes to deep representation models, only their design is discussed in this section and not their training.

2.1.1 From a local descriptor set to a global descriptor: appearance

SPoC (as background). In order to fully define a match kernel in (6), we need to define the *local similarity function* $s(\cdot)$. When dealing only with the appearance of local features, it becomes equivalent to $s(x, y) = s_a(\mathbf{x}_a, \mathbf{y}_a)$, where $s_a : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$. A simple choice is a dot product similarity, *i.e.* $s_a(\mathbf{x}_a, \mathbf{y}_a) = \mathbf{x}_a^\top \mathbf{y}_a$. Due to the linearity of the local similarity function, the following equivalence holds:

$$S_m(X, Y) \propto \sum_{x \in X} \sum_{y \in Y} \mathbf{x}_a^\top \mathbf{y}_a = \left(\sum_{x \in X} \mathbf{x}_a \right)^\top \sum_{y \in Y} \mathbf{y}_a \propto f(X)^\top f(Y) = S_g(X, Y), \tag{7}$$

where proportionality is used because normalization is skipped, *i.e.* $f(X) = \sum_{x \in X} \mathbf{x}_a / \| \sum_{x \in X} \mathbf{x}_a \|$. As a result, the match kernel estimation is obtained, equivalently, with a simple dot product between two vectors. The latter is efficiently computed, while the former is interpreted as cross-matching all pairs of regions and accumulating the individual similarities. Note that $D = d$ in the discussed example.

The generated descriptor is known as SPoC [2] in the literature. It is proposed in the context of deep networks, *i.e.* SPoC is constructed on top of deep local descriptors. Note that SPoC, constructed by simple sum pooling of deep local descriptors, performs well due to the discriminative power of CNN activations. In contrast, such an approach on top of SIFT descriptors fails as shown in a comparison included in our earlier work [10]. Even though SPoC is not our contribution it is included in this manuscript to allow a smooth transition to other approaches and allow for comparison.

MAC. Non-linear operations do not allow the aforementioned equivalence. They are however used in the literature and constitute high performing variants. This is the case for the Maximum Activation of Convolutions (MAC) [14], originally proposed by Razavian *et al.* [76]. It is given by

$$f(X) \propto \max_{x \in X} \mathbf{x}_a, \tag{8}$$

where $\max$ denotes component-wise maximum; sum-pooling of SPoC is replaced by max-pooling. The maximum activation is attained per dimension, *i.e.* feature channel, of the corresponding 3D activation tensor, and is stored as a component of the final representation. Therefore, each MAC descriptor component gets associated with an image region, *i.e.* the receptive field of that activation. As a consequence, dot-product comparison of two such descriptors from different images, can be interpreted as one-to-one region matching (see Figure 2). The number of correspondences is equal to d and the contribution per correspondence equals the product of the respective scalar activations.

In our own work, the MAC representation is employed to go beyond the comparison of two images as a whole. We compare an image depicting a cropped object with another image where the object of interest covers only a small part of it. Object localization is performed by efficiently generating MAC descriptors

for all possible rectangle regions and comparing to the MAC descriptor of the cropped object; the region of maximum similarity is chosen. We make use of the asymptotic behaviour of the generalized mean

$$\lim_{\alpha \to \inf} \left(\frac{1}{|Z|} \sum_{z \in Z} z^\alpha \right)^{1/\alpha} = \max_{z \in Z} z. \tag{9}$$

In practice the maximum operator is approximated well enough for values of α equal to 5 or 10. The benefit is that the maximum is approximated with a summation and the use of integral images is now enabled. Integral images for max pooling allow us to perform an efficient localization procedure which is then used for image retrieval and re-ranking of top-ranked images according to the localized objects and the corresponding similarity. MAC is proposed and used with deep local descriptors.

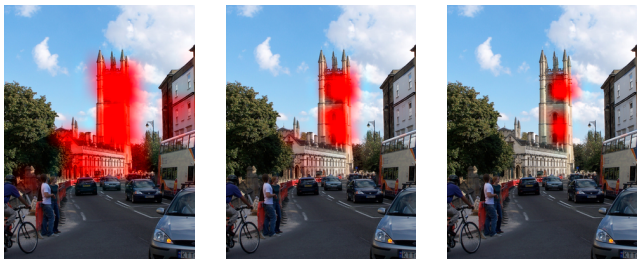
Figure 2: We visualize the regions (receptive fields) related to the 5 MAC components that contribute the most to the image similarity. Each displayed receptive field corresponds to the maximum response of a feature channel. A different color is used for each feature channel, while different feature channels are shown for each image pair.

Publications: [14] in Appendix I

GeM. We use the generalized-mean and propose the GeM descriptor [72] given by

$$f(X) \propto \left(\frac{1}{|X|} \sum_{x \in X} \mathbf{x}_a^\alpha \right)^{\frac{1}{\alpha}}, \tag{10}$$

where $\mathbf{z}^\alpha$ denotes component-wise power. This is a generalization of the previously existing pooling schemes and descriptors. For $\alpha = 1$ this becomes equivalent to average pooling and SPoC, while for $\alpha \to \inf$ this tends to max pooling and MAC, as shown in (9). Figure 3 shows the effect of p in the form of an image heat-map. GeM is proposed and used with deep local descriptors, while parameter α can be fixed or learned jointly with the FCN that generates the local descriptors. GeM consistently improves the matching performance compared to earlier descriptors at no extra computational cost. It is widely adopted in research competitions for landmark recognition on kaggle organized by Google [1].



$p = 1$ $p = 3$ $p = 10$

Figure 3: Visualization of feature map activations raised to the power of α and then projected on the original image. Case $p = 1$ corresponds to SPoC, and larger p corresponds to GeM which becomes more localized.

Publications: [70, 72] in Appendices XXIII and II

[1] https://www.kaggle.com/c/landmark-recognition-challenge

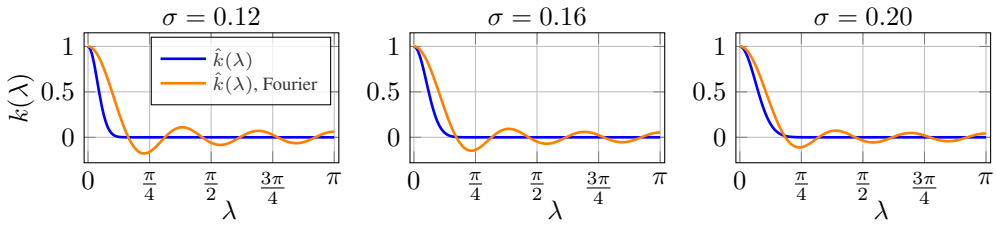


Figure 4: Examples of RBF kernels of different scales and the corresponding approximation

R-MAC. A hybrid scheme that we propose [14] is termed as regional MAC, shortly R-MAC. Now, the regions considered are square and include several locations of the 3D activation tensor. They are sampled uniformly over the image with a fixed step and at different sizes. A MAC descriptor is obtained per region, and all MAC descriptors of an image are sum-pooled to a single vector representation. This scheme combines max-pooling and sum-pooling. R-MAC is proposed by our work as a way of transfer learning for pre-trained networks. It is later fine-tuned in an end-to-end manner in the work Gordo *et al*. [31].
Publications: [14] in Appendix I

2.1.2 From a local descriptor set to a global descriptor: appearance & geometry

The core idea in this work [10] is to jointly encode the local descriptors, *i.e*. appearance, of a local feature and its geometuc shape, such as the dominant orientation which is the first case considered in this section.

To achieve that we rely on a shift-invariant kernel function $k_o : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$. There exists a corresponding symmetric, non-increasing function $\hat{k}_o : \mathbb{R} \to \mathbb{R}$ such that $\hat{k}_o(x_o - y_o) = \hat{k}_o(y_o - x_o) = k(x_o, y_o)$. Examples for $\hat{k}_o(\cdot)$ are the RBF-kernel and the Von Misses function for periodic domains. We additionally rely on a corresponding embedding $\phi_o : \mathbb{R} \to \mathbb{R}^n$ such that $\phi_o(x_o)^\top \phi_o(y_o) \approx k_o(x_o, y_o)$. The construction of this embedding uses the Fourier series approximation of $\hat{k}_o$, but details are skipped in this summary (see Figure 4 for an example). The larger the dimensionality n is the better the approximation. The useful property is that a non-linear kernel is approximated by a linear operation, *i.e*. dot product between two vectors.

Moreover, embedding function $\psi : \mathbb{R}^d \to \mathbb{R}^{d'}$ is used to represent an original local descriptor $\mathbf{x}_a$ by $\psi(\mathbf{x}_a)$, where d' is typically larger than d. The goal of this embedding is to map descriptors to a higher dimensional space that is more discriminative and better balances interferences between descriptors of different images. Handling the interferences is beneficial in the match kernel formulation of (6) where all possible feature combinations are considered; accumulating large values for non-matching features can dominate and distract the final similarity. Common examples of $\psi(\cdot)$ are the embeddings involved in quantization based approaches where the descriptor space is quantized into k cells, commonly known as visual words. In the case of Bag-of-Words (BoW) [88, 21] the embedding is an one-hot encoding of the visual word that the local descriptors belongs to. In the case of VLAD [44, 49], which is an extension of BoW, the embedding has size equal to kd; there are k sub-spaces, one per visual word. Residual vectors are encoded, *i.e*. the vector pointing from the visual world centroid to the local descriptor. The residual is placed into the corresponding sub-vector, while all other $k - 1$ sub-vectors are full of zeros.

The mixed-product property of the kronecker product enables the joint representation by using $\psi(\mathbf{x}_a) \otimes \phi_o(x_o)$. Comparing two features x and y through their joint representation, it holds that

$$(\psi(\mathbf{x}_a) \otimes \phi_o(x_o))^\top \psi(\mathbf{y}_a) \otimes \phi_o(y_o) \quad = \quad (11)$$
$$(\psi(\mathbf{x}_a)^\top \psi(\mathbf{y}_a)) \cdot (\phi_o(x_o)^\top \phi_o(y_o)) \quad \approx \quad (12)$$
$$\psi(\mathbf{x}_a)^\top \psi(\mathbf{y}_a) \cdot k_o(x_o, y_o). \quad (13)$$

As a result, the product between appearance similarity and geometry similarity with a non-linear kernel (13)

is approximated with a linear operation (11). Therefore, the corresponding match kernel becomes

$$\begin{aligned} S_m(X,Y) \propto & \quad \textstyle\sum_{x\in X}\sum_{x\in Y} \psi(\mathbf{x}_a)^\top \psi(\mathbf{y}_a)\cdot k_o(x_o,y_o) && (14)\\ \approx & \quad \textstyle\sum_{x\in X}\sum_{x\in Y} (\psi(\mathbf{x}_a)\otimes\phi_o(x_o))^\top \psi(\mathbf{y}_a)\otimes\phi_o(y_o) && (15)\\ = & \quad \textstyle\sum_{x\in X} (\psi(\mathbf{x}_a)\otimes\phi_o(x_o))^\top \sum_{x\in Y} \psi(\mathbf{y}_a)\otimes\phi_o(y_o) && (16)\\ = & \quad f(X)^\top f(Y) && (17)\\ \propto & \quad S_g(X,Y). && (18) \end{aligned}$$

The achievement is that we are able to construct the global descriptor per image

$$f(X) = \frac{\sum_{x\in X}\psi(\mathbf{x}_a)\otimes\phi_o(x_o)}{\|\sum_{x\in X}\psi(\mathbf{x}_a)\otimes\phi_o(x_o)\|}, \tag{19}$$

and then compare the two images with a simple dot product between the two global descriptors (17). The result approximates the more costly match kernel that enumerates all feature pairs and accumulates the feature pair similarities (14). The feature similarity is high if both the appearance similarity is high, as indicated by the local descriptors, and the dominant orientations are close to each other. The resulting global descriptor, due to the joint encoding, is orientation covariant. Therefore, the resulting descriptor is more discriminative than the one of only encoding appearance, but it is not invariant to the dominant orientation and, consequently, object rotations. The lack of invariance is re-obtained in Section 2.7. As a reference, the appearance-only variant is given by

$$f(X) = \frac{\sum_{x\in X}\psi(\mathbf{x}_a)}{\|\sum_{x\in X}\psi(\mathbf{x}_a)\|}, \tag{20}$$

corresponding to BoW, VLAD, and Fisher vectors [68, 67], depending on the definition of $\psi(\cdot)$.

An extension to encoding of two geometric properties jointly is proposed in our later work [7] with an application to encoding feature 2D position in the image. The joint encoding is able to obtain 30% relative improvement on top of VLAD with a corresponding dimensionality increase ($\times 5$ or $\times 7$) or about 10% for the same dimensionality.

Publications: [10, 7] in Appendices III and IV

2.1.3 From a local descriptor set to image similarity

ASMK: aggregated selective match kernel. The equivalence between global descriptors and match kernels is not possible when the local similarity function for individual features is not linear. However, there are performance benefits by using a non-linear similarity function for comparing high-dimensional local descriptors and this is the contribution of the Selective Match Kernel (SMK) and the Aggregated Selective Match Kernel (ASMK) [5, 6].

Even though approximations of non-linear kernels in high dimensional spaces that would allow this equivalence exist in the literature [15, 74], they require very high dimensinality of the embeddings for good approximation. This makes them not applicable to representations that are used for instance-level search and discovery which are the main applications of our contributions.

The descriptors are quantized by k-means quantizer $q : \mathbb{R}^d \to \mathcal{C} \subset \mathbb{R}^d$, where $\mathcal{C} = \{\mathbf{c}_1, \ldots, \mathbf{c}_k\}$ is a codebook comprising $|\mathcal{C}|$ vectors (visual words). Descriptor $\mathbf{x}_a$ is assigned to its nearest visual word $q(\mathbf{x}_a)$. We denote by $\mathcal{X}_c = \{\mathbf{x}_a \in \mathcal{X} : q(\mathbf{x}_a) = \mathbf{c}\}$ the subset of descriptors in $\mathcal{X}$ that are assigned to visual word $\mathbf{c}$, and by $\mathcal{C}_X$ the set of all visual words that appear in X. Descriptor $\mathbf{x}_a$ is mapped to $\psi(\mathbf{x}_a) \in \mathbb{R}^d$ which is the normalized residual vector, *i.e.* $\psi(\mathbf{x}_a) = {}^{r(\mathbf{x}_a)}/_{\|r(\mathbf{x}_a)\|}$, with $r(\mathbf{x}_a) = \mathbf{x}_a - q(\mathbf{x})$. For efficiency, a binarized variant exists that uses the sign function instead of ℓ_2-normalization.

The SMK similarity of two images, represented by X and Y, respectively, is estimated by cross-matching all pairs of local descriptors with match kernel

$$S_{\text{SMK}}(X,Y) \propto \sum_{x\in X}\sum_{y\in Y} [q(\mathbf{x}_a) = q(\mathbf{y}_a)] k(\psi(\mathbf{x}_a), \psi(\mathbf{y}_a)), \tag{21}$$

where $[\cdot]$ is the Iverson bracket. Function $k : \mathbb{R}^d \times \mathbb{R}^d \to [0, 1]$ is given by

$$k(\psi(\mathbf{x}_a), \psi(\mathbf{y}_a)) = \begin{cases} \left(\psi(\mathbf{x}_a)^\top \psi(\mathbf{y}_a)\right)^\alpha, & \psi(\mathbf{x}_a)^\top \psi(\mathbf{y}_a) \geq \tau \\ 0, & \text{otherwise,} \end{cases} \tag{22}$$

where $\tau \in [0, 1]$ is a threshold parameter, and exponent α controls the shape of the local similarity function. Only descriptor pairs that are assigned to the same visual word contribute to the image similarity in (21). In practice, not all pairs need to be enumerated and image similarity is equivalently given by

$$S_{\text{SMK}}(X, Y) \propto \sum_{\mathbf{c} \in \mathcal{C}_X \cap \mathcal{C}_Y} \sum_{\mathbf{x}_a \in X_\mathbf{c}} \sum_{\mathbf{y}_a \in Y_\mathbf{c}} k(\psi(\mathbf{x}_a), \psi(\mathbf{y}_a)), \tag{23}$$

where cross-matching is only performed within visual words that are common for the two images.

ASMK first *aggregates* the local descriptors assigned to the same visual word into a single vector. This is performed by $\Psi(X_c) = \frac{\left(\sum_{x \in X_c} r(\mathbf{x}_a)\right)}{\left\|\left(\sum_{x \in X_c} r(\mathbf{x}_a)\right)\right\|}$, with $\Psi(X_c) \in \mathbb{R}^d$. The counterpart binarized representation exists in the aggregated variant too. Image similarity in ASMK is given by

$$S_{\text{ASMK}}(X, Y) \propto \sum_{\mathbf{c} \in \mathcal{C}_X \cap \mathcal{C}_Y} k(\Psi(X_c), \Psi(Y_c)). \tag{24}$$

This is computationally and memory-wise more efficient than SMK. In practice, it is known to perform better due to handling the burstiness phenomenon [43]. Figure 5 illustrates several examples of features that are aggregated. They commonly correspond to repeated structures and textured regions. Such bursty features appear in most urban images, and their matches usually dominate the image level similarity. ASMK handles this by keeping only one representative vector out of all bursty descriptors. It is faster, better, and less memory demanding than SMK.

ASMK was the state-of-the-art retrieval approach when it was proposed with classical local features, *i.e.* SIFT, but it achieved top performance even when used with modern deep local features, namely DELF [66] in our work [73].

Publications: [6] in Appendix V

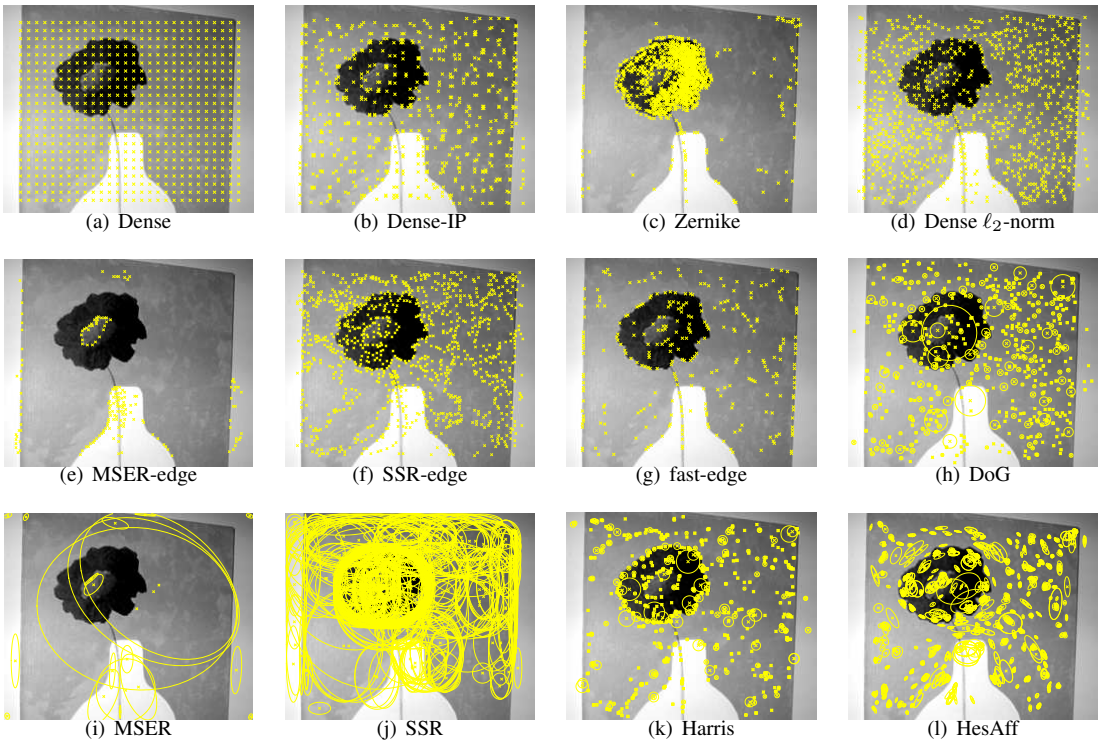
Figure 5: Examples of features mapped to the same visual word (finally aggregated). Each visual word is drawn in a different color. Top 25 visual words, based on the corresponding number of features, are drawn.

2.2 Dense local feature detection

Global image descriptors generated by aggregation of local feature representation have the useful property that the memory requirements of the final representation and the computational cost of the image-to-image similarity does not depend on the number of local features. In our work [41], we take advantage of this property and propose a set of local feature detectors with dense response. In contrast to conventional detectors which prefer a very localized and sparse result, we focus on denser detection such as the one that is repeated across edges aiming to better represent local structures in the image.

We propose different strategies derived from standard interest point detectors (Harris, Hessian and DoG) to extract patches densely. In particular, we modify the detection process by relaxing the standard local maxima criterion so that it focuses on edges and not only corners. This, jointly with the optimization of the scaling factor, is shown to be a key to achieve higher performance. We depart from the typical choice of fitting an ellipse to describe a region of interest extracted by blob detectors (MSER or super-pixels). Instead, we sample patches at several scales along the region's borders. This increases the performance significantly when considering a large number of patches per image. We also show that sampling on the edges produced by a state-of-the-art edge detector offers competitive performance. Finally, we propose two novel response filters to select the patch locations. First, we propose to use a bank of Zernike polynomials as detectors. These filters have been proposed to construct descriptors, but to our knowledge not as detectors. Our second strategy is descriptor-oriented: the response for each pixel is simply the norm of local descriptor associated with the patch centered at this location. These two new approaches appear to perform best in most cases of our experiments. Figure 6 shows detection example for the different variants proposed.

Publications: [41] in Appendix VI



(a) Dense (b) Dense-IP (c) Zernike (d) Dense ℓ_2-norm

(e) MSER-edge (f) SSR-edge (g) fast-edge (h) DoG

(i) MSER (j) SSR (k) Harris (l) HesAff

Figure 6: Visualization of regions detected by different methods. For methods with predefined scales, we visualize the center of points detected on the first scale (a)-(g), while for the ones that perform scale selection we draw the corresponding ellipses or circles (h)-(l).

2.3 Deep local features and descriptors

Training local feature detectors and descriptors typically requires either labeling at the level of image regions or pixels [27], which is costly to obtain, or some sort of self-supervision [25]. We rely on image-level labels, which are more accessible and propose the HOW architecture [12] which generates a global image descriptor during training in a metric learning fashion. During testing, the architecture is re-purposed to detect only the most useful local features while the local descriptors are provided by the activations of internal components of the network.

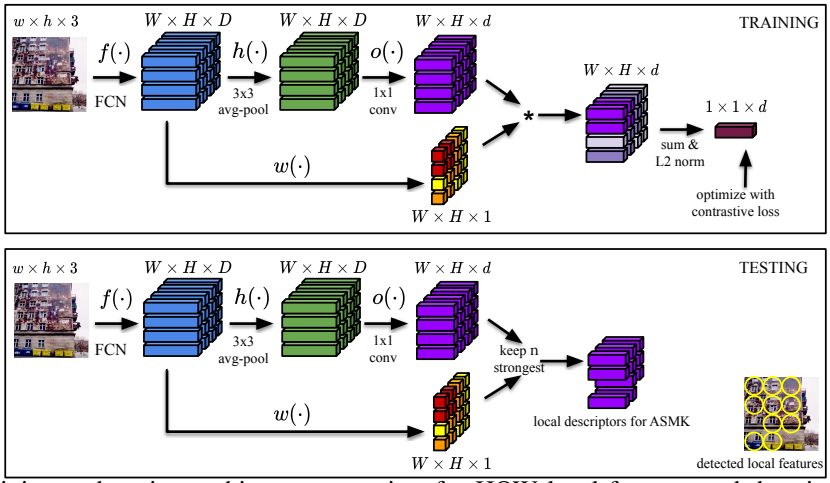


Figure 7: Training and testing architecture overview for HOW local features and descriptors. A global descriptor is generated for each image during training and optimized with contrastive loss and image-level labels. During testing the strongest local descriptors (features), according to the attention map, are kept to represent the image. Then, these are used with ASMK for image search.

We modify the simple match kernel in (7) and introduce a weight per local feature. The resulting match kernel is given by

$$S_m(X,Y) \propto \sum_{x\in X}\sum_{y\in Y} w(x)w(y)\mathbf{x}_a^\top \mathbf{y}_a \tag{25}$$

$$= \Big(\sum_{x\in X} w(x)\mathbf{x}_a\Big)^\top \sum_{y\in Y} w(y)\mathbf{y}_a \tag{26}$$

$$\propto f(X)^\top f(Y) \tag{27}$$

$$= S_g(X,Y). \tag{28}$$

The weighted match kernel (25) is equivalent to the global descriptor based approach (27), where the global descriptor in this case is a weighted summation of local descriptors (26).

During training with global descriptors, the feature weight is indirectly used to weigh the contribution of each feature descriptor in cross-matching. This way, the impact of the weak features is limited during the training. During test time, only the strongest features are selected according to this weight and the corresponding local descriptors are used with ASMK for retrieval.

The feature weight is provided by an attention function, something originally proposed by DELF [66] as a learnable function. In our approach we use a non-parametric function, whose output is simply the ℓ_2 norm of the respective local descriptor shown to work better in practice. The proposed architecture is shown in Figure 7.

The proposed local descriptors when used with ASMK are shown to outperform global descriptors and DELF descriptors for instance-level search and instance-level knn classification. Such a pipeline is extended and improved in a very recent work that uses transformer modules and further boosts the performance [94].
Publications: [12] in Appendix VII

2.4 Local-patch kernel-descriptors

We now consider the representation of a local-patch, *i.e.* a patch as the ones provided by invariant local feature detectors. The resulting representation is called local descriptor. We rely on kernel-based approximation that follows the approach presented in Section 2.1.2. The following two variants are proposed.

2.4.1 Crafted with pixels

Patch X is seen as a set of pixels $x \in X$. We consider a pixel x to be associated with various attributes, *i.e.* coordinates x_x, x_y in Cartesian coordinate system, coordinates x_ρ, x_φ in polar coordinate system, pixel

gradient magnitude x_m, and pixel gradient angle x_θ. It holds for angles that x_φ, $x_\theta \in [0, 2\pi]$, for distance from the center x_ρ that it is normalized to $[0, 1]$, while for coordinates that $x_x, x_y \in \{1, 2, \dots, W\}$ for patches of size $W \times W$. The gradient angle is expressed w.r.t. the patch orientation, *i.e.* x_θ directly, or w.r.t. to the position of the pixel, *i.e.* $x_{\tilde{\theta}} = x_\theta - x_\varphi$.

We use the following match kernel on patches, which can be approximated by dot product between two local descriptors for appropriate design of kernel function $k(\cdot)$. The match kernel is given by

$$S_m(X, Y) = \sum_{x \in X} \sum_{y \in Y} k(x, y) \approx \sum_{x \in X} \sum_{y \in Y} h(x)^\top h(y) = \sum_{x \in X} h(x)^\top \sum_{y \in Y} h(y). \tag{29}$$

Composing kernel $k(\cdot)$ as a product of kernels over different attributes enables easy design of various patch similarities. Correspondingly, different choices define different local descriptors $f(X) = \sum_{x \in X} h(x)$. All attributes are matched by the Von Mises kernel. We focus on the two following match kernels over patches. One in *polar* coordinates

$$S_{m-polar}(X, Y) = \sum_{x \in Y} \sum_{y \in Y} x_g y_g \sqrt{x_m} \sqrt{y_m} k_\varphi(x_\varphi, y_\varphi) k_\rho(x_\rho, y_\rho) k_{\tilde{\theta}}(x_{\tilde{\theta}}, y_{\tilde{\theta}}), \tag{30}$$

and one in *cartesian* coordinates

$$S_{m-cart}(X, Y) = \sum_{x \in Y} \sum_{y \in Y} x_g y_g \sqrt{x_m} \sqrt{y_m} k_\varphi(x_x, y_x) k_y(x_y, y_y) k_\theta(x_\theta, y_\theta), \tag{31}$$

where $x_g = \exp(-x_\rho^2)$ gives more importance to central pixels, in a similar manner to SIFT. These kernels cross-match all pixels between two patches and accumulate the respective similarities. If two pixels are in nearby positions, and they have gradients with similar angles, then similarity is high. Similarity is further weighted according to the magnitude of the gradient on these pixels. This is an intuitive and interpretable way to compare two patches. The corresponding kernel-based descriptors that approximates this result are given by

$$f(X) \propto \sum_{x \in X} x_g x_m \phi_\varphi(x_\varphi) \otimes \phi_\rho(x_\rho) \otimes \phi_{\tilde{\theta}}(x_{\tilde{\theta}}) \tag{32}$$

$$f(X) \propto \sum_{x \in X} x_g x_m \phi_x(x_x) \otimes \phi_x(y_y) \otimes \phi_\theta(x_\theta). \tag{33}$$

The different embeddings $\phi(\cdot)$ are constructed as in Section 2.1.2 such that $\phi(x_\rho)^\top \phi(y_\rho) = k_\rho(x_\rho, y_\rho)$.

The descriptor based on polar coordinates was initially proposed [16], and then it was extended to the one based on cartesian coordinates and their combination [62]. The combination enjoys multiple kinds of invariance properties. It is common to post-process visual descriptor with PCA-whitening [45] which is shown to drastically improve the performance. The impact of whitening in high dimensional spaces is not easy to interpret. In our work on local-patch descriptors and due to the connection with the match kernel we manage to back-project the effect of whitening to the original patch-pixel space [61] and discuss that the effect is semantically meaningful. This hand-crafted local descriptor was shown to outperform early deep local descriptors despite not requiring any training.

Publications: [16, 62, 61] in Appendices VIII, IX, and X

2.4.2 Learned with CNN activations

In an extended version [63], the patch is fed to an FCN network and the output is a 3D activation tensor of size $w \times w \times d$. Pixels x are now defined on this 3D tensor X, with $w \times w$ pixels in total, while the respective representation of the region that corresponds to x is denoted by $\mathbf{x}_a$; it is a d-dimensional vector from position (x_x, x_y) of the 3D tensor. Similar to the crafted example, we rely on kernel encoding to encode the positions of each pixel. However, we use the appearance representation $\mathbf{x}_a$ provided by the FCN and jointly encoded with the position by

$$f(X) \propto \sum_{x \in X} \phi_x(x_x) \otimes \phi_x(y_y) \otimes \mathbf{x}_a. \tag{34}$$

As a result, the corresponding match kernel, which is skipped in this manuscript, is accumulating similarities for all pixels of two features, while each similarity is the product of appearance-based similarity and spatial-based similarity. This descriptor was built on top of HardNet [59] which uses conventional deep learning building blocks; our descriptor is a combination of kernel-based methods and deep networks. It was shown to outperform HardNet by a small but consistent margin. Our analysis shows how a conventional architecture with tensor flattening and a Fully-Connected layer also encodes activation positions in the 3D tensor but is possibly over-parametrized for this purpose.
Publications: [63] in Appendix XI

2.5 Cross-modal visual representation

Many real-world applications include an asymmetric setup where examples from two different modalities need to be compared. Therefore, visual similarity estimation is required in a cross-modal way, and there is the need to map examples from two modalities in the same representation space. This is the case for cross-modal image retrieval. In particular, sketch-based image retrieval where the query is a hand-drawn sketch and the goal is to retrieve images in a shape-based way where the depicted objects resemble the drawn sketch.

Our work proposes both crafted and CNN-based learned representation for this task. In both cases bridging the domain-gap is facilitated with a fist step of performing edge detection on images. Therefore, an image is now represented as an edge-map, while the sketch is a binary edge-map.

2.5.1 Asymmetric feature maps for kernel descriptors

Consider a binary sketch as a set of contour points, *i.e*. the set of pixels X that lie on the contour. A *contour pixel* $x \in X$ is represented by its position and its gradient magnitude and angle. For real images, the contour parameters are obtained from an edge detector. For sketches, the magnitude is equal to one for all contour pixels since the sketch is equivalent to a binary edge-map. This setup is equivalent to that of local patch descriptors generated by representation of pixel positions and gradients with kernels in Section 2.4. We follow the same kernel-based approach to generate a descriptor for an edge-map. To allow multi-scale representation and matching, the sketches, *i.e*. the queries, are processed at three different resolutions. But the shape, *i.e*. σ of the employed RBF kernel, needs to be adjusted according to the resolution; smaller objects needs a more precise matching over the spatial dimensions. We propose a new asymmetric encoding, termed asymmetric feature maps (AFM), that allows to encode the different kernels only on the query side, while storing only a single descriptor per database image. It is the query descriptor that defines the kernel that is used. Evaluation of similarity over multiple translations enables to localize the sketch and this is performed according to the trigonometirc polynomial that is presented Section 2.7. In the application of sketch-based image retrieval this approach performs multi-scale sketch localization while using a single global descriptor per database image. Examples are shown in Figure 8. Potential applications of AFM [8, 9] go beyond this specific task.
Publications: [8, 9] in Appendices XII and XIII

2.5.2 Edge-MAC

In this work [71] we cast shape matching as metric learning with convolutional networks. Metric learning requires labels for image pairs. A pair is labeled positive or negative depending on whether the same object is depicted or not. However, shape-based annotation is not easy to obtain. Instead we rely on (automatically) annotated image datasets (see Section 4.2.1) to obtain image pair labels. We transfer these labels to the edge-maps of the original images as well; if the same object is depicted, we assume that a similar shape is depicted in the edge-maps too.

We proceed with metric learning with contrastive loss on input edge-maps which are fed to an FCN architecture tailored for this input. We introduce an edge filtering layer to address two frequent issues with edge responses (see Figure 9). First, the background often contains close-to-zero responses, which typically introduce noise into the representation. This issue is commonly handled by thresholding the response function. Second, the strength of the edges provides ordering, i.e. higher edge response implies

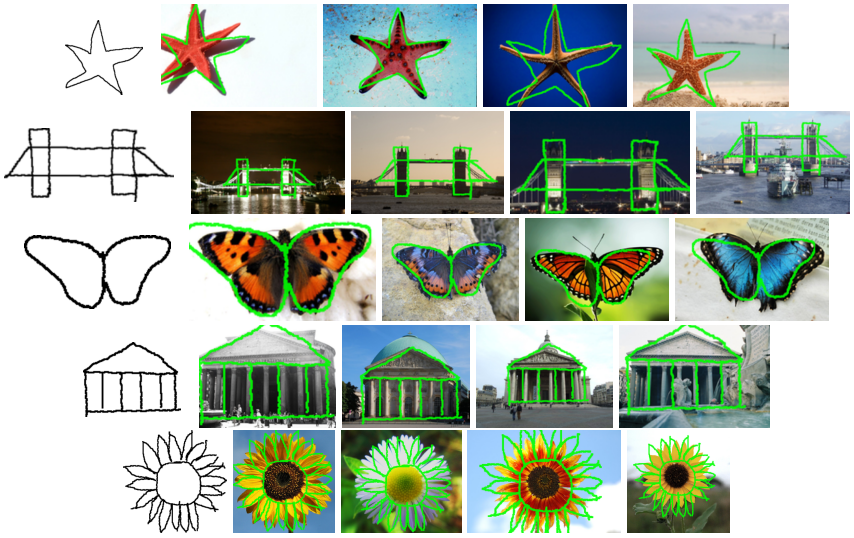


Figure 8: Scale and translation invariant query-by-sketch retrieval. An example of sketch queries and top-retrieved images with the sketch localization overlaid in green color.

that the edge is more likely to be present, however its value typically does not have practical interpretation. Prior to the first convolution layer, a continuous and differentiable function is prepended. This layer is trained together with the rest of the network to transform the edge detector output with soft thresholding by a sigmoid and power transformation. The FCN includes global max pooling at the end, resulting in what we call *edge-MAC* descriptor [71].

The learned representation is evaluated on a range of different tasks, including sketch-based image retrieval. It is also shown effective for domain generalization across domains where the shape matters, *e.g.* across natural images, cartoons, artworks, sketches. In a number of problems, the colour and/or the texture is not available or misleading. Three examples are shown in Figure 10. For sketches or outlines, there is no color or texture available at all. For artwork, the colour and texture is present, but often can be unrealistic to stimulate certain impression rather than exactly capture the reality. Finally, in the case of extreme illumination changes, such as a day-time versus night images, the colors may be significantly distorted and the textures weakened. On the other hand, the image discontinuities in colour or texture, as detected by modern edge detectors, and especially their shapes, carry the information about the content, independent of, or insensitive to, the illumination changes, artistic drawing and outlining

Publications: [71] in Appendix XIV

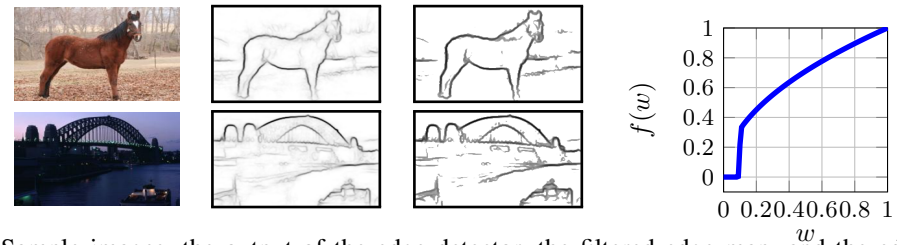


Figure 9: Sample images, the output of the edge detector, the filtered edge map, and the edge-filtering function.

2.6 From an image set to a global descriptor

In this method [40], a single vector representation is obtained for a set of images X, with $|X| = m$. Set elements $x \in X$ are images in this case, and each image is initially represented by vector $\mathbf{x} \in \mathbb{R}^d$. We follow the *memory vector* construction strategy proposed by Iscen *et al*. [42]. The set of image vectors

Figure 10: Three examples where **shape** is the only relevant information: sketch, artwork, extreme illumination conditions. Top retrieved images from the Oxford Buildings dataset [69]: CNN with an RGB input [70] (left), and our shape matching network (right). Query images are shown with black border.

are stacked in a $d \times m$ matrix $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_m]$, and the vector representation for the set is defined by assuming linearly independent columns and using the Moore-Penrose pseudo-inverse $\mathbf{X}^+$ [75]. It is given by

$$f(\mathbf{X}) = (\mathbf{X}^+)^\top \mathbf{1}_n = \mathbf{X}(\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{1}_n. \tag{35}$$

Such representation is theoretically optimized for high dimensional spaces and performs better in practice.

In a retrieval scenario, and in contrast to previous works that aggregate the image vectors only on the dataset side [42] or only on the query side [82], we rather do it for both. This requires that the query is also defined by a set of images, which is the case in location recognition with panorama images. Each location is represented by a set of images offering a 360 degree view. A realistic scenario of this context is autonomous driving and auto-localization where the query is defined by such a set of images.

Assume that m images in a database location are represented by $d \times m$ matrix $\mathbf{X}$ and that k images in the query location by $d \times k$ matrix $\mathbf{Y}$. Analyzing the similarity between the two memory vectors gives

$$S_g(\mathbf{X}, \mathbf{Y}) = f(\mathbf{X})^\top f(\mathbf{Y}) = \mathbf{1}_m^\top G_{\mathbf{X}}^{-1} \mathbf{X}^\top \mathbf{Y} G_{\mathbf{Y}}^{-1} \mathbf{1}_k, \tag{36}$$

where $G_{\mathbf{X}} = \mathbf{X}^\top \mathbf{X}$ is the Gram matrix for $\mathbf{X}$. Compared to simple averaging (note that $\mathbf{1}_m^\top \mathbf{X}^\top \mathbf{Y} \mathbf{1}_k$ is the same as in (7)), *e.g.* as SPoC does with local features, the summation after cross-matching is weighted now, and the weights are given by $G_{\mathbf{X}}^{-1}$ and $G_{\mathbf{Y}}^{-1}$. This is interpreted as "democratizing" [50] the result of cross-matching; the contribution of vectors that are similar within the same set are down-weighted, just as in handling the burstiness phenomenon for local descriptors [50].

This approach when used for location recognition with panoramic views, *i.e.* group of images taken at each location that offer 360 degrees view, achieved the perfect performance for the first time on Pittsburg dataset [90].

Publications: [40] in Appendix XV

2.7 Geometric alignment with kernel descriptors

Kernel-based descriptors presented in Sections 2.1.2 and 2.4 are useful for encoding geometric attributes of features jointly with their appearance. Such geometric attributes are typically measured in a fixed coordinate system, *e.g.* the 2D location w.r.t. the image-aligned coordinate system whose origin is at the top left image corner or the dominant orientation w.r.t. to the orientation of the horizontal image axis. As a result, the generated descriptor is equivariant to the underlined geometric attribute and not invariant. Such equivariance results in more discriminative descriptors. Herein we discuss the proposed approach to efficiently offer invariance by evaluating similarity for multiple shifts of the geometric attribute, *i.e.* 1D or 2D translations if position is encoded, or rotations if orientation is encoded.

Due to the use of embeddings that perform a Fourier series based approximation of the respective kernels, we are able to show that similarity under shift Δu is equivalent to trigonometric polynomial

$$S(X_{\Delta u}, Y) = \sum_{\omega \in \Omega} \left(\beta_\omega \cos(\omega \Delta u) + \gamma_\omega \sin(\omega \Delta u)\right), \tag{37}$$

where Ω is the set of Harmonic frequencies used in the Fourier series, typically with $|\Omega|$ equal to 3 or 4, and u represents the geometric attribute whose invariance is desired. The scalar coeficients β_ω and γ_ω are obtained by dot-products in sub-spaces (sub-vectors) of the original descriptor. Details are skipped and can be found in the original publication [10]. It is worth nothing that the cost to estimate the coefficient values is about double that of estimating dot product for the original descriptors; *i.e.* corresponding to no shift. Then, (37) allows to evaluate similarity for multiple shifts in a very efficient way. Generalization to 2D shifts for 2 geometric properties is straightforward and enables 2D translations and alignment [7]. Figures 11 and 12 depict a matching example and the corresponding estimated similarity under multiple image rotations and rotations, respectively.

Publications: [10, 7] in Appendices III and IV

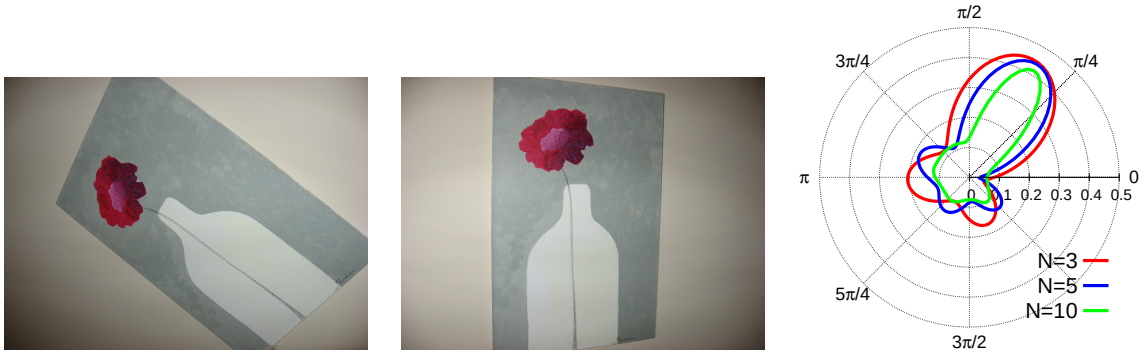


Figure 11: Matching example of two images and similarity estimation for all possible image rotations. The image on the left undergoes rotation. Image similarity versus rotation is shown in polar coordinates, with the angular direction corresponding to the image rotation, and the radial to the image similarity score. This example is computed for VLAD with joint encoding of the dominant orientation, while using 3, 5 or 10 frequencies.

Figure 12: Matching example of two images and similarity estimation for all possible image translations. Image on the left undergoes translation independently to the horizontal and the vertical direction. VLAD jointly encoded with 2D coordinates of the local features is used for the example.

2.8 Adversarial attacks on deep global descriptors

Deep neural networks are known to be easily tricked by adversarial examples. For instance, in a classification setup, the pixel values of an image are perturbed in a way that affects and distorts the class prediction of a deep network classifier, but at the same time the image distortion is not perceivable by the human eye. The task is called targeted or non-targeted misclassification, depending on whether the resulting class has to be a specific one or whether any other class is enough.

We introduce and study a new setup that mainly focuses on privacy aspects but also reveals security issues of visual representation in the context of a retrieval system. The task is called targeted mismatch

adversarial attack [13]. The pixel values of a carrier image are perturbed, so that its representation will match that of another target image. An application scenario is that of a user that uses a retrieval system by uploading queries while not disclosing the true private content of the images; the true image query content is hidden in the carrier image, in particular in its representation.

We propose different ways to achieve that according to different assumptions for the known and unknown parts of the attacked retrieval system. Since there are approaches that are able to reconstruct an image based on internal representation of a CNN, we pay attention to attacks that do not allow such a reconstruction. Examples are shown in Figure 13.

Other than the presented privacy preserving retrieval setup, from a different point of view, malicious scenarios exist too, as in the following example. Someone publicly shares images with inappropriate content whose representation matches that of a very common object, which is often included in user queries. If a retrieval system indexes them, then, its top results might include these inappropriate images when queries of the common objects are used. Our work raises attention to this problem which is worth further investigation.

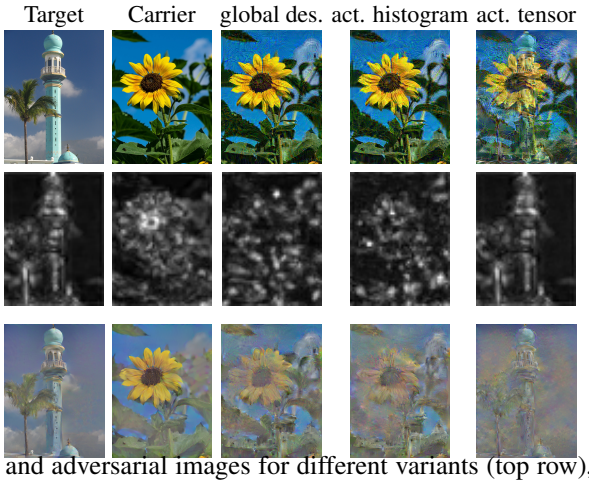


Figure 13: Target, carrier and adversarial images for different variants (top row), a summary of activation tensors by depth-wise maximum (middle row) and the inversion of feature maps obtained from images in the top row (bottom row). The inversion shows whether the target, or any information about it, can be reconstructed from the adversarial image. The first two inversions are presented as a reference.

Publications: [13] in Appendix XVI

3 Visual search

The goal of visual search is to estimate pairwise similarity between a query image and all images of a large image collection. Efficiency of the search is an essential property which is achieved by indexing structures or similarity/distance approximations. Effectiveness of the search is inherited by the properties of the representation and the similarity, while it is also affected by any approximation in the search process.

Global descriptors presented in Section 2 have not only the advantage of compactness but also compatibility with existing methods for efficient search. Standard similarity measures such as cosine similarity or dot product are efficiently implemented by vector to matrix multiplication, while GPU-based implementations offer a significant hardware-based speed-up. Additionally, they are compatible with exact or approximate nearest neighbor search methods in high dimensional spaces, where typical examples are (randomized) KD-trees [4, 83] and quantization-based methods such as product quantization [48, 51, 28].

On the other hand, ASMK is compatible with an inverted-file structure which is also the case for all codebook-based representations. An inverted file consists of a posting list per visual word. Each posting list consists of the image ids of images that have local descriptors assigned to this visual word. In contrast to simple BoW, ASMK stores the aggregated representation (binarized vector for efficiency) together with the image id. ASMK similarity in (24) requires to go only through visual words that appear in both images. With the inverted-file, during query time, one needs to visit only posting lists of visual words that appear in the query image to properly evaluate (24), which is evaluated for all database images at the same time.

This section is about search with query expansion (QE) that departs from the use of pairwise similarity. Similarity between the query and a database image depends not only on these two images, but also on the whole database or a subset of it. Such QE-based similarity considers the database of images as a whole.

3.1 Diffusion for query expansion

Diffusion denotes a mechanism spreading the query similarities over the manifolds composing the dataset. We mainly follow Zhou *et al*. [100] below. Given a database of n examples (examples are typically images), we define the *affinity matrix* $A = (a_{ij}) \in \mathbb{R}^{n \times n}$ having as elements the pairwise similarities between all example pairs obtained by a given global descriptor. In practice, only reciprocal nearest neighbors are let to have non-zero similarity, and all other elements are set to zero.

Matrix A is the adjacency matrix of a weighted undirected graph G with vertices being the database examples. The degree matrix of the graph is $D = \mathrm{diag}(A\mathbf{1}_n)$, *i.e*. a diagonal matrix with the row-wise sum of A. The Laplacian of the graph is defined as $L = D - A$. It is usual to symmetrically normalize these matrices, for instance,

$$S = D^{-1/2} A D^{-1/2}, \tag{38}$$

for the affinity matrix and $\mathcal{L} = I_n - S$ for the Laplacian, where I_n denotes the identity matrix of size n. Matrices $L, \mathcal{L}$ are positive-semidefinite [20].

In the work of Zhou *et al*. [100], a vector $\mathbf{y} = (y_i) \in \mathbb{R}^n$ specifies a set of query points, with $y_i = 1$ if the i-th example is a query and $y_i = 0$ otherwise. The objective is to obtain a ranking score f_i for each database example, represented as vector $\mathbf{f} = (f_i) \in \mathbb{R}^n$.

We focus on a particular diffusion mechanism that, given an initial vector $\mathbf{f}^0$, iterates according to

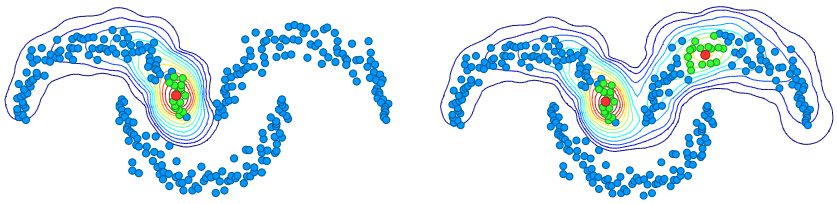
$$\mathbf{f}^t = \alpha S \mathbf{f}^{t-1} + (1 - \alpha)\mathbf{y}. \tag{39}$$

If S is a transition matrix and $\mathbf{y}$ an ℓ^1 unit vector, this defines the following 'random walk' on the graph: with probability α one jumps to an adjacent vertex according to distribution stated in S, and with $1 - \alpha$ to a query point uniformly at random. In this fashion, points spread their ranking score to their neighbors in the graph. The benefit is the ability to capture the intrinsic manifold structure represented by the affinity matrix and to combine multiple query points.

Assuming $0 < \alpha < 1$, Zhou *et al*. [99, 100] show that sequence $\{\mathbf{f}^t\}$ defined by (39) converges to

$$\mathbf{f}^\star = (1 - \alpha)\mathcal{L}_\alpha^{-1}\mathbf{y} \tag{40}$$

where $\mathcal{L}_\alpha = I_n - \alpha S$ is positive-definite. See Figure 14 for a toy illustration of running diffusion (39) on 2D data. Our contributions on diffusion-based search are described in the following.



(a) single query (b) multiple queries

Figure 14: Diffusion on a synthetic dataset in $\mathbb{R}^2$. Dataset points, query points and their k-nearest neighbors are shown in blue, red, and green respectively. Contour lines correspond to ranking scores after diffusion.

3.1.1 Efficient diffusion

In prior work on diffusion, a query point is considered to be contained in the dataset [99, 26]. This does not hold in a retrieval scenario, but a query can be included in the dataset graph at query time [97] by appropriately appending the affinity matrix which has some extra computational cost. Even if we ignore the time needed for this computation, we argue that locking, modifying and augmenting the entire affinity matrix for each query is not acceptable in terms of space requirements. Imagine the case of multiple users querying at the same time; a different matrix per query is required. We introduce an alternative method which defines vector $\mathbf{y}$ in a new way rather than modifying A. Qualitatively, instead of searching for $\mathbf{q}$, we are searching for its neighbors $\mathrm{NN}_k(\mathbf{q})$, appropriately weighted. Vector $\mathbf{y}$ is populated at the elements corresponding to the query neighbors with values equal to their similarity to the query. Note that these neighbors are estimated according to pairwise similarity with the original global descriptors.

We focus [39] on the *closed form* solution (40) rather than its intuitive derivation from iterative process (39). Iteration (39) works well in practice but is slow at large scale. Taking the closed-form solution (40) literally, one may be tempted to compute the inverse $\mathcal{L}_\alpha^{-1}$ offline, but this matrix is not sparse like $\mathcal{L}_\alpha$. We propose a more efficient solution by making the connection to linear system solvers.

Conjugate gradient (CG) [65] is the method of choice for solving linear systems like ours

$$\mathcal{L}_\alpha \mathbf{f} = (1-\alpha)\mathbf{y}, \tag{41}$$

where $\mathcal{L}_\alpha$ is positive-definite, and in particular for graph-related problems [92]. It has been used for random walk problems [32], but not diffusion-based retrieval according to our knowledge. In fact, the linear system formulation has been explicitly avoided in this context [26]. Here we argue, as in [54], that it is the solution of (41) that we seek, rather than the path followed by iteration (39). However, we use CG to approximate this solution, since matrix $\mathcal{L}_\alpha$ is indeed positive-definite.

Publications: [39] in Appendix XVII

3.1.2 Regional diffusion

The diffusion mechanism described so far is applicable to image retrieval when database and query images are globally represented by single vectors. We call this *global diffusion*. Unlike the traditional representation with local descriptors [89, 69], global diffusion fits perfectly with the early CNN-based global features [2, 52, 70].

Global features still fail under severe occlusion or when the object of interest is small. Local CNN features from multiple image regions have been investigated for this purpose, either aggregated [30, 14] or represented as a set [76]. Given a query image, the latter means that one searches for each query region/feature individually.

Fortunately, diffusion as defined in section (41) can already handle multiple queries. In the following, an image is represented by a set of m vectors, one for each region (same regions as the ones used in R-MAC [14]). The affinity matrix has an entry per region of each database image; its size is $nm \times nm$. The query image is also represented by a set of m vectors. In a similar case to global diffusion, we construct vector $\mathbf{y}$ to contain non-zero elements at the regions corresponding to the nearest neighbors of all the query

regions. After diffusion, we obtain similarity between all query regions and all database image regions. The combination of individual region ranking scores into a single score per image is obtained with Generalized Max Pooling (GMP) [64, 42]. We call this mechanism *regional diffusion* [39]. Regional representation used in regional diffusion significantly improves the retrieval performance, especially in the case of small objects as shown in Figure 15. Qualitative examples of the improvement are shown in Figure 16, where small objects are ranked noticeably higher with regional diffusion.

Publications: [39] in Appendix XVII

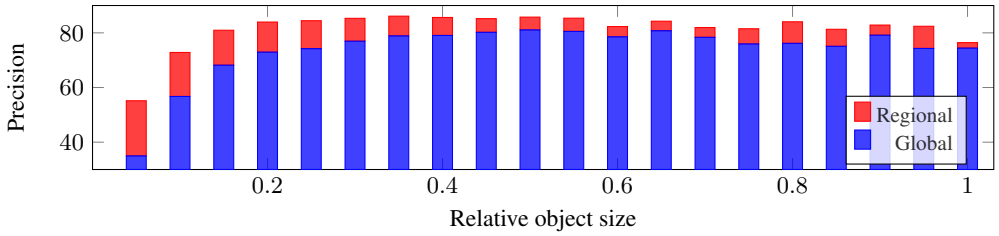


Figure 15: Precision computed at ranked positions of positive image, accumulated per relative object size. Statistics computed on INSTRE dataset [93] over all queries for global and regional diffusion.

Figure 16: Query examples from INSTRE, Oxford, and Paris datasets and retrieved images ranked by decreasing order of ranking difference between global and regional diffusion. We measure precision at the position where each image is retrieved and report this under each image for global and regional diffusion. Average Precision (AP) is reported per query for the two methods.

3.1.3 Diffusion with embeddings

The closed-form solution (40) is not convenient to use because, in contrast to $\mathcal{L}_\alpha$, its inverse is not sparse. In our work [34], we revisit this solution and provide an approximation with embeddings that are constructed in an off-line stage, so that diffusion is efficiently performed during query time.

Recall that $\mathcal{L}_\alpha = I_n - \alpha S$. We consider the low-rank spectral decomposition of sparse matrix S given by $S \approx U \Lambda U^\top$. The approximation is based on an existing randomized algorithm [77]. This allows us to obtain the low-rank decomposition of the non-sparse matrix $\mathcal{L}_\alpha$ and use it to approximate diffusion by

$$\mathbf{f}^\star = U(1-\alpha)(I_n - \alpha\Lambda)^{-1}U^\top \mathbf{y} = ULL^\top U^\top \mathbf{y}. \tag{42}$$

Columns of matrix UL can be seen as embeddings corresponding to the database examples, whose dimensionality controls the amount of approximation and affects the search performance. Estimating and storing these embeddings in advance, makes diffusion search dramatically faster.

Publications: [34] in Appendix XVIII

3.1.4 Hybrid diffusion

Diffusion offered by CG (41) is performed once for every new query represented by $\mathbf{y}$, but the affinity matrix represents the dataset and is fixed. Could CG be accelerated if we had some very limited additional information? On the other extreme, diffusion with embeddings (42) needs a large number of eigenvectors and eigenvalues to provide a high quality approximation, but always leaves some error. Could we reduce this space requirement by allocating some additional query time to recover the approximation error? The answer is positive to both questions and in fact these are the two extreme cases of *hybrid spectral-temporal filtering*, which is presented in our work [35]. Technical details are skipped in this manuscript. A comparison betweem this hybrid scheme, the spectral approach that uses the embeddings given by (42), and the temporal approach that solves (41) by CG is presented in Figure 17. We present all three trade-offs, namely performance vs time, memory vs time, and memory vs performance. Temporal ranking achieves high performance at the cost of high query time and its truncated counterpart (diffusion acts as re-ranking, *i.e.* sub-graph corresponding to top-ranked images is used) saves query time but sacrifices performance. Spectral ranking is not effective at this scale, while our hybrid solution achieves high performance at low query times.

Publications: [35] in Appendix XIX

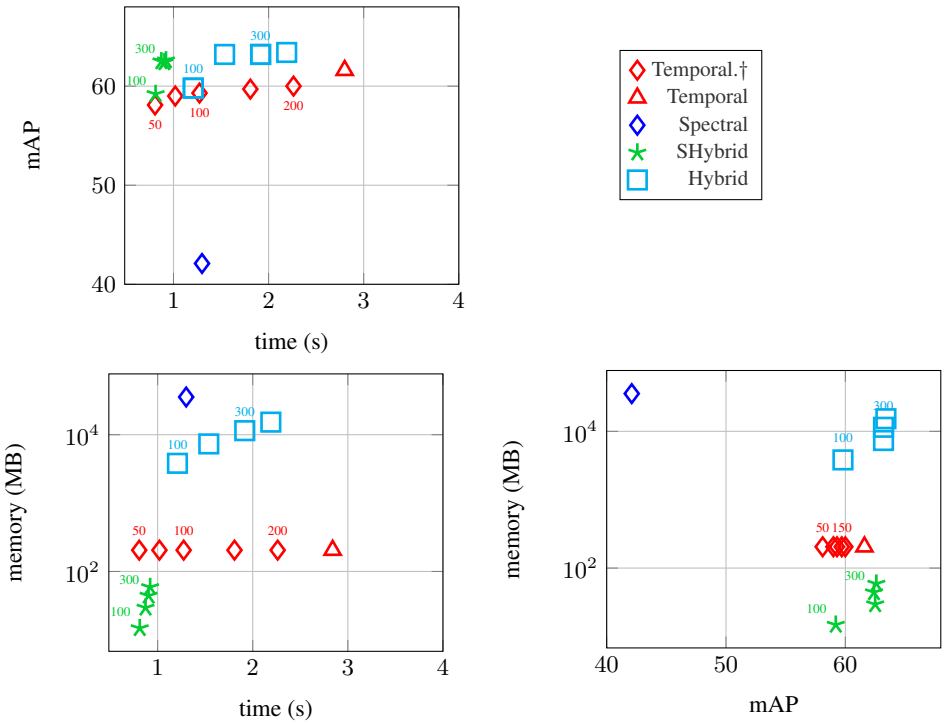


Figure 17: Time (s) - memory (MB) - performance (mAP) for different methods. We show mAP vs time, memory vs time, and memory vs mAP on ROxford+R1m. Methods in the comparison: temporal for 20 CG iterations, truncated temporal for 20 iterations and shortlist of top-ranked images of size 50k, 75k, 100k, 200k and 300k, spectral (FSRw) with embeddings of $5k$ dimensions, hybrid with embeddings of dimensions in $\{100, 200, 300, 400\}$ and 5 CG iterations, sparse hybrid with 99% sparsity (smallest embedding elements are set to 0), with embeddings of dimensions in $\{100, 200, 300, 400\}$ and 5 CG iterations. Text labels indicate the shortlist size (in thousands) for truncated temporal and rank (dimension) for hybrid. Observe that the two plots on the left are aligned horizontally with respect to time, while the two at the bottom vertically with respect to memory.

3.2 Query expansion with local descriptors

Query expansion techniques that rely on local descriptors typically use the BoW model where each descriptor is quantized and simply represented by the corresponding visual word id. This comes with performance benefits but a last step of geometric verification is required to increase the accuracy. In our work [11], for the first time, we propose a novel way to exploit the QE principle on top of an approach that individually matches the local descriptors and does not require geometric verification. The new approach is called Hamming Query Expansion (HQE). The approach that individually matches the local descriptors is Hamming Embedding (HE) [46, 47] (SMK is built on top of HE too). HE and SMK are very similar, but SMK did not exist at the time of HQE; the proposed approach is directly compatible with SMK and ASMK too. We revisit the different stages involved in the QE principle, namely how reliable images are selected and how an enriched query is generated. The modifications are done in a way to exploit the powerful representation of HE, *i.e*. the local binary vectors instead of simply using visual word ids. The technical details of this approach are skipped in this manuscript.
Publications: [11] in Appendix XX

3.3 *Alpha*-weighted query expansion with global descriptors

The most popular approach for QE with global descriptors, mainly due to its simplicity, is simple average query expansion (AQE) [19] widely used with CNN global descriptors [14, 52, 2, 31]. An initial query is issued and AQE acts on the top-ranked n images by average pooling of their descriptors. We argue [72] that tuning n to work well across different datasets is not easy. AQE corresponds to a weighted average where n descriptors have unit weight and all the rest zero. We generalize this scheme and propose performing weighted averaging, where the weight of the i-th ranked image is given by its similarity to the query raised to the power of α; the similarity of each retrieved image matters. We show in our experiments that AQE is difficult to tune for datasets of different statistics, while this is not the case with the proposed approach. We refer to this approach as α-weighted query expansion (αQE).The proposed αQE reduces to AQE for $\alpha = 0$.
Publications: [72] in Appendix XXIII

4 Object discovery

This section presents approaches for discovery of repeatings objects in large image collections. This is either done without any labels or with labels provided for a limited amount of images. In case of no labels, we present approaches that discover repeating objects from the point of view of instance-level matching, *i.e*. the same physical entity depicted in multiple images. The result is used either by directly integrating it to the represetation in a way that is beneficial for search or to pseudo-label the discovered images and use them for representation learning. In case of partial labeling, we explore applications for category-level classification and discover more images than the few labeled ones that depict the same object category, then pseudo-label them, and use them in the training stage for deep image classification.

4.1 Repeating-object discovery improves representation and search

4.1.1 Discovery and aggregation of repeating regions across multiple images

ASMK (see Section 2.1.3) aggregates local descriptors per image and manages to handle bursty matches. As we show in our experiments this process improves performance, compared to SMK, while at the same time memory requirements are reduced. We further propose an approach [6] to identify local features of different images that correspond to the same physical structure, *i.e*. the same object part. We aggregate their representation and index a single vector per group of similar features; an entry in the inverted-file corresponds to multiple images. In this way, we further compress the indexing structure, offer enhanced local representation derived from multiple similar images, and implicitly perform feature augmentation [91]. We adopt this method only with the binarized representation for scalability and efficiency reasons. Some details follow.

Given a set of images indexed with ASMK, we cross-match all indexed descriptors and find strong correspondences; *i.e*. local feature pairs with similarity above threshold τ_l. In practice, we cross match only features assigned to the same visual word. We further examine whether each match originates from images that are globally similar. That is, we discard a match if it refers to a pair of images that has less than τ_g strong correspondences. As a result, false positive matches are eliminated. We refer to thresholds τ_l and τ_g, as *local* and *global consistency thresholds*, respectively. They correspond to the descriptor level and image level matching.

The identified set of matches can be seen as an undirected graph, where nodes are indexed descriptors and edges are the matches. We then find the connected components of this graph, where each component corresponds to a set of similar local descriptors. We assign a single vector to each component, in analogy to the single vector per visual word of ASMK. This vector is constructed by aggregating all binary vectors of the component and binarizing once more. More formally, the vector assigned to component $\mathbf{G}$ is

$$\hat{b}_{\mathbf{G}} = \operatorname{sign}\left(\sum_{x \in \mathbf{G}} \hat{b}_x\right). \tag{43}$$

This process corresponds to majority voting per dimension.

During retrieval, when a query descriptor is compared against the descriptor of a component $\mathbf{G}$, then the score of all images associated with this component is increased. In Figure 18 we show examples of connected components found on images of Oxford5k. It appears that matching of binary signatures with the use of large similarity threshold provides a very fast way to identify true positive matches without any geometry in the loop.

The compressed vectors, used in ASMK, offer an efficient way to detect groups of similar features in large image collections. We thus show that descriptor aggregation can further be applied off-line across multiple images. In contrast to previous approaches of feature augmentation [1] or cross matching images for query expansion [22, 81, 24], this method does not increase either storage requirements or query time.

Publications: [6] in Appendix V

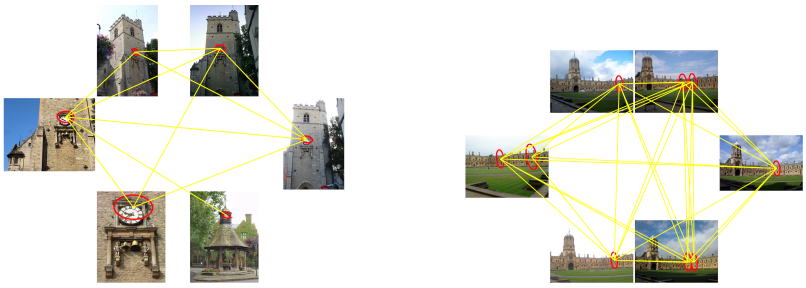


Figure 18: Examples of connected components discovered on Oxford5k.

4.1.2 Discovery of repeating objects using graph centrality

We propose an object discovery method [85, 84] that aims to discover regions corresponding to frequently repeating objects in a retrieval database and then pronounce them in the underlined representation. Therefore, the discovery process employs the existing representation, while its result is used to generate an enriched representation.

Like [91], our objective is to remove transient and non-distinctive objects, and rather focus on objects appearing frequently in a dataset. Beginning with the activation map of a convolutional layer in a CNN, one would need access to a local representation to automatically discover such objects. On the other hand, knowing what these objects are would help forming a local representation by selecting regions depicting them, which appears to be a chicken-and-egg problem. Without an initial region selection, we risk "discovering" uninformative but frequently appearing "stuff"-like patches, for instance sky.

Fortunately, it is possible to make an initial selection based on CNN activations alone, without any training and without bounding box annotations. The proposed mechanism is inspired by CroW [52] and Grad-CAM [80] and generates a *feature saliency* map. This initiates our offline analysis illustrated in Figure 19. A small set of rectangular regions is detected per image from this map. This first round of detection is applied independently per image and depends only on its content. Each region in the dataset is associated to a feature saliency score and a visual descriptor, pooled from the activation map of the corresponding image. It is now possible to compute a *centrality* score per region, representing the "significance" of each region in the dataset. This is based on a region k-NN graph. Now, given a new image, we can infer the "significance" of every region from its nearest neighbors in the graph, yielding a dense *object saliency* map. To achieve this, we suggest a non-parametric k-NN solution. Finally, we detect a small set of rectangular regions on this saliency map and extract a global descriptor to represent dataset images for retrieval. This second detection procedure takes into account all salient and repeating objects appearing in the whole dataset. The entire process is fully unsupervised and only assumes off-the-shelf networks trained on a classification or retrieval task without bounding box annotations.

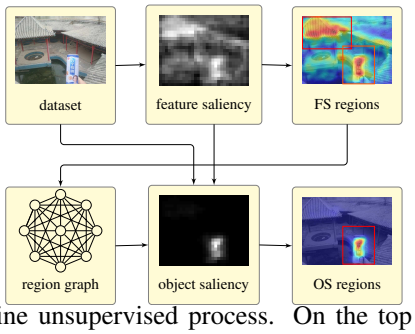


Figure 19: Overview of our offline unsupervised process. On the top row, CNN activations of dataset images are used to extract a *feature saliency* map, on which a set of regions is detected. On the bottom row, a *centrality* measure is obtained per region from a region k-NN graph. Using this measure, a dense *object saliency* map is formed from the original CNN activations and the feature saliency. This map is focusing on objects automatically discovered in the dataset, with background clutter removed. Finally, another set of regions is detected on the object saliency map to extract descriptors and represent the dataset for retrieval.

Publications: [84, 85] in Appendices XXI and XXII

4.2 Repeating-object discovery for train data mining

4.2.1 Bag-of-words and structure-from-motion as train data source

Classical methods for Structure-for-Motion (SfM) are a source of training data for deep models as supported in our work [72, 70]. We rely on a tightly-coupled BoW-based image-retrieval and SfM 3D reconstruction system [79] that starting from a crowd-sourced image collection generates 3D models of popular landmarks and urban scenery. We exploit the outcome of such a system. Images are clustered and a 3D model is constructed per cluster. For each image, the estimated camera position is known, as well as the local features registered on the 3D model. We drop redundant (overlapping) 3D models, while models reconstructing the same landmark but from different and disjoint viewpoints are considered as non-overlapping.

We mine positive and negative image pairs that are used with contrastive loss to learn an embedding network that maps images to their global descriptors such as MAC and GeM. Positive pairs are chosen within the same 3D model as long as they share enough common 3D points. The viewpoint changes have though to be contrained to be neither too large nor too small. The former cannot be captured by the CNN, and the latter are not so useful for training; this behaviour was experimentally verified. These mined examples are some sort of hard-positives, which are not common in other learning tasks. Hard-negatives are common though. Negative pairs are chosen among different 3D models, with hard negative mining being performed. Hard negative examples are the negative examples that are currently mapped in nearby points of the representation space. This mining process results in a training set for deep image retrieval that is widely used in a large number of follow-up methods. Examples of the discovered positive and negative image pairs are shown in Figures 20 and 21. Despite the success of this approach, such 3D models have include much richer geometric information which is yet not explored for deep learning.

After conversion of image into edge-maps and with the use of the SfM information for the original images, the mining results in examples like the ones showin in Figure 22, which are used for training the edgeMAC architecture presented in Section 2.5.2 that generated shape-based descriptors. Observe how the negative examples correspond to edge-maps of similar but not identical shapes.

Publications: [70, 72] in Appendices II and XXIII

Figure 20: Examples of positive (matching) image pairs mined from 3D models obtained by SfM.

Figure 21: Examples of hard-negative (non-matching) image pairs mined through 3D models obtained by SfM. Pairs form by the left-most image and another image in the same row.

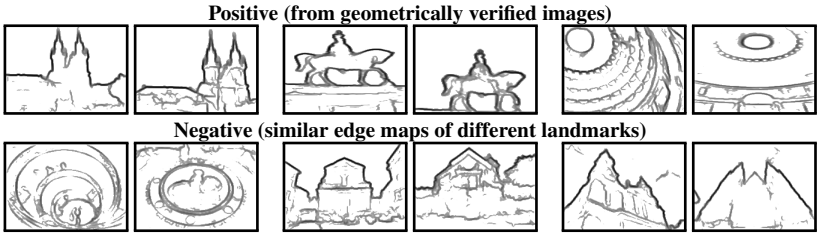


Figure 22: Positive and negative pairs based on SfM after conversion into edge-maps. Hard-negatives obtained based on edgeMac descriptors.

4.2.2 Mining on manifolds as train data source

Diffusion described in Section 3.1.1 takes into account the structure of the representation space as this is indicated by the whole image database, and the corresponding affinity matrix. We term the obtained similarity as manifold similarity, and the nearest neighbors obtained with it as manifold neighbors. This is as opposed to the Euclidean nearest neighbors obtained in a pairwise manner, *i.e*. via Euclidean distance or cosine similarity in the representation space.

We are guided by the manifold similarity to select training samples without any supervision [36]. In particular, we exploit the differences between Euclidean and manifold nearest neighbors of anchor items. See Figure 23 for examples of mined images.

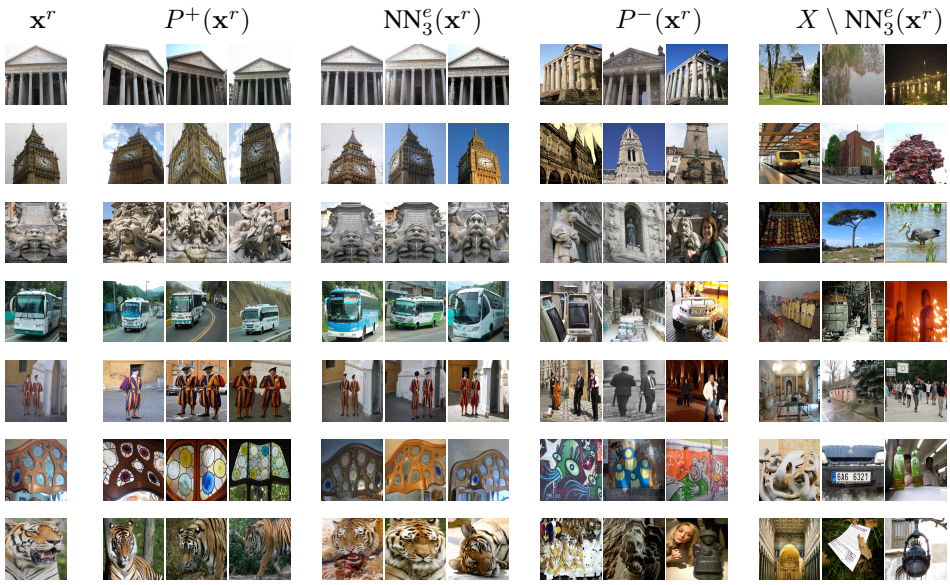


Figure 23: Sample anchor images ($\mathbf{x}^r$), positive images from our method ($P^+(\mathbf{x}^r)$) and baseline ($\mathrm{NN}_3^e(\mathbf{x}^r)$) that chooses the Euclidean neighbors, and negative images from our method ($P^-(\mathbf{x}^r)$) and baseline ($X \setminus \mathrm{NN}_3^e(\mathbf{x}^r)$) that chooses images that are not Euclidean neighbors. The baseline is Euclidean nearest neighbors and non-neighbors [33].

Anchors. We are interested in anchors that have many relevant images in the collection, which facilitates propagating on the manifold and discovering differences with the Euclidean neighborhood. We are also interested in anchors that are diverse, so that there is as little redundancy during training. Both conditions are satisfied by the modes of the nearest neighbor graph, which we compute as follows.

We first compute the stationary probability distribution of a random walk [17] on the nearest neighbor

graph of the whole dataset. This is achieved by the power iteration method [54] yielding the leading left eigenvector of the transition matrix. The probability reflects the *importance* of each node in the graph, as expressed by the probability of a random walker visiting it. We find the local maxima of the stationary distribution on the NN-graph and out of those, we keep a fixed number having the maximum probability. This is defined as the *anchor set*.

Positives. Given an anchor item $\mathbf{x}^r$, an initial feature extractor $g(\cdot)$ (for example a backbone network pre-trained on ImageNet), and the corresponding feature $\mathbf{y}^r = g(\mathbf{x}^r)$, which is used as query, we choose as positives the items that correspond to the manifold nearest neighbors of $\mathbf{y}^r$ that are not Euclidean neighbors. Such difference provides evidence of a matching item that is not retrieved well in the feature space, as illustrated in Figure 24(c). In the embedding space, positives should be attracted to the anchor so that Euclidean and manifold neighbors agree.

We therefore simply compare the sets $\text{NN}_k^m(\mathbf{y}^r)$ and $\text{NN}_k^e(\mathbf{y}^r)$ and select the input items that correspond to their set difference

$$P^+(\mathbf{x}^r) = \{\mathbf{x} \in X : g(\mathbf{x}) \in \text{NN}_k^m(\mathbf{y}^r) \setminus \text{NN}_k^e(\mathbf{y}^r)\} \tag{44}$$

as the *positive pool* of anchor $\mathbf{x}^r$. The value of k controls the visual diversity of positives, with larger values giving the *harder* examples. In practice, we maintain the pool ordered by descending manifold similarity, so that we may truncate by keeping the examples with highest confidence.

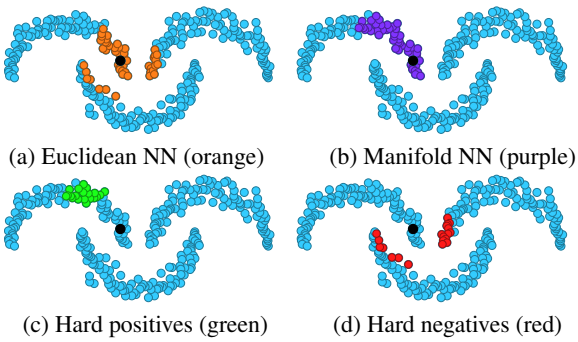
Mining hard positive examples, in contrast to negatives, is not common. Apart from positives being fewer than negatives, this is due to the large intraclass variability; it may result in positives that are too hard to learn. It can be achieved in cases with known geometry of the scene such that extreme cases are avoided [86, 70]. In our case, the hardness is controlled by the manifold similarity according to the current model, so drifting into very tough examples is less likely.

Negatives. Similarly, and as illustrated in Figure 24(d), we choose as negatives the items that correspond to the Euclidean nearest neighbors of $\mathbf{y}^r$ that are not manifold neighbors. Such difference provides evidence of a non-matching item that is too close in the feature space. In the embedding space, positives should be repelled from the anchor. The *negative pool* of anchor $\mathbf{x}^r$ is defined accordingly as

$$P^-(\mathbf{x}^r) = \{\mathbf{x} \in X : g(\mathbf{x}) \in \text{NN}_k^e(\mathbf{y}^r) \setminus \text{NN}_k^m(\mathbf{y}^r)\}. \tag{45}$$

It is common practice, and known to be beneficial [87], to select hard negative examples. By construction, its size is controlled by k. Again, we maintain the pool ordered by descending Euclidean similarity to keep the hardest negative examples.

Publications: [36] in Appendix XXIV



(a) Euclidean NN (orange) (b) Manifold NN (purple)

(c) Hard positives (green) (d) Hard negatives (red)

Figure 24: Given an anchor point (black) and its k nearest Euclidean (NN_k^e) and manifold (NN_k^m) neighbors in a dataset, we choose positive samples as $\text{NN}_k^m \setminus \text{NN}_k^e$, and negative as $\text{NN}_k^e \setminus \text{NN}_k^m$. Data is unlabeled and the selection is fully unsupervised, including anchors.

4.3 Label propagation improves deep classification

In this section, and in contrast to the rest of the manuscript, we consider classification tasks where the labels are defined at category-level, *e.g*. like in ImageNet [78].

4.3.1 Classical label propagation for deep semi-supervised learning

The graph-based diffusion in (41) is the unsupervised counterpart of graph-based label propagation for transductive semi-supervised learning for classification tasks [99]. Transductive classification with this approach is performed by populating vector $\mathbf{y}$ with ones at elements corresponding to labeled examples for a particular class. Then, the resulting values in $\mathbf{f}$ indicate the confidence per example to be classified in that class. The same process is performed for all classes and the class of maximum confidence is picked. This process is illustrated for a 2D toy example in Figure 25.

In our work [37] we propose to use the result of transductive learning, *i.e*. label propagation from labeled to unlabeled examples in the training set, and inject it into inductive learning of a deep network classifier, *i.e*. the classifier is applicable to unseen examples. This is simply performed by pseudo-labeling the training examples according to the label propagation process and appropriately weighing them with the certainty of the pseudo-labeling.

This process is iteratively performed in the following way. We assume a pre-trained backbone that allows to map an image to a global descriptor, and append a linear classifier. Then, we obtain global descriptors for all training images (labeled and unlabeled) and construct the affinity matrix. Pseudo-labeling is performed and its result is used to train the deep network classifier. We use the improved backbone and go back to the step of extracting global descriptors. We show that such pseudo-labeling performs better than pseudo-labeling based on the trained deep network classifier. The interpretation of this result is that our pseudo-labels are complementary to examples that the network can already recognize with high enough certainty.

Publications: [37] in Appendix XXV

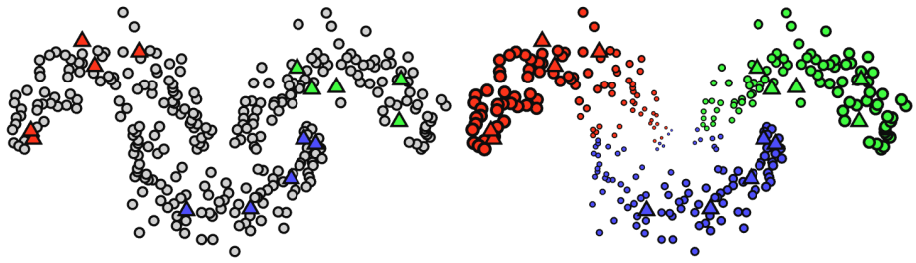


Figure 25: Label propagation on manifolds toy example. Triangles denote labeled, and circles un-labeled training data, respectively. Top: color-coded ground truth for labeled points, and gray color for unlabeled points. Bottom: color-coded pseudo-labels inferred by diffusion that are used to train the CNN. The size reflects the certainty of the pseudo-label prediction.

4.3.2 Graph convolutional networks for learning with few clean and many noisy labels

We consider the problem of learning an object classifier given few examples that are labeled in a clean way and many examples that are labeled in a weak and noisy way. For instance the latter are collected with text-based web crawling using the class names as queries. We assume that we are given a feature extractor mapping an example to a d-dimensional vector. For instance, when examples are images, the feature extractor is typically a CNN. The goal is to train a K-way classifier, using the additional noisy set in order to improve the accuracy compared to only using the small clean set. Our solution [38] uses graph convolutional networks [53] to perform the dataset cleaning, *i.e*. the estimation of a confidence value per noisy image declaring the probability of coming from the related class.

We consider a two-layer GCN with a scalar output per example. This network is applied per class and is a function $F_\Theta : \mathbb{R}^{N\times N} \times \mathbb{R}^{d\times N} \rightarrow \mathbb{R}^N$, where N is the total number of training images for the examined

class. Function F_Θ performs feature propagation through the affinity matrix in an analogy to classical graph-based propagation methods as diffusion.

The output $F_\Theta(\tilde{A}, V)$ is a vector of length N, with $\tilde{A}$ being a normalized version of the affinity matrix A (same as in diffusion), and V the matrix containing image features, *i.e.* global descriptors, as columns. Element $F_\Theta(\tilde{S}, V)_i \in [0, 1]$ represents a relevance value of example x_i for the examined class. To learn the parameters Θ, we treat the GCN as a *binary classifier* where target output 1 corresponds to clean examples and 0 to noisy. In particular, we minimize the loss function

$$L_{\mathcal{G}}(V, \tilde{A}; \Theta) = -\frac{1}{k} \sum_{i=1}^{k} \log \left(F_\Theta(\tilde{A}, V)_i \right) - \frac{\lambda}{N-k} \sum_{i=k+1}^{N} \log \left(1 - F_\Theta(\tilde{A}, V)_i \right). \tag{46}$$

The term on the left goes over the few clean examples and the term on the right goes over the noisy examples. This is a binary cross-entropy loss function where noisy examples are given an importance weight λ. Given the propagation on the nearest neighbor graph, and depending on the relative importance λ of the second term, noisy examples that are strongly connected to clean ones are still expected to receive high class relevance, while noisy examples that are not relevant to the current class are expected to get a class relevance near zero. The semi-supervised learning setup of GCNs [53] uses a loss function that applies only to the labeled examples, and makes discrete predictions on unlabeled examples. In our case, all examples contribute to the loss but with different importance, as we infer real-valued class relevance for the noisy examples, to be used for subsequent learning.

The impact of parameter λ is validated and we show that the fewer the available clean images are (smaller k) the smaller the importance weight should be. As is standard practice for GCNs in classification [53], training is performed in batches of size N, that is the entire set of features. Figure 26 shows examples of clean images, corresponding noisy ones and the predicted relevance. It turns out that using the visual similarity to the clean image, we can use relevance to resolve cases of polysemy, *e.g. black widow (spider) vs. black widow (superhero)*, or cases like *pineapple vs. pineapple juice*.

Our cleaning process applies when the clean labeled examples are few, but assumes a feature extractor. That is, representation learning, label cleaning and classifier learning are decoupled. We perform GCN-based cleaning, and learn a classifier simply by weighting examples according to class relevance. This approach improves classification accuracy up to 10% compared to assuming all weakly labeled images are clean.

Publications: [38] in Appendix XXVI

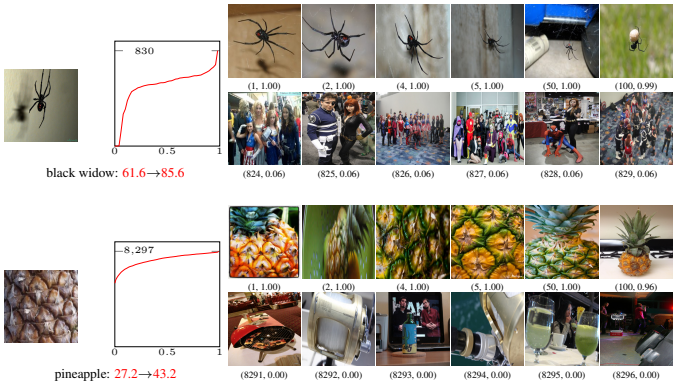


Figure 26: Examples of clean images from the Low-Shot ImageNet Benchmark (left) for 1-shot classification, cumulative histogram of the predicted relevance for noisy images (middle) and representative noisy images (right) by descending order of relevance, with relevance value reported below. Test accuracy without and with additional data is shown next to class names.

5 Datasets for instance-level recognition

Research benchmarks are essential tools to foster research progress, while training datasets are required to exploit deep models that depend on data hungry training. This section presents our contributions to that.

5.1 Instance-level search: revisiting Oxford and Paris

We address [73] issues with image retrieval benchmarking on standard and popular Oxford 5k and Paris 6k datasets. Our earlier work on query expansion [34] achieved very high performance and visual inspection of the top mistakes revealed that severe part of the performance drop is caused by annotation errors. We address such annotation errors, additionally the size of the dataset, and the level of challenge: new annotation for both datasets is created with an extra attention to the reliability of the ground truth. Three new protocols of varying difficulty are introduced. The protocols allow fair comparison between different methods, including those using a dataset pre-processing stage. For each dataset, 15 new challenging queries are introduced. Finally, a new set of 1M hard, semi-automatically cleaned distractors is selected.

The newly created annotation is more details including higher granularity of labels. The possible labels are *Easy*, *Hard*, *Unclear*, and *Negative*. In Figure 27, representative examples of easy, hard, and unclear images are presented for several queries. We construct a new distractor set with exactly 1,001,001 high-resolution (1024×768) images, which we refer to as $\mathcal{R}$1M dataset. It is cleaned by a semi-automatic process. We automatically pick hard images for a number of state-of-the-art methods, resulting in a challenging large scale setup. Example images from the set $\mathcal{R}$1M are shown in Figure 28.

An extensive comparison of the state-of-the-art methods is performed on the new benchmark. Different types of methods are evaluated, ranging from local-feature-based to modern CNN based methods. The best results are achieved by taking the best of the two worlds. Most importantly, the conducted experiments conclude that image retrieval is not yet solved.

Publications: [73] in Appendix XXVII

Figure 27: Sample **query (blue)** images and images that are respectively marked as **easy (dark green)**, **hard (light green)**, and **unclear (yellow)**. Best viewed in color.

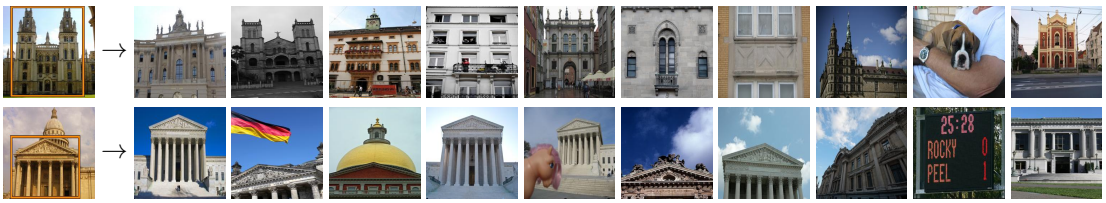


Figure 28: The most distracting images per query for two queries.

5.2 Instance-level classification: Met artworks

We introduce [96] a dataset for large-scale instance-level classification in the domain of artworks. The proposed benchmark exhibits a number of different challenges such as large inter-class similarity, long tail distribution, and many classes. We rely on the open access collection of The Met museum to form a large training set of about 224k classes, where each class corresponds to a museum exhibit with photos taken under studio conditions. Testing is primarily performed on photos taken by museum guests depicting exhibits, which introduces a distribution shift between training and testing. Testing is additionally performed on a set of images not related to Met exhibits making the task resemble an out-of-distribution detection problem. The proposed benchmark follows the paradigm of other recent datasets for instance-level classification on different domains to encourage research on domain independent approaches. A number of suitable approaches are evaluated to offer a testbed for future comparisons. Self-supervised and supervised contrastive learning are effectively combined to train the backbone which is used for non-parametric classification that is shown as a promising direction.

There are several factors that make instance-level classification a challenging task. It is typically required to deal with a large category set, whose size reaches the order of 10^6, with many classes represented by only a few or a single example, while the small between class variability further increases the hardness. Due to these difficulties the choice is often made to handle instance-level classification as an instance-level retrieval task [12]. Particular applications, *e.g.* in the product or art domain require dynamic updates of the category set; images from new categories are continuously added. Therefore, instance-level classification is a form of open set recognition [29]. Creating datasets with accurate ground truth at large scale for this task is a tedious process. As a consequence, many datasets include noise in their labels [23, 95].

The introduced dataset is accompanied by performance evaluation of relevant approaches. We show that non parametric classifiers perform much better than parametric ones. Improving the visual representation becomes essential with the use of non-parametric classifiers. To this end, we show that the recent self-supervised learning methods that rely only on image augmentations are beneficial, but the available labels should not be discarded. A combined self-supervised and supervised contrastive learning approach is the top performer in our benchmark indicating promising future directions. Challenging recognition examples are shown in Figure 29.

Publications: [96] in Appendix XXVIII

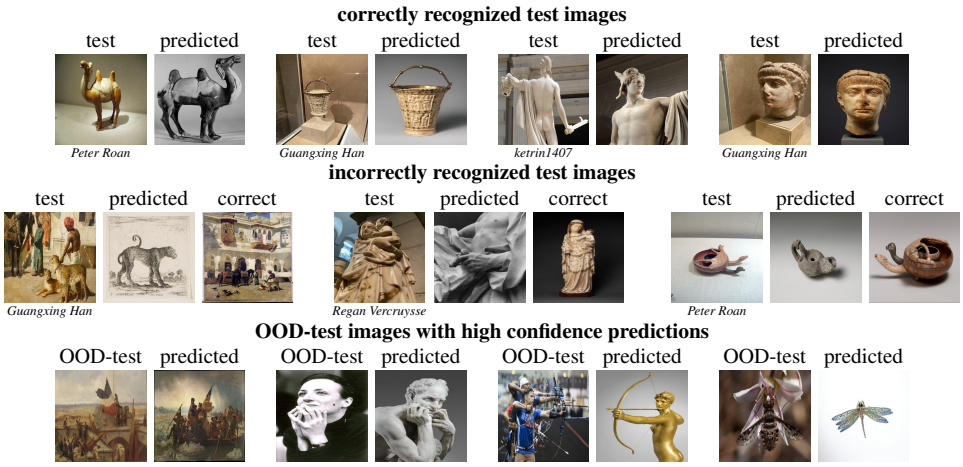


Figure 29: Challenging examples from the Met dataset for the top performing approach. Test images are shown next to their nearest neighbor from the Met exhibits that generated the prediction of the corresponding class. Top row: correct predictions. Middle row: incorrect predictions; an image of the ground truth class is also shown. Bottom row: high confidence predictions for OOD-test images; the goal is to obtain low confidence for these.

6 Summary

In the first part of the thesis, our contributions on visual representation and visual similarity estimation are discussed. We present different methods and a common framework that includes cases from different context where input examples are either image regions or whole images or images sets. We additionally form a bridge between classical approaches that do not include any learning or deep convolutional neural networks. The match kernel formulation is proven to be a useful tool for such a unified view. The link between classical and deep methods emphasizes the need to depart from conventional deep learning building blocks, as in work on deep kernel descriptors, but further exploration seems necessary.

Constructing a single vector representation that captures all the details of an image is arguably a challenging task. In our work, we rely on classical kernel descriptors that offer a flexible tool to design appearance-based, geometric-based, or joint representations. These kernel descriptors come with advantages and drawbacks. They easily allow us to inject the desired invariance properties into the design of the representation. Nevertheless, the underlined increase in dimensionality is a problem; combination with learning-based approaches seems to be worth exploring.

There is an inherent trade-off between memory requirements, or speed, and recognition performance. Single-vector representation per image is compact and allows for efficient search, but is typically less effective than a vector set representation, *i.e*. a set of local descriptors stored and matched separately. A comparison between single-vector and multi-vector representation is performed a few times, in different contexts, in our work. The superiority of the latter is evident in the case of instance-level recognition. Especially in the task of search, vector set representation is well explored both in our own work and in the literature. However, this is not the case for classification tasks that typically rely on a more conventional deep architecture; an input image is mapped to a 3D activation tensor, pooled over the spatial dimensions into a single vector fed to a classifier. In this way, matching is internally performed through single-vector representation. Exploring the multi-vector case for classification seems as a future research direction; our work made some first steps with the use of non parametric classifiers, while extending to parametric ones is yet to be done.

The second part of the thesis focused on search and in particular our contributions on query expansion. The proposed approaches significantly boost the performance of different search systems. Our work includes both online and offline approaches, where most of the work is done during query time or during a large-scale costly pre-processing stage, respectively. The latter corresponds to the graph-based methodology that we followed in the form of diffusion. Diffusion allows us to obtain similarity measures that depart from the classical pairwise estimation where similarity between two examples depends only on their representation; it now depends on the whole collection/database. Therefore, a different collection induces a different similarity measure, and a larger collection potentially induces an improved similarity measure. Large databases typically impose computational and performance challenges in the task of visual search. Nevertheless, diffusion, due to the underlined improved similarity measure obtained from large databases, paves the way for exploring different applications like object discovery.

The third part of the thesis discusses the task of object discovery. Exploring the representation space and capturing the underlined manifold structures allowed us to generate a source of training data that either comes completely for free, *i.e*. without any supervision, or in a semi-supervised way, *i.e*. with few labeled examples. From a different point of view, capturing such structures in the representation spaces induces new, data-dependent, metrics which can be used as a supervisory signal for self-supervised learning. This was successfully achieved in our own work on instance-level and on fine-grained category tasks. Offline experiments, not included in the published manuscripts, indicate failure for the more generic case of object category tasks, *e.g*. ImageNet, and the key ingredient is yet to be found. Label propagation is shown to be a key ingredient for reducing annotation cost. To pursue it we explore both classical and recent learning-based methodology, demonstrating simplicity of the former and higher performance of the latter. Despite our work only exploring the case where data examples are images and the collection is a set of images, there are lots of different setups where such propagation is meaningful. These range from cases where examples are pixels as part of an image, or images frames as part of a video. These are worth investigating and developing task-tailored graph-based propagation. This forms another potential future direction to reduce the required supervision in different computer vision tasks.

References

[1] R. Arandjelovic and A. Zisserman. Three things everyone should know to improve object retrieval. In *CVPR*, Jun. 2012.

[2] Artem Babenko and Victor Lempitsky. Aggregating deep convolutional features for image retrieval. In *ICCV*, 2015.

[3] Axel Barroso-Laguna, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. Key. net: Keypoint detection by handcrafted and learned cnn filters. In *ICCV*, 2019.

[4] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

[5] **Giorgos Tolias**, Yannis Avrithis, and Hervé Jégou. To aggregate or not to aggregate: selective match kernels for image search. In *ICCV*, Sep. 2013.

[6] **Giorgos Tolias**, Yannis Avrithis, and Hervé Jégou. Image search with selective match kernels: aggregation across single and multiple images. *IJCV*, 2016.

[7] **Giorgos Tolias**, Andrei Bursuc, Teddy Furon, and Hervé Jégou. Rotation and translation covariant match kernels for image retrieval. *CVIU*, 140:9–20, 2015.

[8] **Giorgos Tolias** and Ondřej Chum. Asymmetric feature maps with application to sketch based retrieval. In *CVPR*, 2017.

[9] **Giorgos Tolias** and Ondřej Chum. Efficient contour match kernel. *IVC*, 2018.

[10] **Giorgos Tolias**, Teddy Furon, and Hervé Jégou. Orientation covariant aggregation of local descriptors with embeddings. In *ECCV*, 2014.

[11] **Giorgos Tolias** and Hervé Jégou. Visual query expansion with or without geometry: refining local descriptors by feature aggregation. *Pattern Recognition*, Apr. 2014.

[12] **Giorgos Tolias**, Tomas Jenicek, and Ondřej Chum. Learning and aggregating deep local descriptors for instance-level recognition. In *ECCV*, 2020.

[13] **Giorgos Tolias**, Filip Radenovic, and Ondřej Chum. Targeted mismatch adversarial attack: Query with a flower to retrieve the tower. In *ICCV*, 2019.

[14] **Giorgos Tolias**, Ronan Sicre, and Hervé Jégou. Particular object retrieval with integral max-pooling of CNN activations. In *ICLR*, 2016.

[15] Liefeng Bo and Cristian Sminchisescu. Efficient match kernel between sets of features for visual recognition. In *NIPS*, 2009.

[16] Andrei Bursuc, **Giorgos Tolias**, and Hervé Jégou. Kernel local descriptors with implicit rotation matching. In *ICMR*, 2015.

[17] Minsu Cho and Kyoung Mu Lee. Mode-seeking on graphs via random walks. In *CVPR*, 2012.

[18] O. Chum and J. Matas. Geometric hashing with local affine frames. In *CVPR*, 2006.

[19] O. Chum, J. Philbin, J. Sivic, M. Isard, and A. Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *ICCV*, 2007.

[20] Fan RK Chung. *Spectral graph theory*, volume 92. American Mathematical Soc., 1997.

[21] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *ECCV Workshop Statistical Learning in Computer Vision*, 2004.

[22] Q. Danfeng, S. Gammeter, L. Bossard, T. Quack, and L. Van Gool. Hello neighbor: Accurate object retrieval with k-reciprocal nearest neighbors. In *CVPR*, 2011.

[23] Riccardo Del Chiaro, Andrew D Bagdanov, and Alberto Del Bimbo. Noisyart: A dataset for webly-supervised artwork recognition. In *VISIGRAPP*, 2019.

[24] Agni Delvinioti, Hervé Jégou, Laurent Amsaleg, and Michael E Houle. Image retrieval with reciprocal and shared nearest neighbors. In *VISAPP*, 2014.

[25] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *CVPRW*, 2018.

[26] Michael Donoser and Horst Bischof. Diffusion processes for retrieval revisited. In *CVPR*, 2013.

[27] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-net: A trainable cnn for joint description and detection of local features. In *CVPR*, 2019.

[28] Tiezheng Ge, Kaiming He, Qifa Ke, and Jian Sun. Optimized product quantization for approximate nearest neighbor search. In *CVPR*, Jun. 2013.

[29] Chuanxing Geng, Sheng-jun Huang, and Songcan Chen. Recent advances in open set recognition: A survey. *PAMI*, 2020.

[30] Yunchao Gong, Liwei Wang, Ruiqi Guo, and Svetlana Lazebnik. Multi-scale orderless pooling of deep convolutional activation features. In *ECCV*, 2014.

[31] Albert Gordo, Jon Almazan, Jerome Revaud, and Diane Larlus. Deep image retrieval: Learning global representations for image search. In *ECCV*, 2016.

[32] Leo Grady. Random walks for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(11):1768–1783, 2006.

[33] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *CVPR*, 2006.

[34] Ahmet Iscen, Yannis Avrithis, **Giorgos Tolias**, Teddy Furon, and Ondrej Chum. Fast spectral ranking for similarity search. In *CVPR*, 2017.

[35] Ahmet Iscen, Yannis Avrithis, **Giorgos Tolias**, Teddy Furon, and Ondřej Chum. Hybrid diffusion: Spectral-temporal graph filtering for manifold ranking. In *ACCV*, 2018.

[36] Ahmet Iscen, **Giorgos Tolias**, Yannis Avrithis, and Ondřej Chum. Mining on manifolds: Metric learning without labels. In *CVPR*, 2018.

[37] Ahmet Iscen, **Giorgos Tolias**, Yannis Avrithis, and Ondřej Chum. Label propagation for deep semi-supervised learning. In *CVPR*, 2019.

[38] Ahmet Iscen, **Giorgos Tolias**, Yannis Avrithis, Ondřej Chum, and Cordelia Schmid. Graph convolutional networks for learning with few clean and many noisy labels. In *ECCV*, 2020.

[39] Ahmet Iscen, **Giorgos Tolias**, Yannis Avrithis, Teddy Furon, and Ondřej Chum. Efficient diffusion on region manifolds: Recovering small objects with compact CNN representations. In *CVPR*, 2017.

[40] Ahmet Iscen, **Giorgos Tolias**, Yannis Avrithis, Teddy Furon, and Ondřej Chum. Panorama to panorama matching for location recognitionmatching. In *ICMR*, 2017.

[41] Ahmet Iscen, **Giorgos Tolias**, Philippe-Henri Gosselin, and Hervé Jégou. A comparison of dense region detectors for image search and fine-grained classification. IEEE *Transactions on Image Processing*, 24(8):2369–2381, 2015.

[42] Ahmet Iscen, Teddy Furon, Vincent Gripon, Michael Rabbat, and Hervé Jégou. Memory vectors for similarity search in high-dimensional spaces. In *arXiv*, 2014.

[43] H. Jégou, M. Douze, and C. Schmid. On the burstiness of visual elements. In *CVPR*, Jun. 2009.

[44] H. Jegou, M. Douze, C. Schmid, and P. Perez. Aggregating local descriptors into a compact image representation. In *CVPR*, 2010.

[45] Hervé Jégou and Ondřej Chum. Negative evidences and co-occurences in image retrieval: The benefit of pca and whitening. In *ECCV*. Springer, 2012.

[46] Herve Jégou, Matthijs Douze, and Cordelia Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *ECCV*, Oct. 2008.

[47] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Improving bag-of-features for large scale image search. *IJCV*, 87(3):316–336, Feb. 2010.

[48] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *PAMI*, 33(1):117–128, Jan. 2011.

[49] Hervé Jégou, Florent Perronnin, Matthijs Douze, Jorge Sánchez, Patrick Pérez, and Cordelia Schmid. Aggregating local descriptors into compact codes. In *PAMI*, Sep. 2012.

[50] Hervé Jégou and Andrew Zisserman. Triangulation embedding and democratic aggregation for image search. In *CVPR*, 2014.

[51] Y. Kalantidis and Y. Avrithis. Locally optimized product quantization for approximate nearest neighbor search. In *CVPR*, Columbus, Ohio, June 2014.

[52] Yannis Kalantidis, Clayton Mellina, and Simon Osindero. Cross-dimensional weighting for aggregated deep convolutional features. In *ECCVW*, 2016.

[53] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

[54] Amy N Langville and Carl D Meyer. Deeper inside pagerank. *Internet Mathematics*, 1(3):335–380, 2004.

[55] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006.

[56] David G Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, Nov. 2004.

[57] J. Matas, O. Chum, U. Martin, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *BMVC*, pages 384–393, Sep. 2002.

[58] K. Mikolajczyk and C. Schmid. scale and affine invariant interest point detectors. *IJCV*, 60(1):63–86, 2004.

[59] Anastasya Mishchuk, Dmytro Mishkin, Filip Radenovic, and Jiri Matas. Working hard to know your neighbor's margins: Local descriptor learning loss. In *NeurIPS*, 2017.

[60] Dmytro Mishkin, Filip Radenovic, and Jiri Matas. Repeatability is not enough: Learning affine regions via discriminability. In *ECCV*, 2018.

[61] Arun Mukundan, **Giorgos Tolias**, Andrei Bursuc, Herve Jegou, and Ondrej Chum. Understanding and improving kernel local descriptors. *IJCV*, 2019.

[62] Arun Mukundan, **Giorgos Tolias**, and Ondrej Chum. Multiple-kernel local-patch descriptor. In *BMVC*, 2017.

[63] Arun Mukundan, **Giorgos Tolias**, and Ondřej Chum. Explicit spatial encoding for deep local descriptors. In *CVPR*, 2019.

[64] Naila Murray and Florent Perronnin. Generalized max-pooling. In *CVPR*, Jun. 2014.

[65] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer, 2006.

[66] Hyeonwoo Noh, Andre Araujo, Jack Sim, Tobias Weyand, and Bohyung Han. Large-scale image retrieval with attentive deep local features. In *ICCV*, 2017.

[67] F. Perronnin, Y. Liu, J. Sanchez, and H. Poirier. Large-scale image retrieval with compressed Fisher vectors. In *CVPR*, 2010.

[68] Florent Perronnin and Christopher R. Dance. Fisher kernels on visual vocabularies for image categorization. In *CVPR*, Jun. 2007.

[69] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, 2007.

[70] Filip Radenović, **Giorgos Tolias**, and Ondřej Chum. CNN image retrieval learns from BoW: Unsupervised fine-tuning with hard examples. In *ECCV*, 2016.

[71] Filip Radenovic, **Giorgos Tolias**, and Ondrej Chum. Deep shape matching. In *ECCV*, 2018.

[72] Filip Radenović, **Giorgos Tolias**, and Ondřej Chum. Fine-tuning cnn image retrieval with no human annotation. *PAMI*, 2019.

[73] Filip Radenović, Ahmet Iscen, **Giorgos Tolias**, Yannis Avrithis, and Ondřej Chum. Revisiting oxford and paris: Large-scale image retrieval benchmarking. In *CVPR*, 2018.

[74] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *NIPS*, 2007.

[75] C. Radhakrishna Rao and Sujit Kumar Mitra. Generalized inverse of a matrix and its applications. In *Proceedings of the Sixth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Theory of Statistics*, 1972.

[76] Ali S Razavian, Josephine Sullivan, Stefan Carlsson, and Atsuto Maki. Visual instance retrieval with deep convolutional networks. *ITE Trans. on Media Technology and Applications*, 2016.

[77] Vladimir Rokhlin, Arthur Szlam, and Mark Tygert. A randomized algorithm for principal component analysis. *SIAM Journal on Matrix Analysis and Applications*, 31(3):1100–1124, 2009.

[78] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 2015.

[79] Johannes Lutz Schönberger, Filip Radenović, Ondrej Chum, and Jan-Michael Frahm. From single image query to detailed 3D reconstruction. In *CVPR*, 2015.

[80] Ramprasaath R. Selvaraju, Abhishek Das, Ramakrishna Vedantam, Michael Cogswell, Devi Parikh, and Dhruv Batra. Grad-CAM: Why did you say that? visual explanations from deep networks via gradient-based localization. In *arXiv*, 2016.

[81] X. Shen, Z. Lin, J. Brandt, S. Avidan, and Y. Wu. Object retrieval and localization with spatially-constrained similarity measure and k-nn re-ranking. In *CVPR*, Jun. 2012.

[82] Ronan Sicre and Frédéric Jurie. Discriminative part model for visual recognition. *CVIU*, 141:28 – 37, 2015.

[83] C. Silpa-Anan and Richard Hartley. Optimised kd-trees for fast image descriptor matching. In *CVPR*, 2008.

[84] Oriane Simeoni, Ahmet Iscen, **Giorgos Tolias**, Yannis Avrithis, and Ondrej Chum. Unsupervised object discovery for instance recognition. In *WACV*, 2018.

[85] Oriane Simeoni, Ahmet Iscen, **Giorgos Tolias**, Yannis Avrithis, and Ondrej Chum. Graph-based particular object discovery. *MVA*, 2019.

[86] Edgar Simo-Serra, Eduard Trulls, Luis Ferraz, Iasonas Kokkinos, Pascal Fua, and Francesc Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *ICCV*, 2015.

[87] Edgar Simo-Serra, Eduard Trulls, Luis Ferraz, Iasonas Kokkinos, and Francesc Moreno-Noguer. Fracking deep convolutional image descriptors. In *arXiv*, 2014.

[88] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *ICCV*, Oct. 2003.

[89] Josef Sivic and Andrew Zisserman. Video Google: A text retrieval approach to object matching in videos. In *ICCV*, 2003.

[90] Akihiko Torii, Josef Sivic, Tomas Pajdla, and Masatoshi Okutomi. Visual place recognition with repetitive structures. In *CVPR*, 2013.

[91] Panu Turcot and David G Lowe. Better matching with fewer features: The selection of useful features in large database recognition problems. In *CVPR*, 2009.

[92] Nisheeth K Vishnoi. Laplacian solvers and their algorithmic applications. *Theoretical Computer Science*, 8(1-2):1–141, 2012.

[93] Shuang Wang and Shuqiang Jiang. Instre: a new benchmark for instance-level object retrieval and recognition. *ACM TOMM*, 11:37, 2015.

[94] Philippe Weinzaepfel, Thomas Lucas, Diane Larlus, and Yannis Kalantidis. Learning super-features for image retrieval. In *ICLR*, 2022.

[95] Tobias Weyand, André Araujo, Bingyi Cao, and Jack Sim. Google landmarks dataset v2 - A large-scale benchmark for instance-level recognition and retrieval. In *CVPR*, 2020.

[96] Nikolaos-Antonios Ypsilantis, Noa Garcia, Guangxing Han, Sarah Ibrahimi, Nanne Van Noord, and **Giorgos Tolias**. The met dataset - instance-level recognition for artworks. In *NeurIPS*, 2021.

[97] Shaoting Zhang, Ming Yang, Timothee Cour, Kai Yu, and Dimitris N Metaxas. Query specific fusion for image retrieval. In *ECCV*, 2012.

[98] Wanlei Zhao, Hervé Jégou, and Guillaume Gravier. Oriented pooling for dense and non-dense rotation-invariant features. In *BMVC*, Sep. 2013.

[99] Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. In *NIPS*, 2003.

[100] Dengyong Zhou, Jason Weston, Arthur Gretton, Olivier Bousquet, and Bernhard Schölkopf. Ranking on data manifolds. In *NIPS*, 2003.

I Particular object retrieval with integral max-pooling of CNN activations

Title: Particular object retrieval with integral max-pooling of CNN activations
Authors: G. Tolias, R. Sicre, H. Jégou
Published at: ICLR 2016

PARTICULAR OBJECT RETRIEVAL WITH INTEGRAL MAX-POOLING OF CNN ACTIVATIONS

Giorgos Tolias *
Center for Machine Perception
FEE CTU Prague

Ronan Sicre
Irisa Rennes

Hervé Jégou
Facebook AI Research

ABSTRACT

Recently, image representation built upon Convolutional Neural Network (CNN) has been shown to provide effective descriptors for image search, outperforming pre-CNN features as short-vector representations. Yet such models are not compatible with geometry-aware re-ranking methods and still outperformed, on some particular object retrieval benchmarks, by traditional image search systems relying on precise descriptor matching, geometric re-ranking, or query expansion. This work revisits both retrieval stages, namely initial search and re-ranking, by employing the same primitive information derived from the CNN. We build compact feature vectors that encode several image regions without the need to feed multiple inputs to the network. Furthermore, we extend integral images to handle max-pooling on convolutional layer activations, allowing us to efficiently localize matching objects. The resulting bounding box is finally used for image re-ranking. As a result, this paper significantly improves existing CNN-based recognition pipeline: We report for the first time results competing with traditional methods on the challenging Oxford5k and Paris6k datasets.

1 INTRODUCTION

CONTENT based image retrieval has received a sustained attention over the last decade, leading to mature systems for tasks like visual instance retrieval. Current state-of-the-art approaches are derived from the Bag-of-Words model of Sivic & Zisserman (2003) and mainly owe their success to locally invariant features (Lowe, 2004) and large visual codebooks (Philbin et al., 2007). These methods are typically composed of an initial *filtering* stage where all database images are ranked in terms of similarity to a query image and a second *re-ranking* stage, which refines the search results of the top-ranked elements. The filtering stage is improved in several ways, such as incorporating weak geometric information (Jégou et al., 2010), employing compact approximations of the local descriptors (Jégou et al., 2010), or learning smart codebooks (Mikulik et al., 2013; Avrithis & Kalantidis, 2012). In such cases, local descriptors are individually matched and selective matching functions (Tolias et al., 2015; Tao et al., 2014) improve the search quality. Geometric matching models (Philbin et al., 2007; Avrithis & Tolias, 2014) are typically applied in a pairwise manner during the re-ranking stage of a short-list of images. Query expansion approaches significantly increase the performance (Chum et al., 2011), at the cost of larger query times.

The recent advances achieved by Convolutional Neural Networks (CNN) and the use of intermediate layer activations as feature vectors (Donahue et al., 2013) create opportunities for representations that are competitive for image or particular object retrieval, and not only classification tasks. Several works have already investigated this research direction, such as global or local representations based on either fully connected (Babenko et al., 2014; Gong et al., 2014) or convolutional layers (Razavian et al., 2014b; Azizpour et al., 2014; Babenko & Lempitsky, 2015). The performance of CNN-based features has rapidly improved to the point of competing and even outperforming pre-CNN works that aggregate local features (Jégou et al., 2012; Radenović et al., 2015). In particular, activations of convolutional layers followed by a global max-pooling operation (Azizpour et al., 2014) produce highly competitive compact image representations. One limitation is that such approaches are not compatible with the geometric-aware models involved in the final re-ranking stages.

*Research partially conducted while G. Tolias and H. Jégou were at Inria. We would like to thank Florent Perronnin for his valuable feedback. This work was partly supported by MSMT LL1303 ERC-CZ grant.

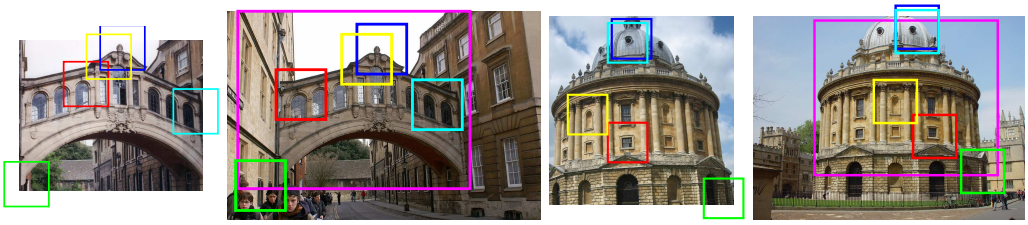


Figure 1: Query objects (left) and the corresponding localization in another image (right) are shown. We visualize the patches that contribute the highest to the image similarity score. Displayed patches correspond to the receptive field of CNN activations. Object localization is displayed in magenta, while different colors are used for patches in correspondence.

This work revisits both filtering and re-ranking stages with CNN-based features. We make the three following contributions.

- First, we propose a compact image representation derived from the convolutional layer activations that encodes multiple image regions without the need to re-feed multiple inputs to the network, in spirit of recent Fast-RCNN (Girshick, 2015) and Faster-RCNN (Ren et al., 2015) methods but here targeting particular object retrieval. The underlying primitive representation is used in all stages (initial retrieval and re-ranking).
- Second, we employ the generalized mean (Dollár et al., 2009) to enable the use of integral images along with max-pooling. This efficient method is exploited for particular object localization (see Figure 1) directly in the 2D maps of CNN activations.
- Third, our localization approach is used for image re-ranking and leads us to define a simple yet effective query expansion method.

These approaches are complementary and, when combined, produce for the first time a system which compete on the Oxford and Paris building benchmarks with state-of-the-art re-ranking approaches based on local features. Our approach outperforms by a large margin previous methods based on CNN, while being more efficient in practice.

2 Related work

CNN based representation. A typical CNN consists of several convolutional layers, followed by fully connected layers and ends with a softmax layer producing a distribution over the training classes. Instead of using this inherent classifier, one can consider the activations of the intermediate layers to train a classifier. In particular, the activations of the fully connected layers have been shown to be very effective and capable of adaptation to various domains (Oquab et al., 2014), such as scene recognition (Donahue et al., 2013; Sicre & Jurie, 2015), object detection (Iandola et al., 2014), and semantic segmentation (Girshick et al., 2014). In the case of image retrieval, fully connected layers are used as global descriptors followed by dimensionality reduction (Babenko et al., 2014). They are also employed as region descriptors to be compared to database descriptors (Razavian et al., 2014a) or aggregated in a VLAD manner (Gong et al., 2014).

Recent works derive visual representations from the activations of the convolutional layers. This is achieved either by stacking activations (Girshick et al., 2014) or by performing spatial max-pooling (Azizpour et al., 2014) or sum-pooling (Babenko & Lempitsky, 2015) for each feature channel. According to Azizpour et al. (2014) such representation offers better generalization properties for test data that are far from the source (training) data. Noticeably, higher performance in particular object or scene retrieval is obtained by using convolutional layers rather than fully connected ones. The very recent work of Babenko & Lempitsky (2015) shows that sum-pooling performs better than max-pooling when the image representation is whitened. In addition to be a costly choice, we will show that this is not optimal in our context of object localization (see Section 8). Finally, Kalantidis et al. (2015) propose spatial and feature channel weighting that significantly improves performance. Their approach is complementary to what we propose for the filtering and the re-ranking stage.

Recent examples utilize information from fully connected layers to perform generic object detection (Iandola et al., 2014; Papandreou et al., 2014). Such approaches are prohibitive for the re-ranking purposes of large scale image retrieval. They have high computational cost and the inherent features are not optimal for particular object matching.

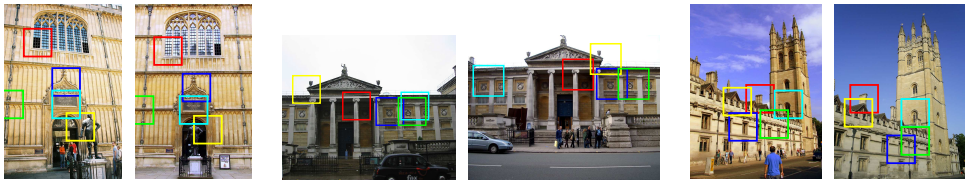


Figure 2: We visualize the receptive fields related to the 5 MAC components that contribute the most to the image similarity. Each displayed receptive field corresponds to the maximum response of a feature channel. A different color is used for each feature channel, while different feature channels are shown for each image pair.

Localization. In the recent years, the sliding window principle has been quite successful for many object localization methods. Due to the large number of possible windows, exhaustive search is extremely costly. However, integral images (Viola & Jones, 2001) offer a constant cost solution to the evaluation of a single region. This attractive alternative is applicable for feature vectors constructed via a sum-pooling operation.

A globally optimal solution is given by Efficient Subwindow Search (ESS) of Lampert et al. (2009), who use branch-and-bound search to avoid exhaustive search. Their work employs integral images, which are also used in later improvements of ESS (An et al., 2009). An et al. (2009) formalize localization as a maximum sub-array problem and similarly to Chen et al. (2013) they employ Bentley's algorithm (Bentley, 1999). Integral images facilitate the evaluation of many region candidates (Uijlings et al., 2013) based on VLAD or Fisher vectors (Van de Sande et al., 2014). All aforementioned approaches take advantage of integral images due to the inherent sum-pooling operation in the given representation. In this paper, we extend integral images to perform max-pooling over CNN activation maps, which is shown to be a better choice for describing regions (as opposed to the entire image).

Several object localization techniques have been proposed in the context of image retrieval as well. Lampert (2009) propose a two layer branch-and-bound method that alternates between regions and images. Integral images offer a significant speed-up in the work of Lin & Brandt (2010) to perform localization through Bag-of-Words. The overall idea bears similarities with our work. However, we differentiate by employing CNN-based representation with max-pooling. Some approaches (Tao et al., 2014; Shen et al., 2014) individually index local features for localization. In our case, the localization method is built on top of a compact representation, initially used for the filtering stage. Finally, Arandjelovic & Zisserman (2013) propose a localization strategy based on VLAD, where similarity is computed for multiple image regions, giving a more precise localization via regression.

3 Background

We consider a pre-trained CNN and discard all the fully connected layers. Given an input image I of size $W_I \times H_I$, the activations (responses) of a convolutional layer form a 3D tensor of $W \times H \times K$ dimensions, where K is the number of output feature channels, i.e. multi-dimensional filters. The spatial resolution $W \times H$ depends on the network architecture, the layer examined, and the input image resolution. We assume that Rectified Linear Units (ReLU) are applied as a last step, guaranteeing that all elements are non-negative.

We represent this 3D tensor of responses as a set of 2D feature channel responses $\mathcal{X} = \{\mathcal{X}_i\}, i = 1 \dots K$, where $\mathcal{X}_i$ is the 2D tensor representing the responses of the i^{th} feature channel over the set Ω of valid spatial locations, and $\mathcal{X}_i(p)$ is the response at a particular position p. Therefore, the feature vector constructed by a spatial max-pooling over all locations (Azizpour et al., 2014) is given by

$$\mathbf{f}_\Omega = [\mathrm{f}_{\Omega,1} \dots \mathrm{f}_{\Omega,i} \dots \mathrm{f}_{\Omega,K}]^\top, \text{ with } \mathrm{f}_{\Omega,i} = \max_{p \in \Omega} \mathcal{X}_i(p). \quad (1)$$

Maximum activations of convolutions (MAC). Two images are compared with the cosine similarity of the K-dimensional vectors produced as described above. This representation, referred to as MAC, does not encode the location of the activations (unlike activations of fully connected layers),

due to the max-pooling operated over a single region of size $W \times H$. It encodes the maximum "local" response of each of the convolutional filters and is therefore translation invariant. In all the following, we consider the last convolutional layer of the examined networks.

Figure 2 visualizes the patches that contribute the most to the image similarity. They correspond either to the same object part or similar parts due to repeated structures. We extract MAC from input images of any resolution or aspect ratio by simply subtracting the mean pixel value (Iandola et al., 2014) from the input images. No crop or change of aspect ratio is required (Azizpour et al., 2014).

The max pooling operation that is performed over a single cell offers translation invariance to the resulting representation. This is in contrast to representation derived from the fully connected layers that requires objects to be aligned. In our case, we assume that objects are up-right and we simply benefit from the rotation tolerance provided by the CNN due to the training data used. The same stands for the tolerance to scale changes.

4 ENCODING REGIONS INTO SHORT VECTORS

This section describes how we exploit the activations of the CNN convolutional layers to derive representations for image regions. Region vectors are aggregated to produce a short signature used in the filtering stage of image retrieval.

Region feature vector. The feature vector $\mathbf{f}_\Omega$ described in Section 3 is a representation for the whole image I. Now, we consider a rectangular region $\mathcal{R} \subseteq \Omega = [1, W] \times [1, H]$, and define the regional feature vector

$$\mathbf{f}_\mathcal{R} = [\mathrm{f}_{\mathcal{R},1} \dots \mathrm{f}_{\mathcal{R},i} \dots \mathrm{f}_{\mathcal{R},K}]^\top \tag{2}$$

where $\mathrm{f}_{\mathcal{R},i} = \max_{p \in \mathcal{R}} \mathcal{X}_i(p)$ is the maximum activation of the i^{th} channel on the considered region.

The regions $\mathcal{R}$ are defined on the space Ω of all valid positions for the considered feature map (and not on the input image plane). A region of size 1 corresponds to a feature vector consisting of a single activation at a particular location. We are now able to construct a representation for multiple regions without re-feeding additional input to the CNN, similarly to recent RNN variants (Ren et al., 2015; Girshick, 2015), which drastically reduces the processing cost.

Now assume a linear mapping of a given region $\mathcal{R}$ back to the original image. The proposed region vector captures a larger image region than the back-projected one, due to the large receptive field. A similar effect occurs in the context of object detection (Iandola et al., 2014), where fully connected layers are applied in a sliding window fashion.

R-MAC: regional maximum activation of convolutions. We now consider a set of R regions of different sizes. The structure of the regions is similar to the one proposed by Razavian et al. (2014b), but we define them on the CNN response maps and not on the original image. We sample square regions at L different scales. At the largest scale ($l = 1$), the region size is determined to be as large as possible, i.e., its height and width are both equal to $\min(W, H)$. The regions are sampled uniformly such that the overlap between consecutive regions is as close as possible to 40%. Remark that the aspect ratio of the original image has an influence on the number m of regions that we extract (1 region only if the input image is square). At every other scale l we uniformly sample $l \times (l + m - 1)$ regions of width $2 \min(W, H)/(l + 1)$, as illustrated in Figure 3 (left).

Then we calculate the feature vector associated with each region, and post-process it with ℓ_2-normalization, PCA-whitening (Jégou & Chum, 2012) and ℓ_2-normalization. We combine the collection of regional feature vectors into a single image vector by summing them and ℓ_2-normalizing in the end. This choice keeps the dimensionality low which is equal to the number of feature channels. However, we show in our experiments that the resulting representation, referred to as R-MAC, offers a significant better performance than the corresponding MAC with same dimensionality. Note, the aggregation of the region vectors can be seen as a simple kernel that cross matches all possible regions, including across different scale.

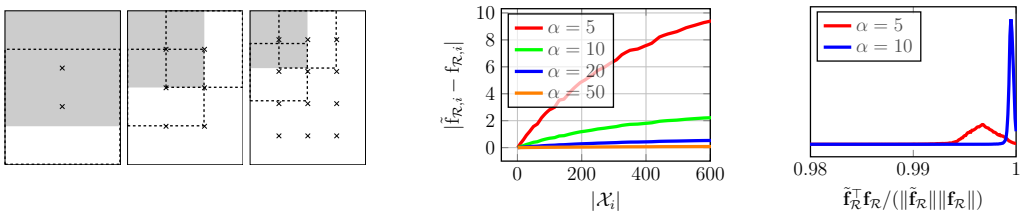


Figure 3: Left: Sample regions extracted at 3 different scales ($l = 1 \dots 3$). We show the top-left region of each scale (gray colored region) and its neighboring regions towards each direction (dashed borders). We depict the centers of all regions with a cross. Middle: Approximation error of the maximum value versus the size of the response set for different values of exponent α. Measurements are performed on 10 randomly selected images by evaluating all possible regions. The responses for this set of images take values in $[0, 151]$. Right: Empirical distribution of the cosine similarity value between the exact vector $\mathbf{f}_{\mathcal{R}}$ and its approximation $\tilde{\mathbf{f}}_{\mathcal{R}}$. Measurements are collected by constructing the exact and approximate vectors of all possible regions on 10 randomly sampled images.

5 Object localization

In this section we propose an extension of integral images to perform approximate max-pooling over a set $\mathcal{X}$ of 2D feature channel response maps, which provide a rough yet efficient localization to our CNN-based method.

Approximate integral max-pooling. Noticing that the responses $\mathcal{X}_i$ are non-negative, we exploit the generalized mean (Dollár et al., 2009) to approximate each feature value $\mathrm{f}_{\mathcal{R},i}$ associated with a given region $\mathcal{R}$ by the estimate

$$\tilde{\mathrm{f}}_{\mathcal{R},i} = \left(\sum_{p \in \mathcal{R}} \mathcal{X}_i(p)^\alpha \right)^{\frac{1}{\alpha}} \approx \max_{p \in \mathcal{R}} \mathcal{X}_i(p) = \mathrm{f}_{\mathcal{R},i}, \tag{3}$$

where the parameter $\alpha > 1$ is such that $\tilde{\mathrm{f}}_i \to \mathrm{f}_i$ when $\alpha \to +\infty$.

Figure 3 (middle) shows the average approximation error $|\tilde{\mathrm{f}}_{\mathcal{R},i} - \mathrm{f}_{\mathcal{R},i}|$ estimated over several image regions. We report the approximation error as a function of the size of the corresponding response set on which the maximum value is computed. The various sizes of response sets are an outcome of using all possible regions. A high value of the exponent α leads to a better approximation, while applying on more elements makes the approximation less precise.

By approximating the maximum in this manner, we can now use integral images (Viola & Jones, 2001) to approximate the regional feature vector $\mathbf{f}_{\mathcal{R}}$ defined on any rectangular region $\mathcal{R}$. For each channel, we construct the integral image of the 2D tensor whose value at position p is equal to $\mathcal{X}_i(p)^\alpha$, $p \in \mathcal{R}$. Then, the sum of Equation (3) is simply given by the sum of 4 terms (Viola & Jones, 2001). This allow us to efficiently compute max-pooling for many regions and therefore to construct the corresponding feature vectors. This is in contrast to the explicit construction of many regions with representation derived from fully connected layers, which is prohibitive due to the need to resize/crop and re-feed each region to the network.

We evaluate the approximation quality by measuring the cosine similarity between the exact vector and its approximate counterpart. The distribution of this similarity is presented in Figure 3 (right) and is measured on all possible regions of 10 randomly selected images. The proposed approximation is very precise even for moderate values of α. We set α equal to 10 in all of our experiments.

Window detection. Let us now assume that there is another image Q depicting a single object, *i.e.* cropped via a bounding box defining the object of interest. We denote as $\mathbf{q}$ the corresponding MAC feature vector. The 2D region, defined on the CNN activations $\mathcal{X}$ of image I, that maximizes the similarity to $\mathbf{q}$ is computed as

$$\hat{\mathcal{R}} = \arg\max_{\mathcal{R} \subseteq \Omega} \frac{\tilde{\mathbf{f}}_{\mathcal{R}}^\top \mathbf{q}}{\|\tilde{\mathbf{f}}_{\mathcal{R}}\| \|\mathbf{q}\|}. \tag{4}$$

The region $\hat{\mathcal{R}}$ maximizing the similarity is mapped back to the original image I with a precision of $(\frac{W}{W_I}, \frac{H}{H_I})$ pixels, providing a rough localization of the object depicted in Q. The corresponding similarity does not take into account all the visual content of image I and is therefore free from the influence of background clutter. The brute-force detection of the optimal region by exhaustive search is expensive, as the number of possible regions is in $\mathcal{O}(W^2H^2)$. In preliminary tests, we have evaluated a globally optimal solution based on *branch and bound* search, as in ESS (Lampert et al., 2009). The necessary bounds are trivially derived for our representation. The search is not significantly sped up in our case: The maxima are not distinct enough and a large number of regions are considered, while the overhead of maintaining the priority queue is high.

AML: approximate max-pooling localization. Instead, we restrict the number of regions that we evaluate and locally refine the best ones with simple heuristics. Candidate regions are uniformly sampled with a *search step* equal to t. In addition, regions having an aspect ratio larger than s times that of the query region are discarded. The parameters of the best region are refined in a coordinate descent manner, while allowing a maximum change of 3 units. The refinement process is repeated up to 5 times. Experiments show that the overlap of the detected region to the optimal one is high.

6 RETRIEVAL, LOCALIZATION AND RE-RANKING

Initial retrieval. The MAC or R-MAC feature vector is computed for all databases images. Similarly, at query time we process the query image and extract the corresponding feature vector. During the filtering stage we directly evaluate cosine similarity between the query and all the database vectors. Therefore, we obtain the initial ranking based on the similarity of MAC or R-MAC vectors.

Re-ranking. We consider a second re-ranking stage, as typically performed in spatial verification (Philbin et al., 2007) with local features. A short-list of N top-ranked images is considered and AML, as described in Section 5, is applied on pairs of query and database images. Note that the query is now represented by the MAC vector, since this is used in AML, while the database image is represented by $\mathcal{X}$. For each re-ranked image we obtain a score given by the region that maximizes the similarity to the query. This similarity is used to re-rank the elements of the short-list. Furthermore, a rough localization of the query object is available.

Remarks: At the filtering stage, whitened MAC (whitening as described in Section 8) or R-MAC can be used, while the localization procedure employs similarity with respect to ℓ_2-normalized MAC. However, once the query object is localized, then, similarity between the query and the detected region is computed via whitened MAC or R-MAC, depending on the chosen filtering method. This similarity score is used to perform re-ranking. The required representation is constructed on query time only for the detected region and is acquired efficiently with integral images.

Query expansion (QE). Re-ranking brings positive images at the very top ranked positions. Then, we collect the 5 top-ranked images, merge them with the query vector, and compute their mean. Finally, the similarity to this mean vector is adopted to re-rank once more the top N images.

7 IMPLEMENTATION DETAILS

We observe that thresholding the response values of $\mathcal{X}$ which are larger than 128 (0.001% of all responses) and mapping each value to the closest smaller integer (floor operation) leads to insignificant losses. This allows the computation of α-th power with a lookup table and speeds-up the construction of integral images. Moreover, we approximate the α-th root of Equation (3) by performing binary search on the same lookup table of α-th power. This process allows the optimal window search to be more efficient.

The response maps represented by $\mathcal{X}$ are sparse (Agrawal et al., 2014). In particular, using the network of Krizhevsky et al. (2012) on Oxford buildings dataset (Philbin et al., 2007) results in 81% of response values being zero, which is convenient for storage purpose. We further decrease the memory requirements by uniformly quantizing the responses into 8 values. This results in more elements mapped to the same value. Therefore, we store the positions of non-zeros values with delta coding and use only 1 byte per non-zero element. Note that an image of resolution equal to 1024×768 corresponds to feature channel response maps of size 30×22 using the same network. Finally, an image requires around 32 kB of memory. At re-ranking time we construct one integral image at a time and use double precision (8 bytes) for its elements.

Table 1: Left: Comparison between the exhaustive sliding window and our alternative of window sampling and refinement. We report the average IoU *w.r.t.* the globally optimal window and the average percentage of windows evaluated *w.r.t.* to the exhaustive search (noted by %W). Measurements are conducted on all pairs of Oxford5k query images and their corresponding positive images. Right: Performance (mAP) of MAC and R-MAC on Oxford5k. Resol. corresponds to the input image resolution (maximum dimension).

Search step t	Aspect ratio change threshold s					
	1.1		1.5		2.0	
	IoU	%W	IoU	%W	IoU	%W
1	81.8	8.9	88.7	27.5	93.7	46.3
2	79.9	0.5	83.8	2.0	86.6	3.6
3	78.7	0.2	81.2	0.5	83.6	0.8
4	77.0	0.1	79.5	0.2	81.5	0.3
5	75.8	0.1	79.0	0.1	80.9	0.2

Network	Resol.	MAC	R-MAC			
			L=1	L=2	L=3	L=4
AlexNet	1024	44.9	47.9	54.6	**56.1**	55.6
	724	44.8	48.4	54.4	54.3	52.6
VGG16	1024	55.2	57.3	64.5	**66.9**	**67.4**
	724	52.2	54.8	58.0	60.9	60.3

8 EXPERIMENTS

This section presents the results of our compact representation for image retrieval, evaluate the localization accuracy AML, and finally employ it for retrieval re-ranking.

Experimental setup. We evaluate the proposed methods on Oxford Buildings dataset (Philbin et al., 2007) and Paris dataset (Philbin et al., 2008), which are composed of 5063 and 6412 images, respectively. We refer to these datasets as Oxford5k and Paris6k. We additionally use 100k Flickr images (Philbin et al., 2007) to compose Oxford105k and Paris106k, respectively. A distractor set of 1 million images from Flickr (Jégou et al., 2010) is additionally used to go at larger scale. Retrieval performance is measured in terms of mean Average Precision (mAP). We follow the standard protocol and use the bounding boxes defined on the query images[1]. These bounding boxes are also employed to evaluate localization accuracy. PCA is learned on Paris6k when testing on Oxford5k and vice versa. In order to be fair, we directly compare our results only to previous methods that do not perform learning on the test set.

The focus of our work is not to train a CNN, but to extract visual descriptors from its convolutional layers. We use networks widely used in the literature: AlexNet by Krizhevsky et al. (2012) and the very deep network (VGG16) by Simonyan & Zisserman (2014). We choose VGG16 instead of VGG19 because we observe that the latter does not always attain better performance while it has higher feature extraction cost. Our representation is extracted from the last pooling layer, which has 256 feature channels for AlexNet and 512 for VGG16. MatConvNet (Vedaldi & Lenc, 2014) is used to extract the features.

Localization accuracy. To evaluate the accuracy of AML, we employ pairs of Oxford5k query images and their corresponding positive images. We first perform exhaustive search to detect the globally optimal window. Then, we apply our speeded-up detector that evaluates fewer regions and in the end refines the best one. In both cases the approximate max-pooling is used for each window evaluation. We report Intersection over Union (IoU) with the optimal window and the percentage of windows evaluated compared to the exhaustive case. Results are shown in Table 1 (left). We provide a large speed-up while maintaining a high overlap with the optimal detection. Recall that our purpose is to apply this detector for fast re-ranking. Measuring IoU provides evidence for localization accuracy, however we observed that it does not directly impact retrieval performance. We finally set $s = 1.1$ and $t = 3$ for re-ranking usage.

In order to evaluate the localization accuracy with respect to ground-truth annotation we cross-match all 5 query images that exist per building. One of them is used as a query (cropped bounding box), while for the other we compare the detected region to the ground-truth annotation. Exhaustive evaluation achieves an IoU equal to 52.6% (52.9%) and the speeded-up approach achieves 51.3% (51.4%) on Oxford5k (Paris6k) datasets. The accuracy loss is limited, while the localization is approximately 180 times faster. AML provides a rough localization at low computational cost. Such

[1]The query regions are cropped and then used as input to the CNN.

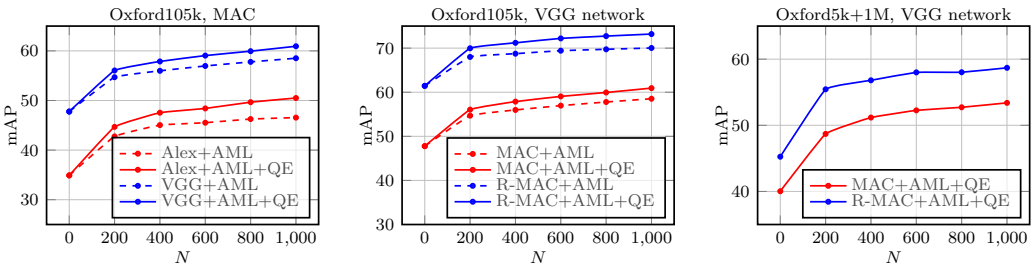


Figure 4: Performance of retrieval with re-ranking by AML versus number of re-ranked images on Oxford105k and Oxford5k combined with 1M distractor images.

Table 2: Performance comparison with state of the art. We report results for compact vector representations (left) and for retrieval approaches employing geometry, re-ranking, query expansion, or vector approximations (right). D = dimensionality. Our approaches are identified with bullets •.

Method	D	Oxf5k	Par6k	Oxf105k	Par106k	Method	Oxf5k	Par6k	Oxf105k	Par106k
Jégou & Zisserman (2014)	1024	56.0	-	50.2	-	Chum et al. (2011)	82.7	80.5	76.7	71.0
Jégou & Zisserman (2014)	128	43.3	-	35.3	-	Danfeng et al. (2011)	81.4	80.3	76.7	-
Babenko et al. (2014)	128	55.7	-	52.3	-	Mikulik et al. (2013)	**84.9**	82.4	**79.5**	77.3
Razavian et al. (2014b)	256	53.3	67.0	48.9	-	Shen et al. (2014)	75.2	74.1	72.9	-
Babenko & Lempitsky (2015)	256	53.1	-	50.1	-	Tao et al. (2014)	77.8	-	-	-
R-MAC •	256	56.1	72.9	47.0	60.1	Tolias et al. (2015)	80.4	77.0	75.0	-
R-MAC •	512	**66.9**	**83.0**	**61.6**	**75.7**	R-MAC+AML+QE •	77.3	**86.5**	73.2	**79.8**

a setup results in an average re-ranking query time of 2.9 sec using AlexNet, when re-ranking 1000 images with a single threaded implementation.

Retrieval and re-ranking. We evaluate retrieval performance using MAC and R-MAC compact representations. The MAC vectors are ℓ_2-normalized, PCA-whitened and ℓ_2-normalized once more, while the corresponding processing of the R-MAC is as described in Section 4. Table 1 (right) presents the results on Oxford5k. We evaluate different input image resolutions and observe that the original image size (1024) provides higher performance. Note that MAC is similar to the one proposed by Azizpour et al. (2014), however their process remains constrained by standard input size and aspect ratio. The proposed R-MAC gives a large performance improvement at no extra cost, as both feature vectors have exactly the same dimensionality. Regions of different scales are aggregated together, meaning that $L = 3$ combines regions at scales $l = 1$, $l = 2$, and $l = 3$. We set $L = 3$ in the following. In order to decompose the components of R-MAC, we construct R-MAC by aggregating only regions of $l = 3$. It achieves mAP equal to 63.0 on Oxford5k with VGG16. Aggregating both regions of $l = 2$ and $l = 3$ improves to 65.4. Finally, adding $l = 1$ (original R-MAC) performs 66.9 (see Table 1 right). Filtering time is 12 ms on average for Oxford105k.

Next, we employ AML for image re-ranking and conduct performance evaluation on Oxford105k by re-ranking up to 1000 images. The performance is consistently improved as shown in Figure 4. R-MAC brings a larger benefit and VGG16 performs better than AlexNet. Query expansion, as described in Section 6, improves the performance at low extra cost, since similarity is re-computed only for the re-ranked short-list. Finally, we carry out experiment at larger scale with 1M distractor images and present results in Figure 4. AML improves the performance by 13% mAP.

Examples of ranking using MAC and re-ranking using AML are presented in Figure 5. Recall that we only provide a rough object localization, since our main goal is to obtain improved image similarity. Furthermore, the provided localization is accurate enough for re-ranking.

Comparison to the state of the art. We compare the proposed methods to state-of-the-art performance of compact representations and approaches based on local features that perform precise

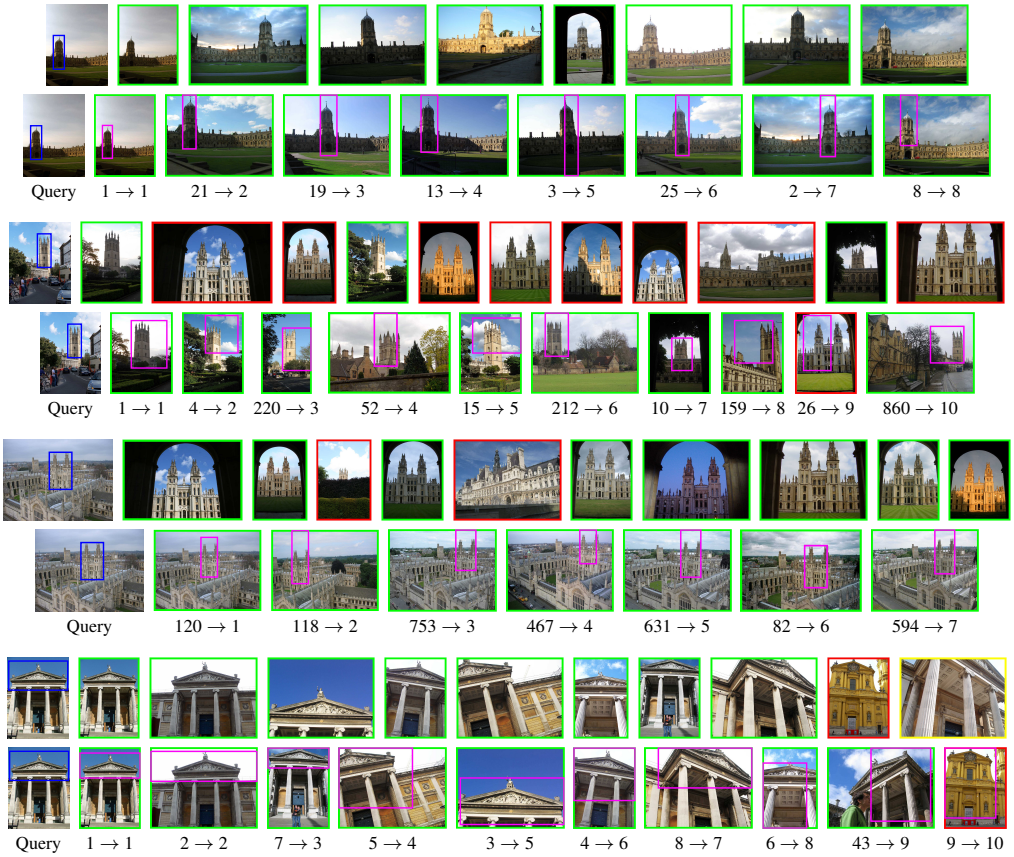


Figure 5: Examples of top retrieved images before (top) and after (bottom) re-ranking with AML. On the left we show the query image and depict the bounding box in blue color. When re-ranking is used, we present the top ranked images and report for each image its initial and final ranking. The localization window is shown in magenta, while positive/negative/junk images are depicted with green/red/yellow border.

descriptor matching, re-ranking or query expansion. Results are shown in Table 2[2]. AlexNex and VGG16 are used to produce the 256D and 512D vectors for R-MAC, respectively. Regarding the compact representations, our short-sized R-MAC outperforms all other approaches. The better performance on Paris is inherited by the nature of the pre-trained networks; the baseline MAC with VGG achieves 55.2 on Oxford5k and 74.7 on Paris6k.

Unlike previous description schemes derived from CNN layers, our approach compete with the best approaches based on local features for geometric matching and query expansion. Our AML can even outperform them: while our results are lower on Oxford, we achieve the best performance on Paris and, to the best of our knowledge, outperform all published results on this benchmark. Higher scores on Paris6k are reported by Arandjelovic & Zisserman (2012) (91.0) and by Zhong et al. (2015) (91.5). These are achieved by learning the codebook on Paris6k itself and by performing pre-processing of the indexed dataset.

Discussion about other CNN-based approaches. Razavian et al. (2014b) propose to perform region cross-matching and accumulate the maximum similarity per query region. We evaluate this cross-matching process on the collection of regional vectors used in R-MAC; we simply skip the final aggregation process and keep the regional vectors individually. The cross-matching achieves 75.2% mAP on Oxford5k as a filtering stage, while re-ranking with AML on top of this acts in a complementary way and increases the performance up to 78.1%. However, cross-matching has

[2]Small differences of scores compared to the first version of the manuscript on arxiv are due to a slightly different evaluation protocol used before. Now, the evaluation protocol is the standard one for these datasets.

two drawbacks. Firstly, the region vectors have to be stored individually and increase the memory requirements by a factor of $|R|$, where $|R|$ is the number of extracted regions. Secondly, the complexity cost is linear in the number of indexed images and quite high since it requires to compute $|R|^2$ (*e.g.* 1024 (Razavian et al., 2014b)) inner products per image. The work of Razavian et al. (2014b) follows a non-standard evaluation protocol by enlarging the provided query bounding boxes. In addition, the cost of their feature extraction is extremely high since they feed 32 images of resolution 576×576 to the CNN. The recent work of Xie et al. (2015) is quite similar to theirs and is applied on both retrieval and classification.

Babenko & Lempitsky (2015) show that global sum-pooling on convolutional layer activations is better than max-pooling when the final image vectors are PCA-whitened. When whitening is not employed, then the latter is better. In the context of object localization we efficiently evaluate a large number of candidate regions on query time with AML. Performing whitening on each candidate region vector significantly increases the cost and is prohibitive for this task. We switch max-pooling to sum-pooling for both our proposed R-MAC and AML and test performance. Note that sum-pooling is a special case of our integral max-pooling with $\alpha = 1$. Switching to sum-pooling makes R-MAC perform 69.8 and R-MAC +AML +QE perform 76.9 on Paris106k. These scores are directly comparable to our scores in Table 2 and reveal that our choice is consistently better in all cases within our pipeline.

9 CONCLUSIONS

In this work, we re-visit both filtering and re-ranking retrieval stages by employing CNN activations of convolutional layers. Our compact vector representation encodes several image regions with simple aggregation method and is shown to outperform state-of-the-art competitors. Our localization increases the performance of the retrieval system that is initially based on a compact representation. The same CNN information adopted during the filtering stage is employed for re-ranking as well. Our approach competes with state-of-the-art methods that employ costly geometric matching or query expansion and we achieve the highest performance on Paris dataset, and provided a much better performance than existing approaches built upon CNN features. A very recent work (Arandjelovic et al., 2015) shows how MAC performance is improved by end-to-end fine tunning where the objective is based on MAC similarity.

REFERENCES

Agrawal, Pulkit, Girshick, Ross, and Malik, Jitendra. Analyzing the performance of multilayer neural networks for object recognition. In *ECCV*, 2014.

An, Senjian, Peursum, Patrick, Liu, Wanquan, and Venkatesh, Svetha. Efficient algorithms for subwindow search in object detection and localization. In *CVPR*, 2009.

Arandjelovic, Relja and Zisserman, Andrew. Three things everyone should know to improve object retrieval. In *CVPR*, Jun. 2012.

Arandjelovic, Relja and Zisserman, Andrew. All about VLAD. In *CVPR*, Jun. 2013.

Arandjelovic, Relja, Gronat, Petr, Torii, Akihiko, Pajdla, Tomas, and Sivic, Josef. Netvlad: Cnn architecture for weakly supervised place recognition. In *arXiv*, 2015.

Avrithis, Yannis and Kalantidis, Yannis. Approximate gaussian mixtures for large scale vocabularies. In *ECCV*, 2012.

Avrithis, Yannis and Tolias, Giorgos. Hough pyramid matching: Speeded-up geometry re-ranking for large scale image retrieval. *IJCV*, 2014.

Azizpour, Hossein, Razavian, Ali Sharif, Sullivan, Josephine, Maki, Atsuto, and Carlsson, Stefan. From generic to specific deep representations for visual recognition. In *arXiv*, 2014.

Babenko, Artem and Lempitsky, Victor. Aggregating deep convolutional features for image retrieval. In *ICCV*, 2015.

Babenko, Artem, Slesarev, Anton, Chigorin, Alexandr, and Lempitsky, Victor. Neural codes for image retrieval. In *ECCV*, Sep. 2014.

Bentley, Joe. *Programming Pearls, 2/E*. Addison-Wesley Professional, 1999.

Chen, Qiang, Song, Zheng, Feris, Rogerio, Datta, Amitava, Cao, Liangliang, Huang, Zhongyang, and Yan, Shuicheng. Efficient maximum appearance search for large-scale object detection. In *CVPR*, 2013.

Chum, Ondrej, Mikulik, A., Perdoch, M., and Matas, J. Total recall II: Query expansion revisited. In *CVPR*, Jun. 2011.

Danfeng, Qin, Gammeter, S., Bossard, L., Quack, T., and Gool, L. Van. Hello neighbor: Accurate object retrieval with k-reciprocal nearest neighbors. In *CVPR*, 2011.

Dollár, Piotr, Tu, Zhuowen, Perona, Pietro, and Belongie, Serge. Integral channel features. In *BMVC*, 2009.

Donahue, Jeff, Jia, Yangqing, Vinyals, Oriol, Hoffman, Judy, Zhang, Ning, Tzeng, Eric, and Darrell, Trevor. Decaf: A deep convolutional activation feature for generic visual recognition. In *arXiv*, 2013.

Girshick, Ross. Fast r-cnn. In *arXiv*, 2015.

Girshick, Ross, Donahue, Jeff, Darrell, Trevor, and Malik, Jitendra. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

Gong, Yunchao, Wang, Liwei, Guo, Ruiqi, and Lazebnik, Svetlana. Multi-scale orderless pooling of deep convolutional activation features. In *ECCV*, 2014.

Iandola, Forrest, Moskewicz, Matt, Karayev, Sergey, Girshick, Ross, Darrell, Trevor, and Keutzer, Kurt. Densenet: Implementing efficient convnet descriptor pyramids. In *arxiv*, 2014.

Jégou, Hervé and Chum, Ondrej. Negative evidences and co-occurences in image retrieval: The benefit of PCA and whitening. In *ECCV*, Oct. 2012.

Jégou, Hervé and Zisserman, Andrew. Triangulation embedding and democratic aggregation for image search. In *CVPR*, 2014.

Jégou, Hervé, Douze, Matthijs, and Schmid, Cordelia. Improving bag-of-features for large scale image search. *IJCV*, 87(3), Feb. 2010.

Jégou, Hervé, Perronnin, Florent, Douze, Matthijs, Sánchez, Jorge, Pérez, Patrick, and Schmid, Cordelia. Aggregating local descriptors into compact codes. *Trans. PAMI*, Sep. 2012.

Kalantidis, Yannis, Mellina, Clayton, and Osindero, Simon. Cross-dimensional weighting for aggregated deep convolutional features. In *arXiv preprint arXiv:1512.04065*, 2015.

Krizhevsky, Alex, Sutskever, Ilya, and Hinton, Geoffrey E. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

Lampert, Christoph H. Detecting objects in large image collections and videos by efficient subimage retrieval. In *ICCV*, 2009.

Lampert, Christoph H, Blaschko, Matthew B, and Hofmann, Thomas. Efficient subwindow search: A branch and bound framework for object localization. *Trans. PAMI*, 31(12):2129–2142, 2009.

Lin, Zhe and Brandt, Jonathan. A local bag-of-features model for large-scale object retrieval. In *ECCV*, 2010.

Lowe, David. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, Nov. 2004.

Mikulik, Andrej, Perdoch, Michal, Chum, Ondřej, and Matas, Jiří. Learning vocabularies over a fine quantization. *IJCV*, 103(1), 2013.

Oquab, Maxime, Bottou, Leon, Laptev, Ivan, and Sivic, Josef. Learning and transferring mid-level image representations using convolutional neural networks. In *CVPR*, 2014.

Papandreou, George, Kokkinos, Iasonas, and Savalle, Pierre-André. Untangling local and global deformations in deep convolutional networks for image classification and sliding window detection. In *arXiv*, 2014.

Philbin, James, Chum, Ondrej, Isard, Michael, Sivic, Josef, and Zisserman, Andrew. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, Jun. 2007.

Philbin, James, Chum, Ondrej, Isard, Michael, Sivic, Josef, and Zisserman, Andrew. Lost in quantization: Improving particular object retrieval in large scale image databases. In *CVPR*, Jun. 2008.

Radenović, Filip, Jegou, Herve, and Chum, Ondrej. Multiple measurements and joint dimensionality reduction for large scale image search with short vectors. In *ICMR*, 2015.

Razavian, Ali Sharif, Azizpour, Hossein, Sullivan, Josephine, and Carlsson, Stefan. CNN features off-the-shelf: An astounding baseline for recognition. In *CVPRW*, 2014a.

Razavian, Ali Sharif, Sullivan, Josephine, Maki, Atsuto, and Carlsson, Stefan. A baseline for visual instance retrieval with deep convolutional networks. In *arXiv*, 2014b.

Ren, Shaoqing, He, Kaiming, Girshick, Ross, and Sun, Jian. Faster r-cnn: Towards real-time object detection with region proposal networks. In *arXiv*, 2015.

Shen, Xiaohui, Lin, Zhe, Brandt, Jonathan, and Wu, Ying. Spatially-constrained similarity measure for large-scale object retrieval. *Trans. PAMI*, 36(6):1229–1241, 2014.

Sicre, Ronan and Jurie, Frdric. Discriminative part model for visual recognition. *CVIU*, 141:28 – 37, 2015. ISSN 1077-3142.

Simonyan, Karen and Zisserman, Andrew. Very deep convolutional networks for large-scale image recognition. In *arXiv*, 2014.

Sivic, Josef and Zisserman, Andrew. Video Google: A text retrieval approach to object matching in videos. In *ICCV*, 2003.

Tao, Ran, Gavves, Efstratios, Snoek, Cees GM, and Smeulders, Arnold WM. Locality in generic instance search from one example. In *CVPR*, 2014.

Tolias, Giorgos, Avrithis, Yannis, and Jégou, Hervé. Image search with selective match kernels: aggregation across single and multiple images. *IJCV*, 2015.

Uijlings, Jasper, Van de Sande, Koen, Gevers, Theo, and Smeulders, Arnold. Selective search for object recognition. *IJCV*, 104(2):154–171, 2013.

Van de Sande, Koen EA, Snoek, Cees GM, and Smeulders, Arnold WM. Fisher and VLAD with flair. In *CVPR*, 2014.

Vedaldi, Andrea and Lenc, Karel. Matconvnet-convolutional neural networks for matlab. In *arXiv*, 2014.

Viola, Paul and Jones, Michael. Robust real-time object detection. *IJCV*, 4:34–47, 2001.

Xie, Lingxi, Tian, Q, Hong, R, and Zhang, B. Image classification and retrieval are one. In *ICMR*, 2015.

Zhong, Zhiyuan, Zhu, Jianke, and Hoi, Steven CH. Fast object retrieval using direct spatial matching. *IEEE Trans. on Multimedia*, 17(8):1391–1397, 2015.

II Fine-tuning CNN Image Retrieval with No Human Annotation

Title: Fine-tuning CNN Image Retrieval with No Human Annotation
Authors: F Radenović, G Tolias, O Chum
Published at: PAMI 2019

Fine-tuning CNN Image Retrieval with No Human Annotation

Filip Radenović Giorgos Tolias Ondřej Chum

Abstract—Image descriptors based on activations of Convolutional Neural Networks (CNNs) have become dominant in image retrieval due to their discriminative power, compactness of representation, and search efficiency. Training of CNNs, either from scratch or fine-tuning, requires a large amount of annotated data, where a high quality of annotation is often crucial. In this work, we propose to fine-tune CNNs for image retrieval on a large collection of unordered images in a fully automated manner. Reconstructed 3D models obtained by the state-of-the-art retrieval and structure-from-motion methods guide the selection of the training data. We show that both hard-positive and hard-negative examples, selected by exploiting the geometry and the camera positions available from the 3D models, enhance the performance of particular-object retrieval. CNN descriptor whitening discriminatively learned from the same training data outperforms commonly used PCA whitening. We propose a novel trainable Generalized-Mean (GeM) pooling layer that generalizes max and average pooling and show that it boosts retrieval performance. Applying the proposed method to the VGG network achieves state-of-the-art performance on the standard benchmarks: Oxford Buildings, Paris, and Holidays datasets.

1 Introduction

In instance image retrieval an image of a particular object, depicted in a query, is sought in a large, unordered collection of images. Convolutional neural networks (CNNs) have recently provided an attractive solution to this problem. In addition to leaving a small memory footprint, the CNN-based approaches achieve high accuracy. Neural networks have attracted a lot of attention after the success of Krizhevsky *et al.* [1] in the image-classification task. Their success is mainly due to the use of very large annotated datasets, *e.g.* ImageNet [2]. The acquisition of the training data is a costly process of manual annotation, often prone to errors. Networks trained for image classification have shown strong adaptation abilities [3]. Specifically, using activations of CNNs, which were trained for the task of classification, as off-the-shelf image descriptors [4], [5] and adapting them for a number of tasks [6], [7], [8] have shown acceptable results. In particular, for image retrieval, a number of approaches directly use the network activations as image features and successfully perform image search [8], [9], [10], [11], [12].

Fine-tuning of the network, *i.e.* initialization by a pre-trained classification network and then training for a different task, is an alternative to a direct application of a pre-trained network. Fine-tuning significantly improves the adaptation ability [13], [14]; however, further annotation of training data is required. The first fine-tuning approach for image retrieval is proposed by Babenko *et al.* [15], in which a significant amount of manual effort was required to collect images and label them as specific building classes. They improved retrieval accuracy; however, their formulation is much closer to classification than to the desired properties of instance retrieval. In another approach, Arandjelovic *et al.* [16] perform fine-tuning guided by geo-tagged image databases and, similar to our work, they directly optimize the similarity measure to be used in the final task by selecting *matching* and *non-matching* pairs to perform the training.

In contrast to previous methods of training-data acquisition for image search, we dispense with the need for manually annotated data or any assumptions on the training dataset. We achieve this by exploiting the geometry and the camera positions from 3D models reconstructed automatically by a structure-from-motion (SfM) pipeline. The state-of-the-art retrieval-SfM pipeline [17] takes an unordered image collection as input and attempts to build all possible 3D models. To make the process efficient, fast image clustering is employed. A number of image clustering methods based on local features have been introduced [18], [19], [20]. Due to spatial verification, the *clusters* discovered by these methods are reliable. In fact, the methods provide not only clusters, but also a matching graph or sub-graph on the cluster images. The SfM filters out virtually all mismatched images and provides image-to-model matches and camera positions for all matched images in the cluster. The whole process, from unordered collection of images to detailed 3D reconstructions, is fully automatic. Finally, the 3D models guide the selection of matching and non-matching pairs. We propose to exploit the training data acquired by the same procedure in the descriptor post-processing stage to learn a discriminative whitening.

An additional contribution of this work lies in the introduction of a novel pooling layer after the convolutional layers. Previously, a number of approaches have been used. These range from fully-connected layers [8], [15], to different global-pooling layers, *e.g.* max pooling [9], average pooling [10], hybrid pooling [21], weighted average pooling [11], and regional pooling [12]. We propose a pooling layer based on a generalized-mean that has learnable parameters, either one global or one per output dimension. Both max and average pooling are its special cases. Our experiments show that it offers a significant performance boost over standard non-trainable pooling layers. Our architecture is shown in Figure 1.

Affiliation: *Visual Recognition Group, Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague*

E-mail: {*filip.radenovic,giorgos.tolias,chum*}*@cmp.felk.cvut.cz*

Data, networks, and code: *cmp.felk.cvut.cz/cnnimageretrieval*

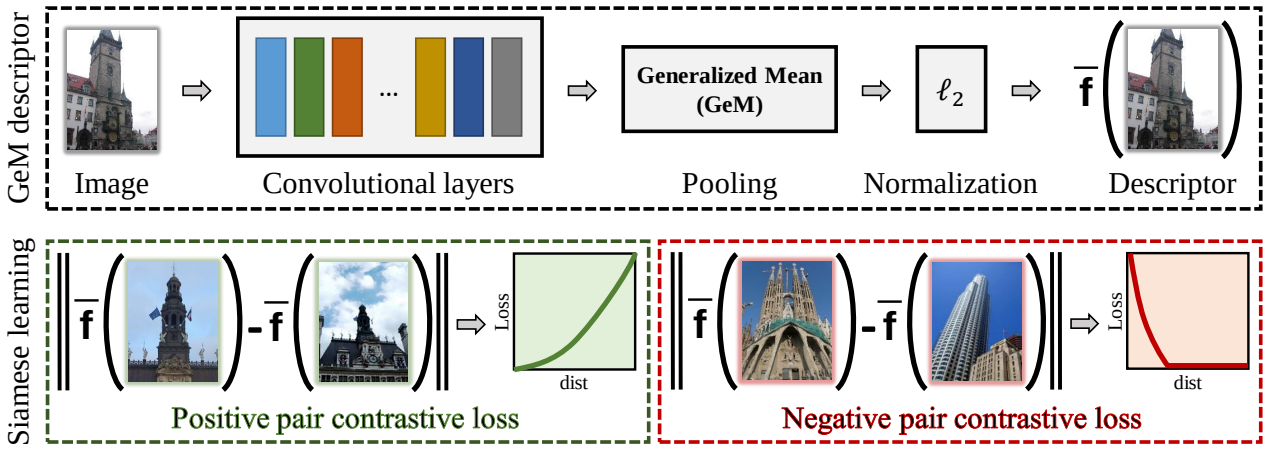


Fig. 1. The architecture of our network with the contrastive loss used at training time. A single vector $\bar{\mathbf{f}}$ is extracted to represent an image.

To summarize, we address the unsupervised fine-tuning of CNNs for image retrieval. In particular, we make the following contributions: (1) We exploit SfM information and enforce, not only hard non-matching (*negative*), but also hard-matching (*positive*) examples for CNN training. This is shown to enhance the derived image representation. We show that compared to previous supervised approaches, the variability in the training data from 3D reconstructions delivers superior performance in the image-retrieval task. (2) We show that the whitening traditionally performed on short representations [22] is, in some cases, unstable. We propose to learn the whitening through the same training data. Its effect is complementary to fine-tuning and it further boosts performance. Also, performing whitening as a post-processing step is better and much faster to train compared to learning it end-to-end. (3) We propose a trainable pooling layer that generalizes existing popular pooling schemes for CNNs. It significantly improves the retrieval performance while preserving the same descriptor dimensionality. (4) In addition, we propose a novel α-weighted query expansion that is more robust compared to the standard average query expansion technique widely used for compact image representations. (5) Finally, we set a new state-of-the-art result for Oxford Buildings, Paris, and Holidays datasets by re-training the commonly used CNN architectures, such as AlexNet [1], VGG [23], and ResNet [24].

This manuscript is an extension of our previous work [25]. We additionally propose a novel pooling layer (Section 3.2), a novel multi-scale image representation (Section 5.2), and a novel query expansion method (Section 5.3). Each one of the newly proposed methods boosts image-retrieval performance, and is accompanied by experiments that give useful insights. In addition, we provide an extended related work discussion including the different pooling procedures used in prior CNN work and descriptor whitening. Finally, we compare our approach to the concurrent related work of Gordo *et al.* [26], [27]. They significantly improve the retrieval performance through end-to-end learning which incorporates building-specific region proposals. In contrast to their work, we focus on the importance of hard-training data examples, and employ a much simpler but equally powerful pooling layer.

The rest of the paper is organized as follows. Related work is discussed in Section 2, our network architecture, learning procedure, and search process is presented in Section 3, and our proposed automatic acquisition of the training data is described in Section 4. Finally, in Section 5 we perform an extensive quantitative and qualitative evaluation of all proposed novelties with different CNN architectures and compare to the state of the art.

2 Related work

The CNN-based representation is an appealing solution for image retrieval and in particular for compact image representations. Previous compact descriptors are typically constructed by an aggregation of local features, where representatives are Fisher vectors [28], VLAD [29] and alternatives [30], [31], [32]. Impressively, in this work we show that CNNs dominate the image search task by outperforming state-of-the-art methods that have reached a higher level of maturity by incorporating large visual codebooks [33], [34], spatial verification [33], [35] and query expansion [36], [37], [38].

In this work, instance retrieval is cast as a metric learning problem, *i.e.*, an image embedding is learned so that the Euclidean distance captures the similarity well. Typical architectures for metric learning, such as the two-branch siamese [39], [40], [41] or triplet networks [42], [43], [44] employ *matching* and *non-matching* pairs to perform the training and better suit to this task. Here, the problem of annotations is even more pronounced, *i.e.*, for classification one needs only object category label, while for particular objects the labels have to be per image pair. Two images from the same object category could potentially be completely different, *e.g.*, different viewpoints of the building or different buildings. We solve this problem in a fully automated manner, without any human intervention, by starting from a large unordered image collection.

In the following text we discuss the related work for our main contributions, *i.e.*, the training data collection, the pooling approach to construct a global image descriptor, and the descriptor whitening.

2.1 Training data

A variety of previous methods apply CNN activations on the task of image retrieval [8], [9], [10], [11], [12], [45]. The achieved accuracy on retrieval is evidence for the generalization properties of CNNs. The employed networks are trained for image classification using ImageNet dataset [2] by minimizing classification error. Babenko *et al.* [15] go one step further and re-train such networks with a dataset that is closer to the target task. They perform training with object classes that correspond to particular landmarks/buildings. Performance is improved on standard retrieval benchmarks. Despite the achievement, still, the final metric and the utilized layers are different to the ones actually optimized during learning.

Constructing such training datasets requires manual effort. In recent work, geo-tagged datasets with timestamps offer the ground for weakly-supervised fine-tuning of a triplet network [16]. Two images taken far from each other can be easily considered as non-matching, while matching examples are picked by the most similar nearby images. In the latter case, similarity is defined by the current representation of the CNN. This is the first approach that performs end-to-end fine-tuning for image retrieval and, in particular, for the geo-localization task. The used training images are now more relevant to the final task. We differentiate by discovering matching and non-matching image pairs in an unsupervised way. Moreover, we derive matching examples based on 3D reconstructions which allows for harder examples.

Even though hard-negative mining is a standard process [6], [16], this is not the case with hard-positive examples. Mining of hard positive examples have been exploited in the work Simo-Serra *et al.* [46], where patch-level examples were extracted though the guidance from a 3D reconstruction. Hard-positive pairs have to be sampled carefully. Extremely hard positive examples (such as minimal overlap between images or extreme scale change) do not allow to generalize and lead to over-fitting.

A concurrent work to ours also uses local features and geometric verification to select positive examples [26]. In contrast to our fully unsupervised method, they start from a landmarks dataset, which had to be manually cleaned, and the landmark labels of the dataset, rather than the geometry, were used to avoid exhaustive evaluation. The same training dataset is used by Noh *et al.* [47] to learn global image descriptors using a saliency mask. However, during test time the CNN activations are seen as local descriptors, indexed independently, and used for a subsequent spatial-verification stage. Such approach boosts accuracy compared to global descriptors, but at the cost of much higher complexity.

2.2 Pooling method

Early approaches to applying CNNs for image retrieval included methods that set the fully-connected layer activations to be the global image descriptors [8], [15]. The work by Razavian *et al.* [9] moves the focus to the activations of convolutional layers followed by a global-pooling operation. A compact image representation is constructed in this fashion with dimensionality equivalent to the number of feature maps of the corresponding convolutional layer. In particular, they propose to use max pooling, which is later approximated with integral max pooling [12].

Sum pooling was initially proposed by Babenko and Lempitsky [10], which was shown to perform well especially due to the subsequent descriptor whitening. One step further is the weighted sum pooling of Kalantidis *et al.* [11], which can also be seen as a way to perform transfer learning. Popular encodings such as BoW, VLAD, and Fisher vectors are adapted in the context of CNN activations in the work of Mohedano *et al.* [48], Arandjelovic *et al.* [16], and Ong *et al.* [49], respectively. Sum pooling is employed once an appropriate embedding is performed beforehand.

A hybrid scheme is the R-MAC method [12], which performs max pooling over regions and finally sum pooling of the regional descriptors. Mixed pooling is proposed globally for retrieval [21] and the standard local pooling is used for object recognition [50]. It is a linear combination of max and sum pooling. A generalization scheme similar to ours is proposed in the work of Cohen *et al.* [51] but in a different context. They replace the standard local max pooling with the generalized one. Finally, generalized mean is used by Morère *et al.* [52] to pool the similarity values under multiple transformations.

2.3 Descriptor whitening

Whitening the data representation is known to be very essential for image retrieval since the work of Jégou and Chum [22]. Their interpretation lies on jointly down-weighting co-occurrences and, thus, handling the problem of over-counting. The benefit of whitening is further emphasized in the case of CNN-based descriptors [5], [10], [12]. Whitening is commonly learned from a generative model in an unsupervised way by PCA on an independent dataset.

We propose to learn the whitening transform in a discriminative manner, using the same acquisition procedure of the training data from 3D models. A similar approach has been used to whiten local-feature descriptors by Mikolajczyk and Matas [53].

In constrast, Gordo *et al.* [26] learn the whitening in the CNN in an end-to-end manner. In our experiments we found this choice to be at most as good as the descriptor post-processing and less efficient due to slower convergence of the learning.

3 Architecture, Learning, Search

In this section we describe the network architecture and present the proposed generalized-pooling layer. Then we explain the process of fine-tuning using the contrastive loss and a two-branch network. We describe how, after fine-tuning, we use the same training data to learn projections that appear to be an effective post-processing step. Finally, we describe the image representation, search process, and a novel query expansion scheme. Our proposed architecture is depicted in Figure 1.

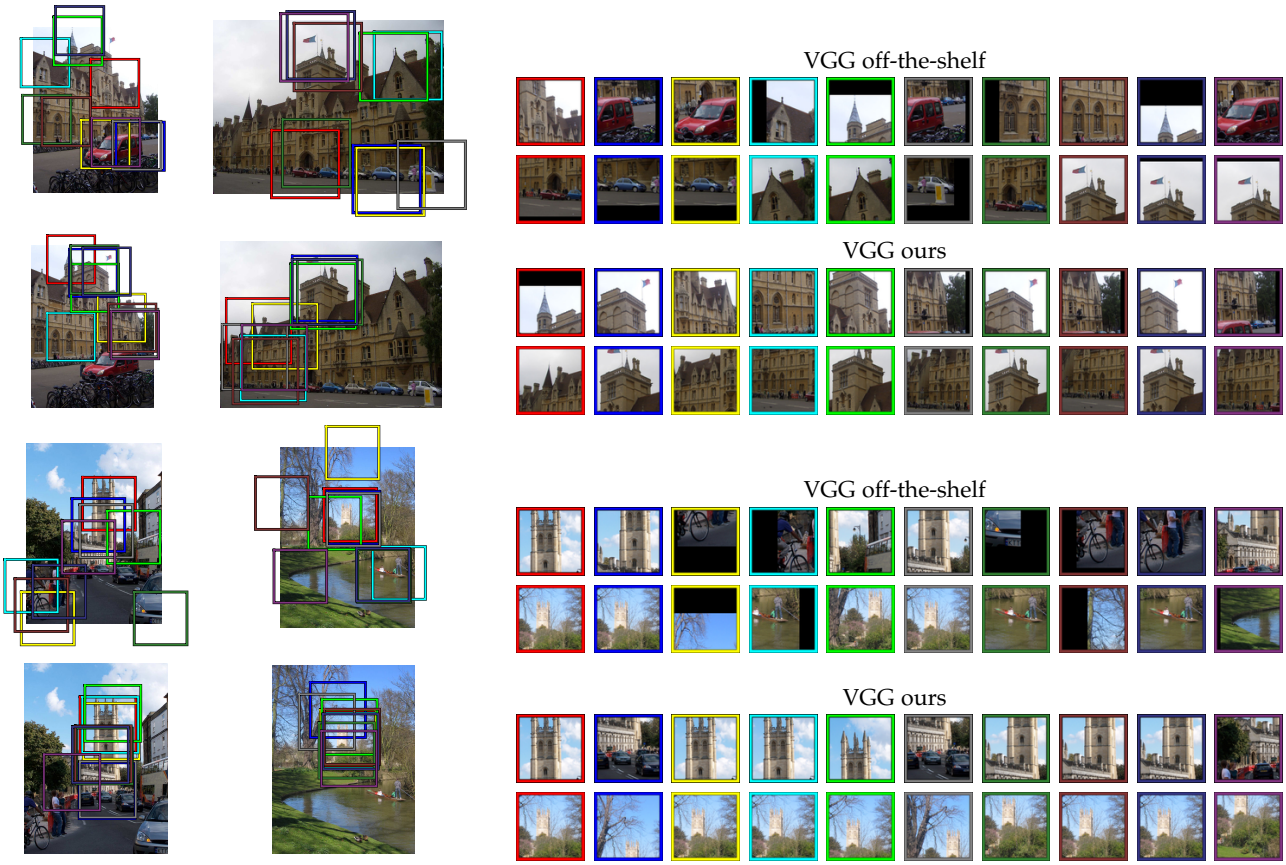


Fig. 2. Visualization of image regions that correspond to MAC descriptor dimensions that have the highest contribution, *i.e.* large product of descriptor elements, to the pairwise image similarity. The example uses VGG before (top) and after (bottom) fine-tuning. Same color corresponds to the same descriptor component (feature map) per image pair. The patch size is equal to the receptive field of the last local pooling layer.

3.1 Fully convolutional network

Our methodology applies to any fully convolutional CNN [54]. In practice, popular CNNs for generic object recognition are adopted, such as AlexNet [1], VGG [23], or ResNet [24], while their fully-connected layers are discarded. This provides a good initialization to perform the fine-tuning.

Now, given an input image, the output is a 3D tensor $\mathcal{X}$ of $W \times H \times K$ dimensions, where K is the number of feature maps in the last layer. Let $\mathcal{X}_k$ be the set of $W \times H$ activations for feature map $k \in \{1 \dots K\}$. The network output consists of K such activation sets or 2D feature maps. We additionally assume that the very last layer is a Rectified Linear Unit (ReLU) such that $\mathcal{X}$ is non-negative.

3.2 Generalized-mean pooling and image descriptor

We now add a pooling layer that takes $\mathcal{X}$ as an input and produces a vector $\mathbf{f}$ as an output of the pooling process. This vector in the case of the conventional global max pooling (MAC vector [9], [12]) is given by

$$\mathbf{f}^{(m)} = [\mathrm{f}_1^{(m)} \dots \mathrm{f}_k^{(m)} \dots \mathrm{f}_K^{(m)}]^\top, \qquad \mathrm{f}_k^{(m)} = \max_{x \in \mathcal{X}_k} x, \tag{1}$$

while for average pooling (SPoC vector [10]) by

$$\mathbf{f}^{(a)} = [\mathrm{f}_1^{(a)} \dots \mathrm{f}_k^{(a)} \dots \mathrm{f}_K^{(a)}]^\top, \qquad \mathrm{f}_k^{(a)} = \frac{1}{|\mathcal{X}_k|} \sum_{x \in \mathcal{X}_k} x. \tag{2}$$

Instead, we exploit the generalized mean [55] and propose to use generalized-mean (GeM) pooling whose result is given by

$$\mathbf{f}^{(g)} = [\mathrm{f}_1^{(g)} \dots \mathrm{f}_k^{(g)} \dots \mathrm{f}_K^{(g)}]^\top, \quad \mathrm{f}_k^{(g)} = \left(\frac{1}{|\mathcal{X}_k|} \sum_{x \in \mathcal{X}_k} x^{p_k} \right)^{\frac{1}{p_k}}. \tag{3}$$

Pooling methods (1) and (2) are special cases of GeM pooling given in (3), *i.e.*, max pooling when $p_k \to \infty$ and average pooling for $p_k = 1$. The feature vector finally consists of a single value per feature map, *i.e.* the generalized-mean activation, and its dimensionality is equal to K. For many popular networks this is equal to 256, 512 or 2048, making it a compact image representation.

The pooling parameter p_k can be manually set or learned since this operation is differentiable and can be part of the back-propagation. The corresponding derivatives (while skipping the superscript (g) for brevity) are given by

$$\frac{\partial \mathrm{f}_k}{\partial x_i} = \frac{1}{|\mathcal{X}_k|} \mathrm{f}_k^{1-p_k} {x_i}^{p_k - 1}, \tag{4}$$

$$\frac{\partial \mathrm{f}_k}{\partial p_k} = \frac{\mathrm{f}_k}{p_k^2} \left(\log \frac{|\mathcal{X}_k|}{\sum_{x \in \mathcal{X}_k} x^{p_k}} + p_k \frac{\sum_{x \in \mathcal{X}_k} x^{p_k} \log x}{\sum_{x \in \mathcal{X}_k} x^{p_k}} \right). \tag{5}$$

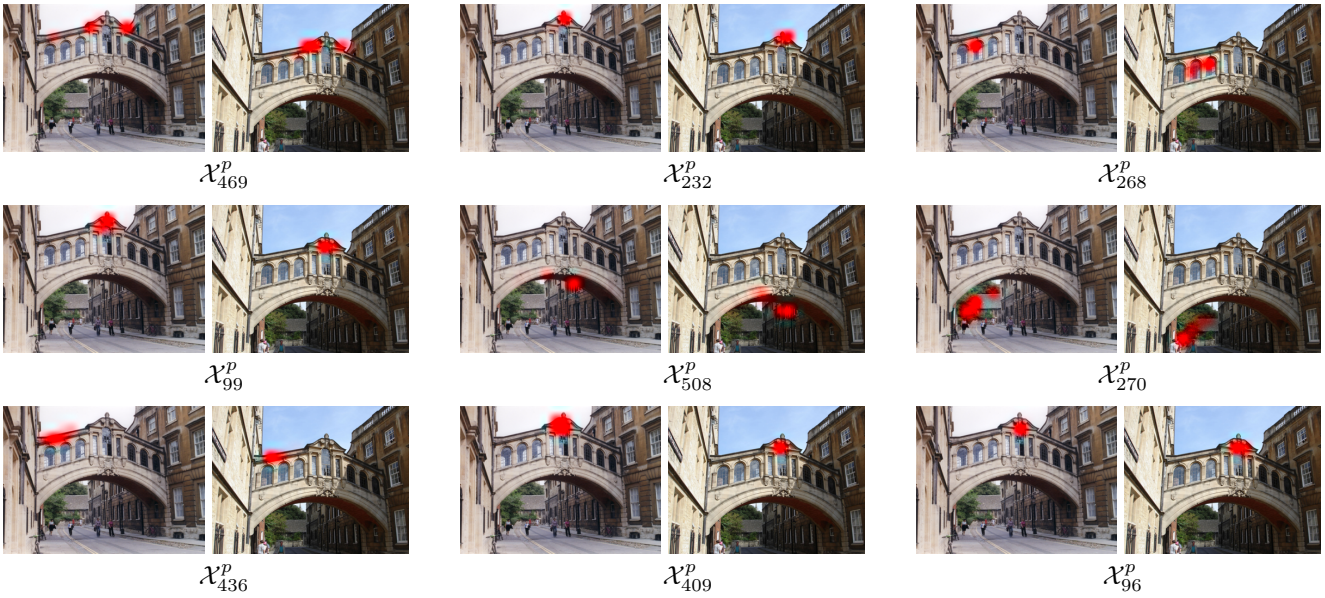


Fig. 3. Visualization of $\mathcal{X}^p_k$ projected on the original image for a pair of query-database image. The 9 feature maps shown are the ones that score highly, *i.e.* large product of GeM descriptor components, for the database image (right) but low for the top-ranked non-matching images. The example uses fine-tuned VGG with GeM and single p for all feature maps, which converged to 2.92.

There is a different pooling parameter per feature map in (3), but it is also possible to use a shared one. In this case $p_k = p, \forall k \in [1, K]$ and we simply denote it by p and not p_k. We examine such options in the experimental section and compare to hand-tuned and fixed parameter values.

Max pooling, in the case of MAC, retains one activation per 2D feature map. In this way, each descriptor component corresponds to an image patch equal to the receptive field. Then, pairwise image similarity is evaluated via descriptor inner product. Therefore, MAC similarity implicitly forms patch correspondences. The strength of each correspondence is given by the product of the associated descriptor components. In Figure 2 we show the image patches in correspondence that contribute most to the similarity. Such implicit correspondences are improved after fine-tuning. Moreover, the CNN fires less on ImageNet classes, *e.g.* cars and bicycles.

In Figure 4 we show how the spatial distribution of the activations is affected by the generalized mean. The larger the p the more localized the feature map responses are. Finally, in Figure 3 we present an example of a query and a database image matched with the fine-tuned VGG with GeM pooling layer (GeM layer in short). We show the feature maps that contribute the most into making this database image being distinguished from non-matching ones that have large similarity, too.

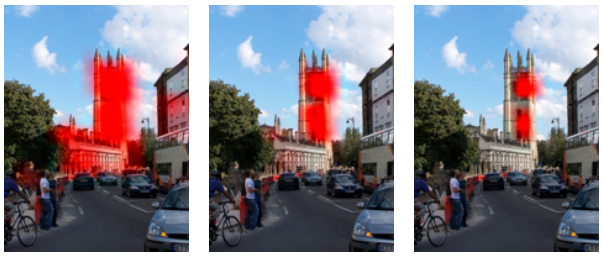


Fig. 4. Visualization of $\mathcal{X}^p_k$ projected on the original image for three different values of p. Case $p = 1$ corresponds to SPoC, and larger p corresponds to GeM before the summation of (3). Examples shown use the off-the-shelf VGG.

The last network layer comprises an ℓ_2-normalization layer. Vector $\mathbf{f}$ is ℓ_2-normalized so that similarity between two images is finally evaluated with inner product. In the rest of the paper, GeM vector corresponds to the ℓ_2-normalized vector $\bar{\mathbf{f}}$ and constitutes the image descriptor.

3.3 Siamese learning and loss function

We adopt a siamese architecture and train a two-branch network. Each branch is a clone of the other, meaning that they share the same parameters. The training input consists of image pairs (i, j) and labels $Y(i, j) \in \{0, 1\}$ declaring whether a pair is non-matching (label 0) or matching (label 1). We employ the contrastive loss [39] that acts on matching and non-matching pairs and is defined as

$$\mathcal{L}(i,j) = \begin{cases} \frac{1}{2}||\bar{\mathbf{f}}(i) - \bar{\mathbf{f}}(j)||^2, & \text{if } Y(i,j) = 1 \\ \frac{1}{2}\left(\max\{0, \tau - ||\bar{\mathbf{f}}(i) - \bar{\mathbf{f}}(j)||\}\right)^2, & \text{if } Y(i,j) = 0 \end{cases} \quad (6)$$

where $\bar{\mathbf{f}}(i)$ is the ℓ_2-normalized GeM vector of image i, and τ is a margin parameter defining when non-matching pairs have large enough distance in order to be ignored by the loss. We train the network using a large number of training pairs created automatically (see Section 4). In contrast to other methods [16], [42], [43], [44], we find that the contrastive loss generalizes better and converges at higher performance than the triplet loss.

3.4 Whitening and dimensionality reduction

In this section, the post-processing of fine-tuned GeM vectors is considered. Previous methods [10], [12] use PCA of an independent set for whitening and dimensionality reduction, *i.e.* the covariance matrix of all descriptors is analyzed. We propose to leverage the labeled data provided by the 3D models and use linear discriminant projections originally proposed by Mikolajczyk and Matas [53]. The projection is decomposed into two parts: whitening and rotation. The whitening part is the inverse of the square-root of the intraclass (matching pairs) covariance matrix $C_S^{-\frac{1}{2}}$, where

$$C_S = \sum_{Y(i,j)=1} \left(\bar{\mathbf{f}}(i) - \bar{\mathbf{f}}(j)\right)\left(\bar{\mathbf{f}}(i) - \bar{\mathbf{f}}(j)\right)^\top. \quad (7)$$

The rotation part is the PCA of the interclass (non-matching pairs) covariance matrix in the whitened space $\text{eig}(C_S^{-\frac{1}{2}} C_D C_S^{-\frac{1}{2}})$, where

$$C_D = \sum_{Y(i,j)=0} \left(\bar{\mathbf{f}}(i) - \bar{\mathbf{f}}(j)\right)\left(\bar{\mathbf{f}}(i) - \bar{\mathbf{f}}(j)\right)^\top. \quad (8)$$

The projection $P = C_S^{-\frac{1}{2}} \text{eig}(C_S^{-\frac{1}{2}} C_D C_S^{-\frac{1}{2}})$ is then applied as $P^\top(\bar{\mathbf{f}}(i) - \mu)$, where μ is the mean GeM vector to perform centering. To reduce the descriptor dimensionality to D dimensions, only eigenvectors corresponding to D largest eigenvalues are used. Projected vectors are subsequently ℓ_2-normalized.

Our approach uses all available training pairs efficiently in the optimization of the whitening. It is not optimized in an end-to-end manner and it is performed without using batches of training data. We first optimize the GeM descriptor and then optimize the whitening.

The described approach acts as a post-processing step, equivalently, once the fine-tuning of the CNN is finished. We additionally compare with the end-to-end learning of whitening. Whitening consists of vector shifting and projection which is modeled in a straightforward manner by a fully connected layer[1]. The results favor our approach and are discussed in the experimental section.

3.5 Image representation and search

Once the training is finished, an image is fed to the network shown in Figure 1. The extracted GeM descriptor is whitened and re-normalized. This constitutes the global descriptor for an image at a single scale. Scale invariance is learned to some extent by the training samples; however, additional invariance is added by multi-scale processing during test time without any additional learning. We follow a standard approach [27] and feed the image to the network at multiple scales. The resulting descriptors are finally pooled and re-normalized. This vector constitutes a multi-scale global image representation. We adopt GeM pooling for this state too, which is shown, in our experiments, consistently superior to the standard average pooling.

Image retrieval is simply performed by exhaustive Euclidean search over database descriptors *w.r.t.* the query descriptor. This is equivalent to the inner product evaluation of ℓ_2 normalized vectors, *i.e.* vector-to-matrix multiplication, and sorting. CNN-based descriptors are shown to be highly compatible with approximate-nearest neighbor search methods, in fact, they are very compressible [27]. In order to directly evaluate the effectiveness of the learned representation, we do not consider this alternative in this work. In practice, each descriptor requires 4 bytes per dimension to be stored.

It has recently become a standard policy to combine CNN global image descriptors with simple average query expansion (AQE) [10], [11], [12], [27]. An initial query is issued by Euclidean search and AQE acts on the top-ranked nQE images by average pooling of their descriptors. Herein, we argue that tuning nQE to work well across different datasets is not easy. AQE corresponds to a weighted average where nQE descriptors have unit weight and all the rest zero. We generalize this scheme and we propose performing weighted averaging, where the weight of the i-th ranked image is given by $(\bar{\mathbf{f}}(q)^\top \bar{\mathbf{f}}(i))^\alpha$. The similarity of each retrieved image matters. We show in our experiments that AQE is difficult to tune for datasets of different statistics, while this is not the case with the proposed approach. We refer to this approach as α-weighted query expansion (αQE). The proposed αQE reduces to AQE for $\alpha = 0$.

4 Training Dataset

In this section we briefly summarize the tightly-coupled Bag-of-Words (BoW) image-retrieval and Structure-from-Motion (SfM) 3D reconstruction system [17], [56] that is employed to automatically select our training data. Then, we describe how we use the 3D information to select harder matching pairs and hard non-matching pairs with larger variability.

4.1 BoW and 3D reconstruction

The retrieval engine used in the work of Schonberger *et al.* [17] builds upon BoW with fast spatial verification [33]. It uses Hessian affine local features [57], RootSIFT descriptors [58], and a fine vocabulary of 16M visual words [59]. Then, query images are chosen via min-hash and spatial verification, as in [18]. Image retrieval based on BoW is used to collect images of the objects/landmarks. These images serve as the initial matching graph for the succeeding SfM reconstruction, which is performed using the state-of-the-art SfM pipeline [60], [61], [62]. Different mining techniques, *e.g.* zoom in, zoom out [63], [64], sideways crawl [17], help to build a larger and more complete model.

In this work, we exploit the outcome of such a system. Given a large unannotated image collection, images are clustered and a 3D model is constructed per cluster. We use the terms *3D model*, *model* and *cluster* interchangeably. For each image, the estimated camera position is known, as well as the local features registered on the 3D model. We drop redundant (overlapping) 3D models, that might have been constructed from different seeds. Models reconstructing the same landmark but from different and disjoint viewpoints are considered as non-overlapping.

1. The bias is equal to the projected shifting vector.

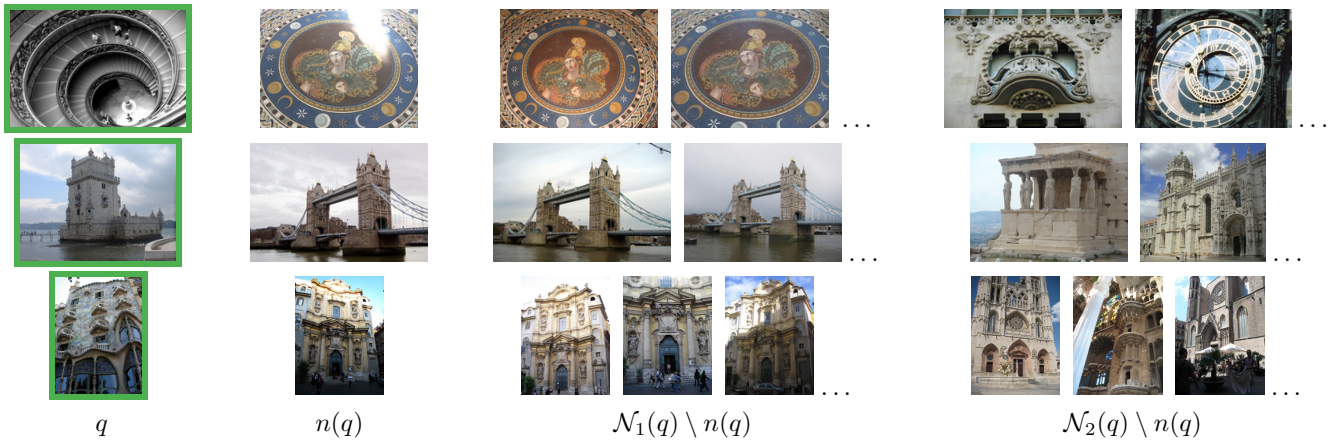


Fig. 5. Examples of training query image q (one per row shown in green border), and their corresponding negatives chosen by different strategies. We show the hardest non-matching image $n(q)$, and the additional non-matching images selected as negative examples by $\mathcal{N}_1(q)$ and our method $\mathcal{N}_2(q)$. The former chooses k-nearest neighbors among all non-matching images, while the latter chooses k-nearest neighbors but with at most one image per 3D model.

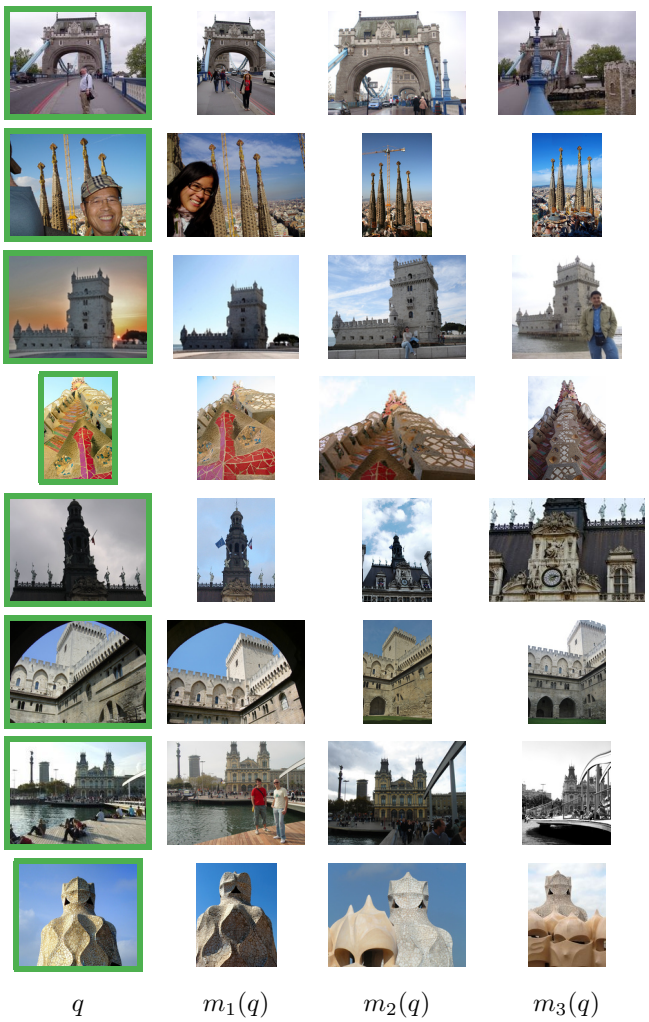


Fig. 6. Examples of training query images (green border) and matching images selected as positive examples by methods: $m_1(q)$ – the most similar image based on the current network; $m_2(q)$ – the most similar image based on the BoW representation; and our proposed $m_3(q)$ – a hard image depicting the same object.

4.2 Selection of training image pairs

A 3D model is described as a bipartite visibility graph $\mathbb{G} = (\mathcal{I} \cup \mathcal{P}, \mathcal{E})$ [65], where images $\mathcal{I}$ and points $\mathcal{P}$ are the vertices of the graph. The edges of this graph are defined by visibility relations between cameras and points, *i.e.* if a point $p \in \mathcal{P}$ is visible in an image $i \in \mathcal{I}$, then there exists an edge $(i, p) \in \mathcal{E}$. The set of points observed by an image i is given by

$$\mathcal{P}(i) = \{p \in \mathcal{P} : (i, p) \in \mathcal{E}\}. \tag{9}$$

We create a dataset of tuples $(q, m(q), \mathcal{N}(q))$, where q represents a query image, $m(q)$ is a positive image that matches the query, and $\mathcal{N}(q)$ is a set of negative images that do not match the query. These tuples are used to form training image pairs, where each tuple corresponds to $|\mathcal{N}(q)| + 1$ pairs. For a query image q, a pool $\mathcal{M}(q)$ of candidate positive images is constructed based on the camera positions in the cluster of q. It consists of the k images with camera centers closest to the query. Due to the wide range of camera orientations, these do not necessarily depict the same object. We therefore compare three different ways to select the positive image. The positive examples are fixed during the whole training process for all three strategies.

Positive images: CNN descriptor distance. The image that has the lowest descriptor distance to the query is chosen as positive, formally

$$m_1(q) = \underset{i \in \mathcal{M}(q)}{\operatorname{argmin}} ||\bar{\mathbf{f}}(q) - \bar{\mathbf{f}}(i)||. \tag{10}$$

This strategy is similar to the one followed by Arandjelovic *et al.* [16]. They adopt this choice since only GPS coordinates are available and not camera orientations. As a consequence, the chosen matching images already have small descriptor distance and, therefore, small loss too. The network is thus not forced to drastically change/learn by the matching examples, which is the drawback of this approach.

Positive images: maximum inliers. In this approach, the 3D information is exploited to choose the positive image, independently of the CNN descriptor. In particular, the image that has the highest number of co-observed 3D points with the query is chosen. That is,

$$m_2(q) = \underset{i \in \mathcal{M}(q)}{\operatorname{argmax}} |\mathcal{P}(q) \cap \mathcal{P}(i)|. \tag{11}$$

This measure corresponds to the number of spatially verified features between two images, a measure commonly used for ranking in BoW-based retrieval. As this choice is independent of the CNN representation, it delivers more challenging positive examples.

Positive images: relaxed inliers. Even though both previous methods choose positive images depicting the same object as the query, the variance of viewpoints is limited. Instead of using a pool of images with similar camera position, the positive example is selected at random from a set of images that co-observe enough points with the query, but do not exhibit too extreme of a scale change. The positive example in this case is

$$m_3(q) = \texttt{rnd}\left\{ i \in \mathcal{M}(q) : \frac{|\mathcal{P}(i) \cap \mathcal{P}(q)|}{|\mathcal{P}(q)|} \geq t_i,\ \texttt{scale}(i,q) \leq t_s \right\}, \tag{12}$$

where $\texttt{scale}(i,q)$ is the scale change between the two images. This method results in selecting harder matching examples that are still guaranteed to depict the same object. Method m_3 chooses different image than m_1 on 86.5% of the queries. In Figure 6 we present examples of query images and the corresponding positives selected with the three different methods. The relaxed method increases the variability of viewpoints.

Negative images. Negative examples are selected from clusters different than the cluster of the query image, as the clusters are non-overlaping. We choose hard negatives [6], [46], that is, non-matching images with the most similar descriptor. Two different strategies are proposed: In the first, $\mathcal{N}_1(q)$, k-nearest neighbors from all non-matching images are selected. In the second, $\mathcal{N}_2(q)$, the same criterion is used, but at most one image per cluster is allowed. While $\mathcal{N}_1(q)$ often leads to multiple, and very similar, instances of the same object, $\mathcal{N}_2(q)$ provides higher variability of the negative examples, see Figure 5. While positives examples are fixed during the whole training process, hard negatives depend on the current CNN parameters and are re-mined multiple times per epoch.

5 Experiments

In this section we discuss implementation details of our training, evaluate different components of our method, and compare to the state of the art.

5.1 Training setup and implementation details

Structure-from-Motion (SfM). Our training samples are derived from the dataset used in the work of Schonberger *et al.* [17], which consists of 7.4 million images downloaded from Flickr using keywords of popular landmarks, cities and countries across the world. The clustering procedure [18] gives around 20k images to serve as query seeds. The extensive retrieval-SfM reconstruction [56] of the whole dataset results in $1,474$ reconstructed 3D models. Removing overlapping models leaves us with 713 3D models containing more than 163k unique images from the initial dataset. The initial dataset contains, on purpose, all images of Oxford5k and Paris6k datasets. In this way, we are able to exclude 98 clusters that contain any image (or their near duplicates) from these test datasets.

Training pairs. The size of the 3D models varies from 25 to 11k images. We randomly select 551 models (around 133k images) for training and 162 (around $30k$ images) for validation. The number of training queries per 3D model is 10% of its size and limited to be less or equal to 30. Around $6,000$ and $1,700$ images are selected for training and validation queries per epoch, respectively.

Each training and validation tuple contains 1 query, 1 positive and 5 negative images. The pool of candidate positives consists of $k = 100$ images with the closest camera centers to the query. In particular, for method m_3, the inlier-overlap threshold is $t_i = 0.2$, and the scale-change threshold $t_s = 1.5$. Hard negatives are re-mined 3 times per epoch, *i.e.* roughly every $2,000$ training queries. Given the chosen queries and the chosen positives, we further add 20 images per model to serve as candidate negatives during re-mining. This constitutes a training set of around 22k images per epoch when all the training 3D models are used. The query-tuple selection process is repeated every epoch. This slightly improves the results.

Learning configuration. We use MatConvNet [66] for the fine-tuning of networks. To perform the fine-tuning as described in Section 3, we initialize by the convolutional layers of AlexNet [1], VGG16 [23], or ResNet101 [24]. AlexNet is trained using stochastic gradient descent (SGD), while training of VGG and ResNet is more stable with Adam [67]. We use initial learning rate equal to $l_0 = 10^{-3}$ for SGD, initial stepsize equal to $l_0 = 10^{-6}$ for Adam, an exponential decay $l_0 \exp(-0.1i)$ over epoch i, momentum 0.9, weight decay 5×10^{-4}, margin τ for contrastive loss 0.7 for AlexNet, 0.75 for VGG, and 0.85 for ResNet, justified by the increase in the dimensionality of the embedding, and a batch size of 5 training tuples. All training images are resized to a maximum size of 362×362, while keeping the original aspect ratio. Training is done for at most 30 epochs and the best network is selected based on performance, measured via mean Average Precision (mAP) [33], on validation tuples. Fine-tuning of VGG for one epoch takes around 2 hours on a single TITAN X (Maxwell) GPU with 12 GB of memory.

We overcome GPU memory limitations by associating each query to a tuple, *i.e.*, query plus 6 images (5 positive and 1 negative). Moreover, the whole tuple is processed in the same batch. Therefore, we feed 7 images to the network, which represents 6 pairs. In a naive approach, when the query image is different for each pair, 6 pairs require 12 images.

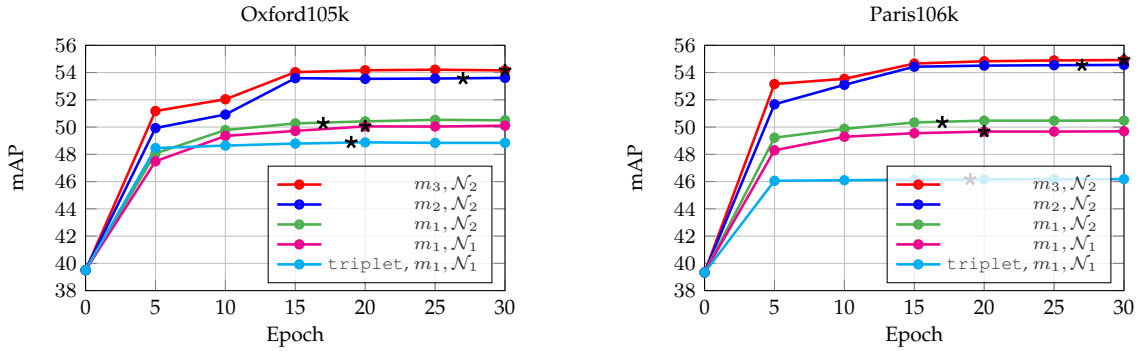


Fig. 7. Performance comparison of methods for positive and negative example selection. Evaluation is performed with AlexNet MAC on Oxford105k and Paris106k datasets. The plot shows the evolution of mAP with the number of training epochs. Epoch 0 corresponds to the off-the-shelf network. All approaches use the contrastive loss, except if otherwise stated. The network with the best performance on the validation set is marked with $\star$.

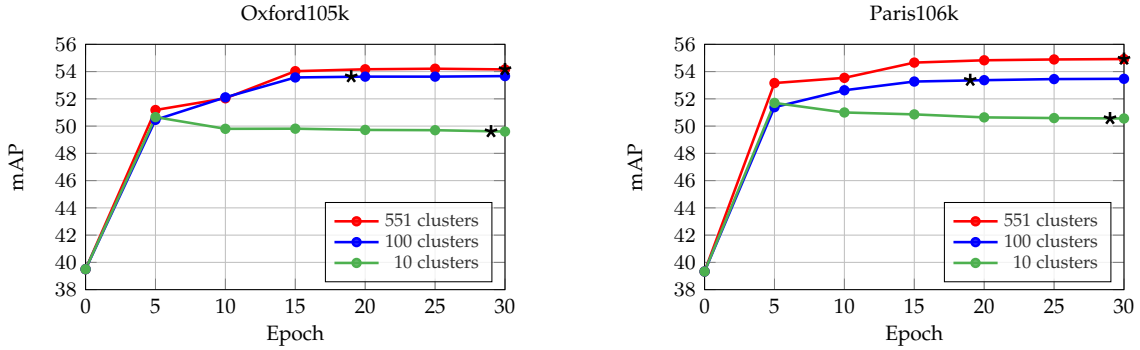


Fig. 8. Influence of the number of 3D models used for CNN fine-tuning. Performance is evaluated with AlexNet MAC on Oxford105k and Paris106k datasets using 10, 100 and 551 (all available) 3D models. The network with the best performance on the validation set is marked with $\star$.

5.2 Test datasets and evaluation protocol

Test datasets. We evaluate our approach on Oxford buildings [33], Paris [68] and Holidays[2] [69] datasets. The first two are closer to our training data, while the last is differentiated by containing similar scenes and not only man-made objects or buildings. These are also combined with 100k distractors from Oxford100k to allow for evaluation at larger scale. The performance is measured via mAP. We follow the standard evaluation protocol for Oxford and Paris and crop the query images with the provided bounding box. The cropped image is fed as input to the CNN.

Single-scale evaluation. The dimensionality of the images fed into the CNN is limited to 1024×1024 pixels. In our experiments, no vector post-processing is applied if not otherwise stated.

Multi-scale evaluation. Multi-scale representation is only used during test time. We resize the input image to different sizes, then feed multiple input images to the network, and finally combine the global descriptors from multiple scales into a single descriptor. We compare the baseline average pooling [27] with our generalized mean whose pooling parameter is equal to the value learned in the global pooling layer of the network. In this case, the whitening is learned on the final multi-scale image descriptors. In our experiments, a single-scale evaluation is used if not otherwise stated.

2. We use the up-right version of Holidays dataset where images are manually rotated so that depicted objects are up-right. This makes us directly comparable to [27]. A different version of up-right Holidays is used in our earlier work [25], where EXIF metadata is used to rotate the images.

5.3 Results on image retrieval

Learning. We evaluate the off-the-shelf CNN and our fine-tuned ones after different number of training epochs. The different methods for positive and negative selection are evaluated independently in order to isolate the benefit of each one. Finally, we also perform a comparison with the triplet loss [16], trained on the same training data as the contrastive loss. Note that a triplet forms two pairs. Results are presented in Figure 7. The results show that positive examples with larger viewpoint variability and negative examples with higher content variability acquire a consistent increase in the performance. The triplet loss[3] appears to be inferior in our context; we observe oscillation of the error in the validation set from early epochs, which implies over-fitting. In the rest of the paper, we adopt the $m_3, \mathcal{N}_2$ approach.

Dataset variability. We perform fine-tuning by using a subset of the available 3D models. Results are presented in Figure 8 with 10, 100 and 551 (all available) clusters, while keeping the amount of training data, *i.e.* number of training queries, fixed. In the case of 10 and 100 models, we use the largest ones. It is better to train with all 3D models due to the resulting higher variability in the training set. Remarkably, significant increase in performance is achieved even with 10 or 100 models. However, the network is able to over-fit in the case of few clusters. In the rest of our experiments we use all 551 3D models for training.

3. The margin parameter for the triplet loss is set equal to 0.1 [16].

TABLE 1
Performance (mAP) comparison after CNN fine-tuning for different pooling layers. GeM is evaluated with a single shared pooling parameter or multiple pooling parameters (one for each feature map), which are either fixed or learned. A single value or a range is reported in the case of a single or multiple parameters, respectively. Results reported with AlexNet and with the use of L_w. The best performance highlighted in **bold**.

Pooling	Initial p	Learned p	Oxford5k	Oxford105k	Paris6k	Paris106k	Holidays	Hol101k
MAC	inf	–	62.2	52.8	68.9	54.7	78.4	66.0
SPoC	1	–	61.2	54.9	70.8	58.0	79.9	70.6
GeM	3	–	**67.9**	60.2	74.8	61.7	83.2	73.3
	[2, 5]	–	66.8	59.7	74.1	60.8	**84.0**	73.6
	[2, 10]	–	65.6	57.8	72.2	58.9	81.9	71.9
	3	2.32	67.7	**60.6**	**75.5**	**62.6**	83.7	**73.7**
	3	[1.0, 6.5]	66.3	57.8	74.0	60.5	83.2	72.7
	[2, 10]	[1.6, 9.9]	65.3	56.4	71.4	58.6	81.4	70.8

TABLE 2
Performance (mAP) comparison of CNN vector post-processing: no post-processing, PCA-whitening [22] (PCA_w) and our learned whitening (L_w). No dimensionality reduction is performed. Fine-tuned AlexNet (Alex) produces a 256D vector and fine-tuned VGG a 512D vector. The best performance highlighted in **bold**, the worst in **blue**. The proposed method consistently performs either the best (22 out of 24 cases) or on par with the best method. On the contrary, PCA_w [22] often hurts the performance significantly. Best viewed in color.

Net	Post	Dim	Oxford5k		Oxford105k		Paris6k		Paris106k		Holidays		Hol101k	
			MAC	GeM	MAC	GeM	MAC	GeM	MAC	GeM	MAC	GeM	MAC	GeM
Alex	–	256	60.2	**60.1**	**54.2**	54.1	67.5	**68.6**	**54.9**	**56.9**	**74.5**	**78.7**	64.8	**70.9**
	PCA_w		**56.9**	63.7	**44.1**	**53.7**	**64.3**	73.2	**46.8**	57.4	75.4	82.5	**63.1**	71.8
	L_w		**62.2**	**67.7**	52.8	**60.6**	**68.9**	**75.5**	54.7	**62.6**	**78.4**	**83.7**	**66.0**	**73.7**
VGG	–	512	82.0	**82.0**	76.0	**76.9**	**78.3**	**79.7**	71.2	**72.6**	**79.9**	**83.1**	**69.4**	**74.5**
	PCA_w		**78.4**	83.1	**71.3**	77.7	80.6	84.5	**70.9**	76.9	82.2	86.6	70.0	75.9
	L_w		**82.3**	**85.9**	**77.0**	**81.7**	**83.8**	**86.0**	**76.2**	**79.6**	**84.1**	**87.3**	**71.9**	**77.1**

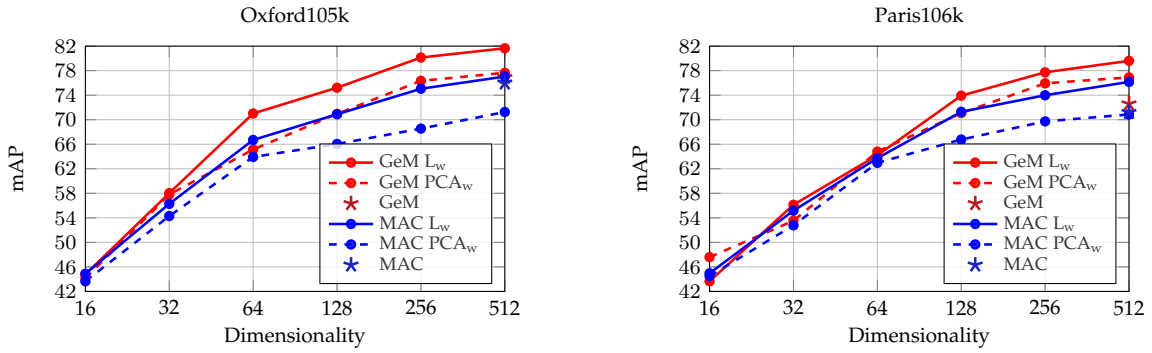


Fig. 9. Performance comparison of the dimensionality reduction performed by PCA_w and our L_w with the fine-tuned VGG with MAC layer and the fine-tuned VGG with GeM layer on Oxford105k and Paris106k datasets.

Pooling methods. We evaluate the effect of different pooling layers during CNN fine-tuning. We present the results in Table 1. GeM layer consistently outperforms the conventional max and average pooling. This holds for each of the following cases, (i) a single shared pooling parameter p is used, (ii) each feature map has different p_k and (iii) the pooling parameter(s) is (are) either fixed or learned. Learning a shared parameter turns out to be better than learning multiple ones, as the latter makes the cost function more complex. Additionally, the initial values seem to matter to some extent, with a preference for intermediate values. Finally, a shared fixed parameter and a shared learned parameter perform similarly, with the latter being slightly better. This is the case which we adopt for the rest of our experiments, *i.e.* a single shared parameter p that is learned.

Learned projections. The PCA-whitening [22] (PCA_w) is shown to be essential in some cases of CNN-based descriptors [10], [12], [15]. On the other hand, it is shown that on some datasets, the performance after PCA_w substantially drops compared to the raw descriptors (max pooling on Oxford5k [10]). We perform comparison of the traditional whitening methods and the proposed learned discriminative whitening (L_w), described in Section 3.4. Table 2 shows results without post-processing, with PCA_w and with L_w.

TABLE 3
Performance (mAP) evaluation of the multi-scale representation using the fine-tuned VGG with GeM layer. The original scale and down-sampled versions of it are jointly represented. The pooling parameter used by the generalized mean is the same as the one learned in the GeM layer of the network and equal to 2.92. The results reported include the use of L_w.

Pooling over scales	Scale $1/1$	$1/\sqrt{2}$	$1/2$	$1/\sqrt{8}$	$1/4$	Oxford5k	Oxford105k	Paris6k	Paris106k	Holidays	Hol101k
–	■					85.9	81.7	86.0	79.6	87.3	77.1
Average	■	■				86.8	82.6	86.7	80.2	88.1	79.3
	■	■	■			87.2	82.4	87.3	80.6	89.1	79.6
	■	■	■	■		86.6	81.9	88.2	81.3	89.9	79.9
	■	■	■	■	■	85.1	80.1	88.8	81.6	90.6	80.5
Generalized mean	■	■				87.3	83.1	86.9	80.5	88.1	79.5
	■	■	■			87.9	83.3	87.7	81.3	89.5	79.9
	■	■	■	■		87.7	83.2	88.7	82.3	89.9	80.2
	■	■	■	■	■	86.8	82.4	89.4	82.7	91.1	81.4

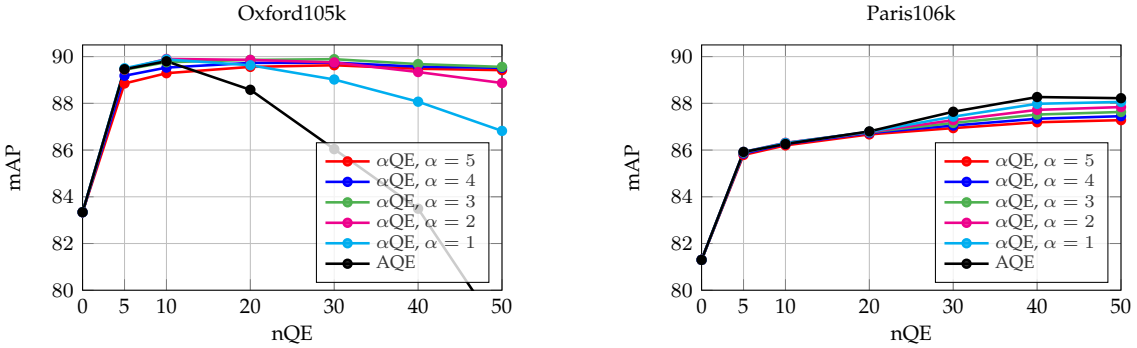


Fig. 10. Performance evaluation of our α-weighted query expansion (αQE) with the VGG with GeM layer, multi-scale representation, and L_w on Oxford105k and Paris106k datasets. We compare the standard average query expansion (AQE) to our αQE for different values of α and number of images used nQE.

TABLE 4
Performance (mAP) evaluation for varying descriptor dimensionality after reduction with L_w. Results reported with the fine-tuned VGG with GeM and the fine-tuned ResNet (Res) with GeM. Multi-scale representation is used at the test time for both networks.

Net	Dim	Oxf5k	Oxf105k	Par6k	Par106k	Hol	Hol101k
VGG	512	87.9	83.3	87.7	81.3	89.5	79.9
	256	85.4	79.7	85.7	78.2	87.8	77.2
	128	81.6	75.4	83.4	74.9	84.4	72.6
	64	77.0	69.9	77.4	66.7	81.1	66.2
	32	66.9	57.4	72.2	58.6	72.9	54.3
	16	56.2	44.4	63.5	45.5	60.9	36.9
	8	34.1	25.7	43.9	29.0	43.4	13.8
Res	2048	87.8	84.6	92.7	86.9	93.9	87.9
	1024	86.2	82.4	91.8	85.3	92.5	86.1
	512	84.6	80.4	90.0	82.6	90.6	83.2
	256	83.1	77.3	87.5	78.8	88.4	80.2
	128	79.5	72.2	84.5	74.3	85.9	76.5
	64	74.0	65.8	78.4	65.3	80.3	66.9
	32	57.9	48.5	70.8	56.1	71.2	51.9
	16	40.3	31.8	61.8	45.6	56.4	31.3
	8	25.3	16.3	44.3	27.8	37.8	11.4

Our experiments confirm that PCA_w often reduces the performance. In contrast to that, the proposed L_w achieves the best performance in most cases and is never the worst-performing method. Compared with the no post-processing baseline, L_w reduces the performance twice for AlexNet, but the drop is negligible compared to the drop observed for PCA_w. For VGG, the proposed L_w *always* outperforms the no post-processing baseline.

We conduct an additional experiment by appending a whitening layer at the end of the network during fine-tuning. In this way, whitening is learned in an end-to-end manner, along with the convolutional filters and with the same training data in batch-mode. Dropout [70] is additionally used for this layer which we find to be essential. We observe that convergence of the network comes much slower in this case, *i.e.* after 60 epochs. Moreover, the final achieved performance is not higher than our L_w. In particular, end-to-end whitening on AlexNet MAC achieves 49.6 and 52.1 mAP on Oxford105k and Paris106k, respectively, while our L_w on the same network achieves 52.8 and 54.7 mAP on Oxford105k and Paris106k, respectively. Therefore, we adopt L_w as it is much faster to train and more effective.

Dimensionality reduction. We compare dimensionality reduction performed with PCA_w [22] and with our L_w. The performance for varying descriptor dimensionality is plotted in Figure 9. The plots suggest that L_w works better in most dimensionalities.

TABLE 5
Performance (mAP) comparison with the state-of-the-art image retrieval using VGG and ResNet (Res) deep networks, and using local features.
F-tuned: Use of the fine-tuned network (yes), or the off-the-shelf network (no), not applicable for the methods using local features (n/a).
Dim: Dimensionality of the final compact image representation, not applicable (n/a) for the BoW based methods due to their sparse representation.
Our methods are marked with ⋆ and they are always accompanied by the multi-scale representation and our learned whitening L_w.
Previous state of the art is highlighted in **bold**, new state of the art in **red outline**. Best viewed in color.

Net	Method	F-tuned	Dim	Oxford5k	Oxford105k	Paris6k	Paris106k	Holidays	Hol101k
Compact representation using deep networks									
VGG	MAC [9]†	no	512	56.4	47.8	72.3	58.0	79.0	66.1
	SPoC [10]†	no	512	68.1	61.1	78.2	68.4	83.9	**75.1**
	CroW [11]	no	512	70.8	65.3	79.7	72.2	85.1	–
	R-MAC [12]	no	512	66.9	61.6	83.0	75.7	86.9‡	–
	BoW-CNN [48]	no	n/a	73.9	59.3	82.0	64.8	–	–
	NetVLAD [16]	no	4096	66.6	–	77.4	–	88.3	–
	NetVLAD [16]	yes	512	67.6	–	74.9	–	86.1	–
	NetVLAD [16]	yes	4096	71.6	–	79.7	–	87.5	–
	Fisher Vector [49]	yes	512	81.5	76.6	82.4	–	–	–
	R-MAC [26]	yes	512	**83.1**	**78.6**	**87.1**	**79.7**	**89.1**	–
	⋆ GeM	yes	512	**87.9**	**83.3**	**87.7**	**81.3**	**89.5**	**79.9**
Res	R-MAC [12]‡	no	2048	69.4	63.7	85.2	77.8	91.3	–
	R-MAC [27]	yes	2048	**86.1**	**82.8**	**94.5**	**90.6**	**94.8**	–
	⋆ GeM	yes	2048	**87.8**	**84.6**	92.7	86.9	93.9	**87.9**
Re-ranking (R) and query expansion (QE)									
n/a	BoW+R+QE [36]	n/a	n/a	82.7	76.7	80.5	71.0	–	–
	BoW-fVocab+R+QE [59]	n/a	n/a	84.9	79.5	82.4	77.3	75.8	–
	HQE [38]	n/a	n/a	88.0	84.0	82.8	–	–	–
VGG	CroW+QE [11]	no	512	74.9	70.6	84.8	79.4	–	–
	R-MAC+R+QE [12]	no	512	77.3	73.2	86.5	79.8	–	–
	BoW-CNN+R+QE [48]	no	n/a	78.8	65.1	84.8	64.1	–	–
	R-MAC+QE [26]	yes	512	**89.1**	**87.3**	**91.2**	**86.8**	–	–
	⋆ GeM+αQE	yes	512	**91.9**	**89.6**	**91.9**	**87.6**	–	–
Res	R-MAC+QE [12]‡	no	2048	78.9	75.5	89.7	85.3	–	–
	R-MAC+QE [27]	yes	2048	**90.6**	**89.4**	**96.0**	**93.2**	–	–
	⋆ GeM+αQE	yes	2048	**91.0**	**89.5**	95.5	91.9	–	–

†: Our evaluation of MAC and SPoC with PCA_w and with the off-the-shelf network.
‡: Evaluation of R-MAC by [27] with the off-the-shelf network.

Multi-scale representation. We evaluate multi-scale representation constructed at test time without any additional learning. We compare the previously used averaging of descriptors at multiple image scales [27] with our generalized-mean of the same descriptors. Results are presented in Table 3, where there is a significant benefit when using the multi-scale GeM. It also offers some improvement over average pooling. In the rest of our experiments we adopt multi-scale representation, pooled by generalized mean, for scales 1, $1/\sqrt{2}$, and $1/2$. Results using the supervised dimensionality reduction by L_w on the multi-scale GeM representation are shown in Table 4.

Query expansion. We evaluate the proposed αQE, which reduces to AQE for $\alpha = 0$, and present results in Figure 10. Note that Oxford and Paris have different statistics in terms of the number of relevant images per query. The average, minimum, and maximum number of positive images per query on Oxford is 52, 6, and 221, respectively. The same measurements for Paris are 163, 51, and 289. As a consequence, AQE behaves in a very different way across these dataset, while our αQE is a more stable choice. We finally set $\alpha = 3$ and $nQE = 50$.

Over-fitting and generalization. In all experiments, all clusters including any image (not only query landmarks) from Oxford5k or Paris6k datasets are removed. We now repeat the training using all 3D models, including those of Oxford and Paris landmarks. In this way, we evaluate whether the network tends to over-fit to the training data or to generalize. The same amount of training queries is used for a fair comparison. We observe negligible difference in the performance of the network on Oxford and Paris evaluation results, *i.e.* the difference in mAP was on average $+0.3$ over all testing datasets. We conclude that the network generalizes well and is relatively insensitive to over-fitting.

Comparison with the state of the art. We extensively compare our results with the state-of-the-art performance on compact image representations and on approaches that do query expansion. The results for the fine-tuned GeM based networks are summarized together with previously published results in Table 5. The proposed methods outperform the state of the art on all datasets when the VGG network architecture and initialization are used. Our method is outperformed by the work of Gordo *et al.* on Paris with the ResNet architecture, while we have the state-of-the-art score on Oxford. We are on par with the state-of-the-

art on Holidays. Note, however, that we did not perform any manual labeling or cleaning of our training data, while in their work landmark labels were used. We additionally combine GeM with query expansion and further boost the performance.

6 Conclusions

We addressed fine-tuning of CNN for image retrieval. The training data are selected from an automated 3D reconstruction system applied on a large unordered photo collection. The reconstructions consist of buildings and popular landmarks; however, the same process is applicable to any rigid 3D objects. The proposed method does not require any manual annotation and yet achieves top performance on standard benchmarks. The achieved results reach the level of the best systems based on local features with spatial matching and query expansion while being faster and requiring less memory. The proposed pooling layer that generalizes previously adopted mechanisms is shown to improve the retrieval accuracy while it is also effective for constructing a joint multi-scale representation. Training data, trained models, and code are publicly available.

Acknowledgments

The authors were supported by the MSMT LL1303 ERC-CZ grant. We would also like to thank Karel Lenc for insightful discussions.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012. 1, 2, 4, 8

[2] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *IJCV*, 2015. 1, 3

[3] H. Azizpour, A. S. Razavian, J. Sullivan, A. Maki, and S. Carlsson, "From generic to specific deep representations for visual recognition," in *CVPRW*, 2015. 1

[4] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "DeCAF: A deep convolutional activation feature for generic visual recognition," in *ICML*, 2014. 1

[5] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: An astounding baseline for recognition," in *CVPRW*, 2014. 1, 3

[6] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *CVPR*, 2014. 1, 3, 8

[7] F. Iandola, M. Moskewicz, S. Karayev, R. Girshick, T. Darrell, and K. Keutzer, "DenseNet: Implementing efficient ConvNet descriptor pyramids," in *arXiv:1404.1869*, 2014. 1

[8] Y. Gong, L. Wang, R. Guo, and S. Lazebnik, "Multi-scale orderless pooling of deep convolutional activation features," in *ECCV*, 2014. 1, 3

[9] A. S. Razavian, J. Sullivan, S. Carlsson, and A. Maki, "Visual instance retrieval with deep convolutional networks," *ITE Trans. MTA*, 2016. 1, 3, 4, 12

[10] A. Babenko and V. Lempitsky, "Aggregating deep convolutional features for image retrieval," in *ICCV*, 2015. 1, 3, 4, 6, 10, 12

[11] Y. Kalantidis, C. Mellina, and S. Osindero, "Cross-dimensional weighting for aggregated deep convolutional features," in *ECCVW*, 2016. 1, 3, 6, 12

[12] G. Tolias, R. Sicre, and H. Jégou, "Particular object retrieval with integral max-pooling of CNN activations," in *ICLR*, 2016. 1, 3, 4, 6, 10, 12

[13] N. Zhang, J. Donahue, R. Girshick, and T. Darrell, "Part-based R-CNNs for fine-grained category detection," in *ECCV*, 2014. 1

[14] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *CVPR*, 2014. 1

[15] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky, "Neural codes for image retrieval," in *ECCV*, 2014. 1, 3, 10

[16] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: CNN architecture for weakly supervised place recognition," in *CVPR*, 2016. 1, 3, 5, 7, 9, 12

[17] J. L. Schönberger, F. Radenović, O. Chum, and J.-M. Frahm, "From single image query to detailed 3D reconstruction," in *CVPR*, 2015. 1, 6, 8

[18] O. Chum and J. Matas, "Large-scale discovery of spatially related images," *PAMI*, 2010. 1, 6, 8

[19] T. Weyand and B. Leibe, "Discovering details and scene structure with hierarchical iconoid shift," in *ICCV*, 2013. 1

[20] J. Philbin, J. Sivic, and A. Zisserman, "Geometric latent dirichlet allocation on a matching graph for large-scale image datasets," *IJCV*, 2011. 1

[21] A. Mousavian and J. Kosecka, "Deep convolutional features for image based retrieval and scene categorization," in *arXiv:1509.06033*, 2015. 1, 3

[22] H. Jégou and O. Chum, "Negative evidences and co-occurences in image retrieval: The benefit of PCA and whitening," in *ECCV*, 2012. 2, 3, 10, 11

[23] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *arXiv:1409.1556*, 2014. 2, 4, 8

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016. 2, 4, 8

[25] F. Radenović, G. Tolias, and O. Chum, "CNN image retrieval learns from BoW: Unsupervised fine-tuning with hard examples," in *ECCV*, 2016. 2, 9

[26] A. Gordo, J. Almazan, J. Revaud, and D. Larlus, "Deep image retrieval: Learning global representations for image search," in *ECCV*, 2016. 2, 3, 12

[27] A. Gordo, J. Almazan, J. Revaud, and D. Larlus, "End-to-end learning of deep visual representations for image retrieval," *IJCV*, 2017. 2, 6, 9, 12

[28] F. Perronnin, Y. Liu, J. Sanchez, and H. Poirier, "Large-scale image retrieval with compressed Fisher vectors," in *CVPR*, 2010. 2

[29] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid, "Aggregating local descriptors into compact codes," *PAMI*, 2012. 2

[30] F. Radenović, H. Jegou, and O. Chum, "Multiple measurements and joint dimensionality reduction for large scale image search with short vectors," in *ICMR*, 2015. 2

[31] R. Arandjelovic and A. Zisserman, "All about VLAD," in *CVPR*, 2013. 2

[32] G. Tolias, T. Furon, and H. Jégou, "Orientation covariant aggregation of local descriptors with embeddings," in *ECCV*, 2014. 2

[33] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," in *CVPR*, 2007. 2, 6, 8, 9

[34] Y. Avrithis and Y. Kalantidis, "Approximate Gaussian mixtures for large scale vocabularies," in *ECCV*, 2012. 2

[35] X. Shen, Z. Lin, J. Brandt, and Y. Wu, "Spatially-constrained similarity measure for large-scale object retrieval," *PAMI*, 2014. 2

[36] O. Chum, A. Mikulik, M. Perdoch, and J. Matas, "Total recall II: Query expansion revisited," in *CVPR*, 2011. 2, 12

[37] Q. Danfeng, S. Gammeter, L. Bossard, T. Quack, and L. V. Gool, "Hello neighbor: Accurate object retrieval with k-reciprocal nearest neighbors," in *CVPR*, 2011. 2

[38] G. Tolias and H. Jégou, "Visual query expansion with or without geometry: refining local descriptors by feature aggregation," *Pattern Recognition*, 2014. 2, 12

[39] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *CVPR*, 2005. 2, 5

[40] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *CVPR*, 2006. 2

[41] J. Hu, J. Lu, and Y.-P. Tan, "Discriminative deep metric learning for face verification in the wild," in *CVPR*, 2014. 2

[42] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu, "Learning fine-grained image similarity with deep ranking," in *CVPR*, 2014. 2, 5

[43] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *CVPR*, 2015. 2, 5

[44] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," in *ICLRW*, 2015. 2, 5

[45] L. Zheng, Y. Zhao, S. Wang, J. Wang, and Q. Tian, "Good practice in CNN feature transfer," in *arXiv:1604.00133*, 2016. 3

[46] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, and F. Moreno-Noguer, "Fracking deep convolutional image descriptors," in *arXiv:1412.6537*, 2014. 3, 8

[47] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han, "Large-scale image retrieval with attentive deep local features," in *ICCV*, 2017. 3

[48] E. Mohedano, K. McGuinness, N. E. O'Connor, A. Salvador, F. Marques, and X. Giro-i Nieto, "Bags of local convolutional features for scalable instance search," in *ICMR*, 2016. 3, 12

[49] E.-J. Ong, S. Husain, and M. Bober, "Siamese network of deep fisher-vector descriptors for image retrieval," in *arXiv:1702.00338*, 2017. 3, 12

[50] C.-Y. Lee, P. W. Gallagher, and Z. Tu, "Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree," in *AISTATS*, 2016. 3

[51] N. Cohen, O. Sharir, and A. Shashua, "Deep simnets," in *CVPR*, 2016. 3

[52] O. Morère, J. Lin, A. Veillard, L.-Y. Duan, V. Chandrasekhar, and T. Poggio, "Nested invariance pooling and rbm hashing for image instance retrieval," in *ICMR*, 2017. 3

[53] K. Mikolajczyk and J. Matas, "Improving descriptors for fast tree matching by optimal linear projection," in *ICCV*, 2007. 3, 6

[54] G. Papandreou, I. Kokkinos, and P.-A. Savalle, "Modeling local and global deformations in deep learning: Epitomic convolution, multiple instance learning, and sliding window detection," in *CVPR*, 2015. 4

[55] P. Dollár, Z. Tu, P. Perona, and S. Belongie, "Integral channel features." in *BMVC*, 2009. 4

[56] F. Radenović, J. L. Schönberger, D. Ji, J.-M. Frahm, O. Chum, and J. Matas, "From dusk till dawn: Modeling in the dark," in *CVPR*, 2016. 6, 8

[57] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. V. Gool, "A comparison of affine region detectors," *IJCV*, 2005. 6

[58] R. Arandjelovic and A. Zisserman, "Three things everyone should know to improve object retrieval," in *CVPR*, 2012. 6

[59] A. Mikulik, M. Perdoch, O. Chum, and J. Matas, "Learning vocabularies over a fine quantization," *IJCV*, 2013. 6, 12

[60] J.-M. Frahm, P. Georgel, D. Gallup, T. Johnson, R. Raguram, C. Wu, Y.-H. Jen, E. Dunn, B. Clipp, S. Lazebnik, and M. Pollefeys, "Building Rome on a cloudless day," in *ECCV*, 2010. 6

[61] S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. M. Seitz, and R. Szeliski, "Building Rome in a day," *Communications of the ACM*, 2011. 6

[62] J. L. Schönberger and J.-M. Frahm, "Structure-from-Motion revisited," in *CVPR*, 2016. 6

[63] A. Mikulik, O. Chum, and J. Matas, "Image retrieval for online browsing in large image collections," in *SISAP*, 2013. 6

[64] A. Mikulík, F. Radenović, O. Chum, and J. Matas, "Efficient image detail mining," in *ACCV*, 2014. 6

[65] Y. Li, N. Snavely, and D. P. Huttenlocher, "Location recognition using prioritized feature matching," in *ECCV*, 2010. 7

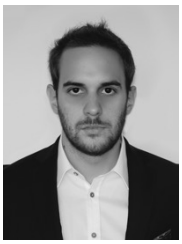
[66] A. Vedaldi and K. Lenc, "MatConvNet: Convolutional neural networks for matlab," in *ACM Multimedia*, 2015. 8

[67] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015. 8

[68] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Lost in quantization: Improving particular object retrieval in large scale image databases," in *CVPR*, 2008. 9

[69] H. Jégou, M. Douze, and C. Schmid, "Hamming embedding and weak geometric consistency for large scale image search," in *ECCV*, 2008. 9

[70] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *JMLR*, 2014. 11



Filip Radenović obtained his Master's degree from the Faculty of Electrical Engineering, University of Montenegro in 2013. Currently, he is a PhD candidate at the Visual Recognition Group, which is a part of the Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague. His research interests are mainly large-scale image retrieval problems.

Giorgos Tolias obtained his PhD from NTU of Athens and then moved to Inria Rennes as a post-doctoral researcher. Currently, he is a post-doctoral researcher at the Visual Recognition Group of CTU in Prague. He enjoys working on large-scale visual recognition problems.

Ondřej Chum is leading a team within the Visual Recognition Group at the Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague. He received the MSc degree in computer science from Charles University, Prague, in 2001 and the PhD degree from the Czech Technical University in Prague, in 2005. From 2006 to 2007, he was a postdoctoral researcher at the Visual Geometry Group, University of Oxford, United Kingdom. His research interests include large-scale image and particular object retrieval, object recognition, and robust estimation of geometric models. He is a member of Image and Vision Computing editorial board, and he has served in various roles at major international conferences. He co-organizes Computer Vision and Sports Summers School in Prague. He was the recipient of the Best Paper Prize at the British Machine Vision Conference in 2002. He was awarded the 2012 Outstanding Young Researcher in Image and Vision Computing runner up for researchers within seven years of their PhD. In 2017, he was the recipient of the Longuet-Higgins Prize.

III Orientation covariant aggregation of local descriptors with embeddings

Title: Orientation covariant aggregation of local descriptors with embeddings
Authors: G. Tolias, T. Furon, H. Jégou
Published at: ECCV 2014

Orientation covariant aggregation of local descriptors with embeddings

Giorgos Tolias, Teddy Furon & Hervé Jégou

Inria

Abstract. Image search systems based on local descriptors typically achieve orientation invariance by aligning the patches on their dominant orientations. Albeit successful, this choice introduces too much invariance because it does not guarantee that the patches are rotated consistently. This paper introduces an aggregation strategy of local descriptors that achieves this covariance property by jointly encoding the angle in the aggregation stage in a continuous manner. It is combined with an efficient monomial embedding to provide a codebook-free method to aggregate local descriptors into a single vector representation.
Our strategy is also compatible and employed with several popular encoding methods, in particular bag-of-words, VLAD and the Fisher vector. Our geometric-aware aggregation strategy is effective for image search, as shown by experiments performed on standard benchmarks for image and particular object retrieval, namely Holidays and Oxford buildings.

1 Introduction

This paper considers the problem of particular image or particular object retrieval. This subject has received a sustained attention over the last decade. Many of the recent works employ local descriptors such as SIFT [1] or variants [2] for the low-level description of the images. In particular, approaches derived from the bag-of-visual-words framework [3] are especially successful to solve problems like recognizing buildings. They are typically combined with spatial verification [4] or other re-ranking strategies such as query expansion [5].

Our objective is to improve the quality of the first retrieval stage, before any re-ranking is performed. This is critical when considering large datasets, as re-ranking methods depend on the quality of the initial short-list, which typically consists of a few hundred images. The initial stage is improved by better matching rules, for instance with Hamming embedding [6], by learning a fine vocabulary [7], or weighting the distances [8, 9]. In addition to the SIFT, it is useful to employ some geometrical information associated with the region of interest [6]. All these approaches rely on matching individual descriptors and therefore store some data on a per descriptor basis. Moreover, the quantization of the query's descriptors on a relatively large vocabulary causes delays.

Recently, very short yet effective representations have been proposed based on alternative encoding strategies, such as local linear coding [10], the Fisher vector [11] or VLAD [12]. Most of these representations have been proposed first

for image classification, yet also offer very effective properties in the context of extremely large-scale image search. A feature of utmost importance is that they offer vector representations compatible with cosine similarity. The representation can then be effectively binarized [13] with cosine sketches, such as those proposed by Charikar [14] (*a.k.a.* LSH), or aggressively compressed with principal component dimensionality reduction (PCA) to very short vectors. Product quantization [15] is another example achieving a very compact representation of a few dozens to hundreds bytes and an efficient search because the comparison is done directly in the compressed domain.

This paper focuses on such short- and mid-sized vector representations of images. Our objective is to exploit some geometrical information associated with the regions of interest. A popular work in this context is the spatial pyramid kernel [16], which is widely adopted for image classification. However, it is ineffective for particular image and object retrieval as the grid is too rigid and the resulting representation is not invariant enough, as shown by Douze *et al.* [17].

Here, we aim at incorporating some relative angle information to ensure that the patches are consistently rotated. In other terms, we want to achieve a covariant property similar to that offered by Weak Geometry Consistency (WGC) [6], but directly implemented in the coding stage of image vector representations like Fisher, or VLAD. Some recent works in classification [18] and image search [19] consider a similar objective. They suffer from several shortcomings. In particular, they simply quantize the angle and use it as a pooling variable. Moreover the encoding of a rough approximation of the angles is not straightforwardly compatible with generic match kernels.

In contrast, we achieve the covariant property for any method provided that it can be written as a match kernel. This holds for the Fisher vector, LLC, bag-of-words and efficient match kernels listed in [20]. Our method is inspired by the kernel descriptor of Bo *et al.* [21], from which we borrow the idea of angle kernelization. Our method however departs from this work in several ways. First, we are interested in aggregating local descriptors to produce a vector image representation, whereas they construct new local descriptors. Second, we do not encode the gradient orientation but the dominant orientation of the region of interest jointly with the corresponding SIFT descriptor, in order to achieve the covariant property of the local patches. Finally, we rely on explicit feature maps [22] to encode the angle, which provides a much better approximation than efficient match kernel for a given number of components.

This paper is organized as follows. Section 2 introduces notation and discusses some important related works more in details. Our approach is presented in Section 3 and evaluated in Section 4 on several popular benchmarks for image search, namely Oxford5k [4], Oxford105k and Inria Holidays [23]. These experiments show that our approach gives a significant improvement over the state of the art on image search with vector representations. Importantly, we achieve competitive results by combining our approach with monomial embeddings, *i.e.*, with a *codebook-free* approach, as opposed to coding approaches like VLAD.

2 Preliminaries: match kernels and monomial embeddings

We consider the context of match kernels. An image is typically described by a set of local descriptors $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_i, \dots\},\ \mathbf{x}_i \in \mathbb{R}^d, \|\mathbf{x}_i\| = 1$. Similar to other works [24, 20, 6], two images described by $\mathcal{X}$ and $\mathcal{Y}$ are compared with a match kernel K of the form

$$K(\mathcal{X}, \mathcal{Y}) = \beta(\mathcal{X})\beta(\mathcal{Y}) \sum_{\mathbf{x}\in\mathcal{X}} \sum_{\mathbf{y}\in\mathcal{Y}} k(\mathbf{x}, \mathbf{y}), \tag{1}$$

where k is referred to as the local kernel and where the proportionality factor β ensures that $K(\mathcal{X}, \mathcal{X}) = K(\mathcal{Y}, \mathcal{Y}) = 1$. A typical way to obtain such a kernel is to map the vectors $\mathbf{x}$ to a higher-dimensional space with a function $\varphi : \mathbb{R}^d \to \mathbb{R}^D$, such that the inner product similarity evaluates the local kernel $k(\mathbf{x}, \mathbf{y}) = \langle \varphi(\mathbf{x}) | \varphi(\mathbf{y}) \rangle$. This approach then represents a set of local descriptors by a single vector

$$\mathbf{X} = \beta(\mathcal{X}) \sum_{\mathbf{x}\in\mathcal{X}} \varphi(\mathbf{x}_i), \qquad \text{(such that } \|\mathbf{X}\| = 1\text{)} \tag{2}$$

because the match kernel is computed with a simple inner product as

$$K(\mathcal{X}, \mathcal{Y}) = \beta(\mathcal{X})\beta(\mathcal{Y}) \sum_{\mathbf{x}\in\mathcal{X}} \sum_{\mathbf{y}\in\mathcal{Y}} \langle \varphi(\mathbf{x}) | \varphi(\mathbf{y}) \rangle = \langle \mathbf{X} | \mathbf{Y} \rangle. \tag{3}$$

This framework encompasses many approaches such as bag-of-words [3, 25], LLC [10], Fisher vector [11], VLAD [12], or VLAT [26]. Note that some non-linear processing, such as power-law component-wise normalization [8, 27], is often applied to the resulting vector. A desirable property of k is to have $k(\mathbf{x}, \mathbf{y}) \approx 0$ for unrelated features, so that they do not interfere with the measurements between the true matches. It is somehow satisfied with the classical inner product $k(\mathbf{x}, \mathbf{y}) = \langle \mathbf{x} | \mathbf{y} \rangle$. with the aforementioned embeddings. Several authors [24, 26, 9] propose to increase the contrast between related and unrelated features with a monomial match kernel of degree p of the form

$$K(\mathcal{X}, \mathcal{Y}) = \beta(\mathcal{X})\beta(\mathcal{Y}) \sum_{\mathbf{x}\in\mathcal{X}} \sum_{\mathbf{y}\in\mathcal{Y}} \langle \mathbf{x} | \mathbf{y} \rangle^p. \tag{4}$$

All monomial (and polynomial) embeddings admit exact finite-dimensional feature maps whose length rapidly increases with degree p (in $\mathcal{O}(d^p/p!)$). The order $p = 2$ has already demonstrated some benefit, for instance recently for semantic segmentation [28] or in image classification [26]. In this case, the kernel is equivalent to comparing the set of features based on their covariance matrix [26]. Equivalently, by observing that some components are identical, we can define the embedding $\varphi_2 : \mathbb{R}^d \to \mathbb{R}^{d(d+1)/2}$ mapping $\mathbf{x} = [x_1, \dots, x_d]^\top$ to

$$\varphi_2(\mathbf{x}) = [x_1^2, \dots, x_d^2, x_1 x_2 \sqrt{2}, \dots, x_{d-1} x_d \sqrt{2}]^\top. \tag{5}$$

Similarly, the simplified exact monomial embedding associated with $p = 3$ is the function $\varphi_3 : \mathbb{R}^d \to \mathbb{R}^{(d^3+3d^2+2d)/6}$ defined as

$$\varphi_3(\mathbf{x}) = [x_1^3, \dots, x_d^3, x_1^2 x_2 \sqrt{3}, \dots, x_d^2 x_{d-1} \sqrt{3}, x_1 x_2 x_3 \sqrt{6}, \dots, x_{d-2} x_{d-1} x_d \sqrt{6}]^\top. \tag{6}$$

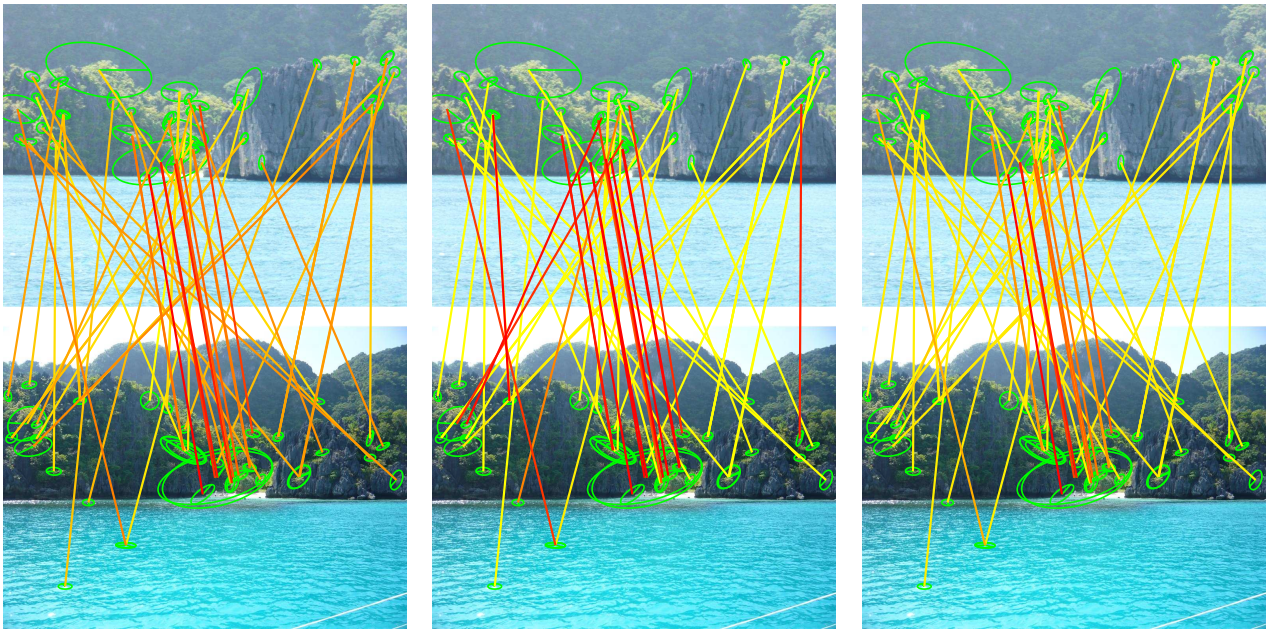


Fig. 1. Similarities between regions of interest, based on SIFT kernel k (*left*), angle consistency kernel k_θ (*middle*) and both (*right*). For each local region, we visualize the values $k(\mathbf{x}, \mathbf{y})$, $k_\theta(\Delta\theta)$ and their product by the colors of the link (red=1).

3 Covariant aggregation of local descriptors

The core idea of the proposed method is to exploit jointly the SIFT descriptors and the dominant orientation θ_x associated with a region of interest. For this purpose, we now assume that an image is represented by a set $\mathcal{X}^\star$ of tuples, each of the form $(\mathbf{x}, \theta_x)$, where $\mathbf{x}$ is a SIFT descriptor and $\theta_x \in [-\pi, \pi]$ is the dominant orientation. Our objective is to obtain an approximation of a match kernel of the form

$$K^\star(\mathcal{X}^\star, \mathcal{Y}^\star) = \beta(\mathcal{X}^\star)\beta(\mathcal{Y}^\star) \sum_{(\mathbf{x},\theta_x)\in\mathcal{X}^\star} \sum_{(\mathbf{y},\theta_y)\in\mathcal{Y}^\star} k(\mathbf{x}, \mathbf{y})\, k_\theta(\theta_x, \theta_y) \tag{7}$$

$$= \langle \mathbf{X}^\star | \mathbf{Y}^\star \rangle, \tag{8}$$

where k is a local kernel identical to that considered in Section 2 and k_θ reflects the similarity between angles. The interest of enriching this match kernel with orientation is illustrated by Figure 1, where we show that several incorrect matches are downweighted thanks to this information.

The kernel in (7) resembles that implemented in WGC [6] with a voting approach. In contrast, we intend to approximate this kernel with an inner product between two vectors as in (8), similar to the linear match kernel simplification in (3). Our work is inspired by the kernel descriptors [21] of Bo *et al.*, who also consider a kernel of a similar form, but at the patch level, to construct a local descriptor from pixel attributes, such as gradient and position.

In our case, we consider the coding/pooling stage and employ a better approximation technique, namely explicit feature maps [22], to encode $\mathcal{X}^\star$. This section first explains the feature map of the angle, then how it modulates the descriptors, and finally discusses the match kernel design and properties.

3.1 A feature map for the angle

The first step is to find a mapping $\boldsymbol{\alpha} : [-\pi, \pi] \to \mathbb{R}^M$ from an angle θ to a vector $\boldsymbol{\alpha}(\theta)$ such that $\boldsymbol{\alpha}(\theta_1)^\top \boldsymbol{\alpha}(\theta_2) = k_\theta(\theta_1 - \theta_2)$. The function $k_\theta : \mathbb{R} \to [0, 1]$ is a shift invariant kernel which should be symmetric ($k_\theta(\Delta\theta) = k_\theta(-\Delta\theta)$), pseudo-periodic with period of 2π and monotonically decreasing over $[0, \pi]$. We consider in particular the following function:

$$k_{\mathsf{VM}}(\Delta\theta) = \frac{\exp(\kappa \cos(\Delta\theta)) - \exp(-\kappa)}{2 \sinh(\kappa)}. \tag{9}$$

It is derived from Von Mises distribution $f(\Delta\theta; \kappa)$, which is often considered as the probability density distribution of the noise of the measure of an angle, and therefore regarded as the equivalent Gaussian distribution for angles. Although this is not explicitly stated in their paper, the regular Von Mises distribution is the kernel function implicitly used by Bo *et al.* [21] for kernelizing angles. Our function k_{VM} is a shifted and scaled variant of Von Mises, designed such that its range is $[0, 1]$, which ensures that $k_{\mathsf{VM}}(\pi) = 0$.

The periodic function k_{VM} can be expressed as a Fourier series whose coefficients are (see [29][Eq. (9.6.19)]):

$$k_{\mathsf{VM}}(\Delta\theta) = \left(I_0(\kappa) - e^{-\kappa} + 2 \sum_{n=1}^{\infty} I_n(\kappa) \cos(n\Delta\theta) \right) . \frac{1}{2 \sinh(\kappa)}, \tag{10}$$

where $I_n(\kappa)$ is the modified Bessel function of the first kind of order n. We now consider the truncation $\bar{k}^N_{\mathsf{VM}}$ of the series to the first N terms:

$$\bar{k}^N_{\mathsf{VM}}(\Delta\theta) = \sum_{n=0}^{N} \gamma_n \cos(n\Delta\theta) \quad \text{with } \gamma_0 = \frac{(I_0(\kappa) - e^{-\kappa})}{2 \sinh(\kappa)} \text{ and } \gamma_n = \frac{I_n(\kappa)}{\sinh(\kappa)} \text{ if } n > 0. \tag{11}$$

We design the feature map $\boldsymbol{\alpha}(\theta)$ as follows:

$$\boldsymbol{\alpha}(\theta) = (\sqrt{\gamma_0}, \sqrt{\gamma_1}\cos(\theta), \ldots, \sqrt{\gamma_N}\cos(N\theta), \sqrt{\gamma_1}\sin(\theta), \ldots, \sqrt{\gamma_N}\sin(N\theta))^\top. \tag{12}$$

This vector has $2N + 1$ components. Moreover

$$\boldsymbol{\alpha}(\theta_1)^\top \boldsymbol{\alpha}(\theta_2) = \gamma_0 + \sum_{n=1}^{N} \gamma_n (\cos(n\theta_1)\cos(n\theta_2) + \sin(n\theta_1)\sin(n\theta_2)) \tag{13}$$

$$= \sum_{n=0}^{N} \gamma_n \cos(n(\theta_1 - \theta_2)) \tag{14}$$

$$= \bar{k}^N_{\mathsf{VM}}(\theta_1 - \theta_2) \approx k_{\mathsf{VM}}(\theta_1 - \theta_2) \tag{15}$$

This process of designing a feature map is explained in full details by Vedaldi and Zisserman [22]. This feature map gives an approximation of the target function k_{VM}, which is more accurate as N is bigger.

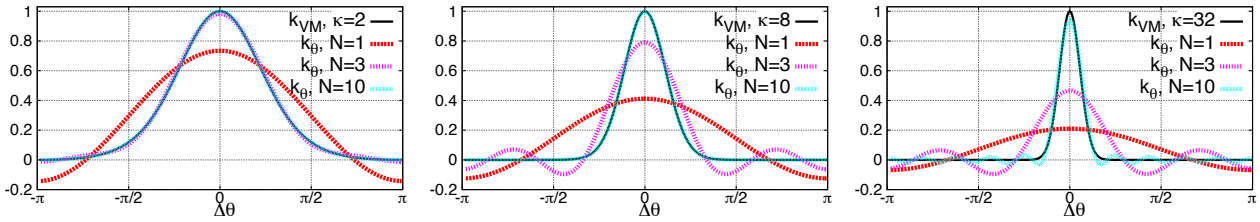


Fig. 2. Function k_{VM} for different values of κ and its approximation $\bar{k}^N_{\mathsf{VM}}$ using 1, 3 and 10 frequencies, as implicitly defined by the corresponding mapping $\boldsymbol{\alpha} : [\pi, \pi] \rightarrow \mathbb{R}^{2N+1}$.

Figure 2 illustrates the function k_{VM} for several values of the parameter κ and its approximation $\bar{k}^N_{\mathsf{VM}}$ for different values of N. First note that $\bar{k}^N_{\mathsf{VM}}$ may not fulfill the original requirements: its range might be wider than $[0, 1]$ and it might not be monotonically decreasing over $[0, \pi]$. Larger values of κ produce a more "selective" function of the angle, yet require more components (larger N) to obtain an accurate estimation. Importantly, the approximation stemming from this explicit angle mapping is better than that based on efficient match kernels [20], which converges slowly with the number of components. Efficient match kernels are more intended to approximate kernels on vectors than on scalar values. As a trade-off between selectivity and the number of components, we set κ=8 and N=3 (see Section 4). Accordingly, we use $\bar{k}^3_{\mathsf{VM}}$ as k_θ in the sequel. The corresponding embedding $\boldsymbol{\alpha} : \mathbb{R} \rightarrow \mathbb{R}^7$ maps any angle to a 7-dimensional vector.

Remark: Instead of approximating a kernel on angles with finite Fourier series, one may rather consider directly designing a function satisfying our initial requirements (pseudo-period, symmetric, decreasing over $[0, \pi]$), such as

$$k_P(\Delta\theta) = \cos(\Delta\theta/2)^P \text{ with } P \text{ even.} \tag{16}$$

This function, thanks to power reduction trigonometric identities for even P, is re-written as

$$k_P(\Delta\theta) = \sum_{p=0}^{P/2} \gamma_p \cos(p\Delta\theta) \tag{17}$$

$$\text{with} \qquad \gamma_0 = \frac{1}{2^P}\binom{P}{P/2}, \gamma_p = \frac{1}{2^{P-1}}\binom{P}{P/2-p} \quad 0 < p \leq P/2. \tag{18}$$

Applying (12) leads to a feature map $\boldsymbol{\alpha}(\theta)$ with $P+1$ components such that $\boldsymbol{\alpha}(\theta_1)^\top \boldsymbol{\alpha}(\theta_2) = k_P(\theta_1 - \theta_2)$. For this function, the interesting property is that the scalar product is exactly equal to the target kernel value $k_P(\theta_1 - \theta_2)$, and that the original requirements now hold. From our experiments, this function gives reasonable results, but requires more components than $\bar{k}_{\mathsf{VM}}$ to achieve a shape narrow around $\Delta\theta = 0$ and close to 0 otherwise. The results for our image search application task using this function are slightly below our Von Mises variant for a given dimensionality. So, despite its theoretical interest we do not use it in our experiments. Ultimately, one would rather directly learn a Fourier embedding for the targeted task, in the spirit of recent works on Fourier kernel learning [30].

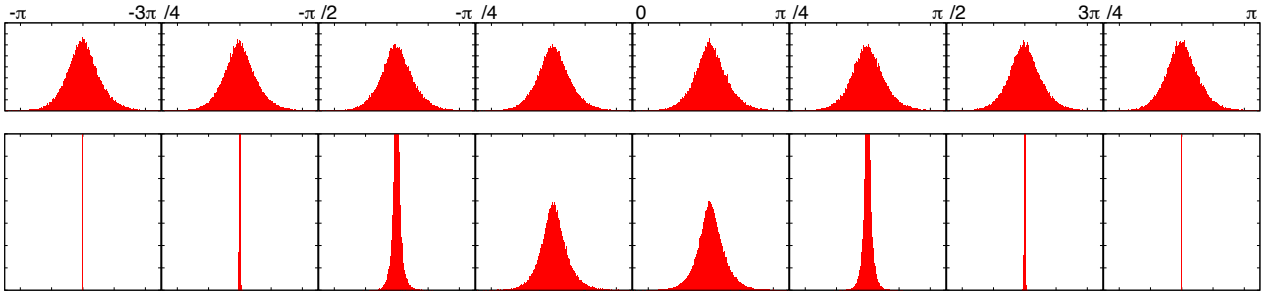


Fig. 3. Distribution of patch similarity for different values of orientation difference. In this figure, we split the angular space into 8 equally-sized bins and present the similarity distribution separately for each of these bins. Horizontal axis represents the similarity value between matching features. *Top*: distribution of similarities with kernel on SIFTs. *Bottom:* Distribution after modulation with α.

3.2 Modulation and covariant match kernel

The vector $\boldsymbol{\alpha}$ encoding the angle θ "modulates"[1] any vector $\mathbf{x}$ (or pre-mapped descriptor $\varphi(\mathbf{x})$) with a function $m : \mathbb{R}^{2N+1} \times \mathbb{R}^D \to \mathbb{R}^{(2N+1)D}$. Thanks to classical properties of the Kronecker product $\otimes$, we have

$$m(\mathbf{x}, \boldsymbol{\alpha}(\theta)) = \mathbf{x} \otimes \boldsymbol{\alpha}(\theta) = (x_1 \boldsymbol{\alpha}(\theta)^\top, x_2 \boldsymbol{\alpha}(\theta)^\top \ldots, x_d \boldsymbol{\alpha}(\theta)^\top)^\top. \tag{19}$$

We now consider two pairs of vectors and angle, $(\mathbf{x}, \theta_x)$ and $(\mathbf{y}, \theta_y)$, and their modulated descriptors $m(\mathbf{x}, \boldsymbol{\alpha}(\theta_x))$ and $m(\mathbf{y}, \boldsymbol{\alpha}(\theta_y))$. In the product space $\mathbb{R}^{(2N+1)D}$, the following holds:

$$\begin{aligned} m(\mathbf{x}, \boldsymbol{\alpha}(\theta_x))^\top m(\mathbf{y}, \boldsymbol{\alpha}(\theta_y) &= (\mathbf{x} \otimes \boldsymbol{\alpha}(\theta_x))^\top (\mathbf{y} \otimes \boldsymbol{\alpha}(\theta_y)) \\ &= (\mathbf{x}^\top \otimes \boldsymbol{\alpha}(\theta_x)^\top)(\mathbf{y} \otimes \boldsymbol{\alpha}(\theta_y)) = (\mathbf{x}^\top \mathbf{y}) \otimes (\boldsymbol{\alpha}(\theta_x)^\top \boldsymbol{\alpha}(\theta_y)) \\ &= (\mathbf{x}^\top \mathbf{y}) k_\theta(\theta_x - \theta_y). \end{aligned} \tag{20}$$

Figure 3 shows the distribution of the similarities between regions of interest before and after modulation, as a function of the difference of angles. Interestingly, there is no obvious correlation between the difference of angle and the SIFT: the similarity distribution based on SIFT is similar for all angles. This suggests that the modulation with angle provides complementary information.

Combination with coding/pooling techniques. Consider any coding method φ that can be written as match kernel (Fisher, LLC, Bag-of-words, VLAD, etc). The match kernel in (7), with our k_θ approximation, is re-written as

$$\begin{aligned} K^\star(\mathcal{X}^\star, \mathcal{Y}^\star) =& \beta(\mathcal{X}^\star)\beta(\mathcal{Y}^\star) \sum_{(\mathbf{x},\theta_x)\in\mathcal{X}^\star} \sum_{(\mathbf{y},\theta_y)\in\mathcal{Y}^\star} m(\varphi(\mathbf{x}), \boldsymbol{\alpha}(\theta_x))^\top m(\varphi(\mathbf{y}), \boldsymbol{\alpha}(\theta_y)), \\ =& \beta(\mathcal{X}^\star) \left(\sum_{(\mathbf{x},\theta_x)} m(\varphi(\mathbf{x}), \boldsymbol{\alpha}(\theta_x)) \right)^\top \beta(\mathcal{Y}^\star) \left(\sum_{(\mathbf{y},\theta_y)} m(\varphi(\mathbf{y}), \boldsymbol{\alpha}(\theta_y) \right), \end{aligned} \tag{21}$$

[1] By analogy to communications, where modulation refers to the process of encoding information over periodic waveforms.

where we observe that the image can be represented as the summation $\mathbf{X}^\star$ of the embedded descriptors modulated by their corresponding dominant orientation, as

$$\mathbf{X}^\star = \beta(\mathcal{X}^\star) \sum_{(\mathbf{x},\theta_x)\in\mathcal{X}^\star} m(\varphi(\mathbf{x}), \boldsymbol{\alpha}(\theta_x)). \tag{22}$$

This representation encodes the relative angles and is already more discriminative than an aggregation that does not consider them. However, at this stage, the comparison assumes that the images have the same global orientation. This is the case on benchmarks like Oxford5k building, where all images are orientated upright, but this is not true in general for particular object recognition.

3.3 Rotation invariance

We now describe how to produce a similarity score when the orientations of related images may be different. We represent the image vector $\mathbf{X}^\star$ as the concatenation of $2N+1$ D-dimensional subvectors associated to one term of the finite Fourier series: $\mathbf{X}^\star = [\mathbf{X}_0^{\star\top}, \mathbf{X}_{1,c}^{\star\top}, \mathbf{X}_{1,s}^{\star\top}, \ldots, \mathbf{X}_{N,c}^{\star\top}, \mathbf{X}_{N,s}^{\star\top}]^\top$. The vector $\mathbf{X}_0^\star$ is associated with the constant term in the Fourier expansion, $\mathbf{X}_{n,c}^\star$ and $\mathbf{X}_{n,s}^\star$, $1 \le n \le N$, correspond to the cosine and sine terms, respectively.

Imagine now that this image undergoes a global rotation of angle θ. Denote $\breve{\mathcal{X}}$ the new set of pairs $(\mathbf{x}, \breve{\theta}_x)$ with $\breve{\theta}_x = \theta_x + \theta$, and $\breve{\mathbf{X}}^\star$ is the new image vector derived from these local descriptors. It occurs that $\breve{\mathbf{X}}_0^\star = \mathbf{X}_0^\star$ because this term does not depend on the angle, and that, for a given frequency bin n, elementary trigonometry identities lead to

$$\breve{\mathbf{X}}_{n,c}^\star = \mathbf{X}_{n,c}^\star \cos n\theta + \mathbf{X}_{n,s}^\star \sin n\theta \tag{23}$$

$$\breve{\mathbf{X}}_{n,s}^\star = -\mathbf{X}_{n,c}^\star \sin n\theta + \mathbf{X}_{n,s}^\star \cos n\theta. \tag{24}$$

This in turn shows that $\|\breve{\mathbf{X}}^\star\| = \|\mathbf{X}^\star\|$. Therefore the rotation has no effect on the normalization factor $\beta(\mathcal{X}^\star)$.

When comparing two images with such vectors, the linearity of the inner product ensures that

$$\langle \breve{\mathbf{X}}^\star | \mathbf{Y}^\star \rangle = \langle \mathbf{X}_0^\star | \mathbf{Y}_0^\star \rangle + \sum_{n=1}^{N} \cos n\theta \ \left(\langle \mathbf{X}_{n,c}^\star | \mathbf{Y}_{n,c}^\star \rangle + \langle \mathbf{X}_{n,s}^\star | \mathbf{Y}_{n,s}^\star \rangle \right) \tag{25}$$

$$+ \sum_{n=1}^{N} \sin n\theta \ \left(-\langle \mathbf{X}_{n,c}^\star | \mathbf{Y}_{n,s}^\star \rangle + \langle \mathbf{X}_{n,s}^\star | \mathbf{Y}_{n,c}^\star \rangle \right). \tag{26}$$

Here, we stress that the similarity between two images is a real trigonometric polynomial in θ (rotation angle) of degree N. Its $2N+1$ components are fully determined by computing $\langle \mathbf{X}_0^\star | \mathbf{Y}_0^\star \rangle$ and the inner products between the subvectors associated with each frequency, *i.e.*, $\langle \mathbf{X}_{n,c}^\star | \mathbf{Y}_{n,c}^\star \rangle$, $\langle \mathbf{X}_{n,s}^\star | \mathbf{Y}_{n,s}^\star \rangle$, $\langle \mathbf{X}_{n,c}^\star | \mathbf{Y}_{n,s}^\star \rangle$ and $\langle \mathbf{X}_{n,s}^\star | \mathbf{Y}_{n,c}^\star \rangle$. Finding the maximum of this polynomial amounts to finding the rotation maximizing the score between the two images.

Computing the coefficients of this polynomial requires a total of $D \times (1+4N)$ elementary operations for a vector representation of dimensionality $D \times (1+2N)$, that is, less than twice the cost of the inner product between $\mathbf{X}^\star$ and $\mathbf{Y}^\star$. Once these components are obtained, the cost of finding the maximum value achieved by this polynomial is negligible for large values of D, for instance by simply sampling a few values of θ. Therefore, if we want to offer the orientation invariant property, the complexity of similarity computation is typically twice the cost of that of a regular vector representation (whose complexity is equal to the number of dimensions).

Remark: This strategy for computing the scores for all possible orientations of the query is not directly compatible with non-linear post-processing of $\mathbf{X}^\star$ such as component-wise power-law normalization [27], except for the subvector $\mathbf{X}_0^\star$. We propose two possible options to overcome this problem.

1. The naive strategy is to compute the query for several hypothesis of angle rotation, typically 8. In theory, this multiplies the query complexity by the same factor 8. However, in practice, it is faster to perform the matrix-matrix multiplication, with the right matrix representing 8 queries, than computing separately the corresponding 8 matrix-vector multiplications. We use this simpler approach in the experimental section.
2. Alternately, the power-law normalization is adapted to become compatible with our strategy: we compute the modulus of the complex number represented by two components (sin and cos) associated with the same frequency n and the same original component in $\varphi(\mathbf{x})$. These two components are then divided by the square-root (or any power) of this modulus. Experimentally, this strategy is as effective as the naive option.

4 Experiments

We evaluate the performance of the proposed approaches and compare with state of the art methods on two publicly available datasets for image and particular object retrieval, namely Inria Holidays [23] and Oxford Buildings 5k [4]. We also combine the latter with 100k distractor images to measure the performance on a larger scale. The merged dataset is referred to as Oxford105k. The retrieval performance is measured with mean Average Precision (mAP) [4].

Our approach modulates any coding/pooling technique operating as a match kernel. Therefore, we evaluate the benefit of our approach combined with several coding techniques, namely

- VLAD [12], which encodes a SIFT descriptor by considering the residual vector to the centroid.
- The Fisher vector [11, 27, 31]. For image classification, Chatfield *et al.* [32] show that it outperforms concurrent coding techniques, in particular LLC [10]. We adopt the standard choice for image retrieval and use only the gradient with respect to the mean [12].

- Monomomial embeddings of order 2 and 3 applied on local descriptors (See below for pre-processing), *i.e.*, the functions φ_2 in (5) and φ_3 in (6). For the sake of consistency, we also denote by φ_1 the function $\varphi_1 : x \to x$.

We refer to these methods combined with our approach with the symbol "$\otimes$": VLAD$\otimes$, Fisher$\otimes$, $\varphi_1\otimes$, $\varphi_2\otimes$ and $\varphi_3\otimes$, correspondingly. In addition, we compare against the most related work, namely the recent CVLAD [19] method, which also aims at producing an image vector representation integrating the dominant orientations of the patches. Whenever the prior work is not referenced, results are produced using our own (improved) implementations of VLAD, Fisher and CVLAD, so that the results are directly comparable with the same features.

4.1 Implementation Details

Local descriptors. We use the Hessian-Affine detector [33] to extract the regions of interest, that are subsequently described by SIFT descriptors [1] post-processed with RootSIFT [34]. Then, following the pre-processing required for the Fisher vector [11, 27, 12], we apply PCA to reduce the vector to 80 components. An exception is done for VLAD and CVLAD with which we only use the PCA basis to center and rotate descriptors as suggested by Delhumeau [35], without dimensionality reduction. The resulting vector is subsequently ℓ_2-normalized.

The improved Hessian-Affine detector of Perdoch *et al.* [36] improves the retrieval performance. However, we do not use it, since it ignores rotations by making the gravity vector assumption. Instead, we use the original detector modified so that it has similar parameters (patch size to 41).

Codebook. For all methods based on codebooks, we only consider distinct datasets for learning. More precisely and following common practice, the k-means and GMM (for VLAD and Fisher, respectively) are learned on Flickr60k for Inria Holidays and Paris6k [37] for Oxford buildings. We rely on the Yael library [38] for codebook construction and VLAD and Fisher encoding.

Post-processing. The final image vector obtained by each method is power-law normalized [8, 27, 12]. This processing improves the performance by efficiently handling the burstiness phenomenon. Exploiting the dominant orientation in our covariant match kernel provides a complementary way to further handle the same problem. We mention that using the dominant orientation is shown effective in a recent work by Torii *et al.* [39]. With our angle modulation, this post-processing inherently captures and down weights patches with similar dominant orientation. The power-law exponent is set to 0.4 for Fisher and VLAD and to 0.2 for monomial embeddings. These values give best or close-to-best performance for the initial representations. The resulting vector is ℓ_2-normalized.

In addition to power-law normalization, we rotate the aggregated vector representation with a PCA rotation matrix [40, 41]. This aims at capturing the co-occurrences to down-weight them either by whitening [40] or a second power-law normalization [41]. We adopt the latter choice (with exponent 0.5) to avoid

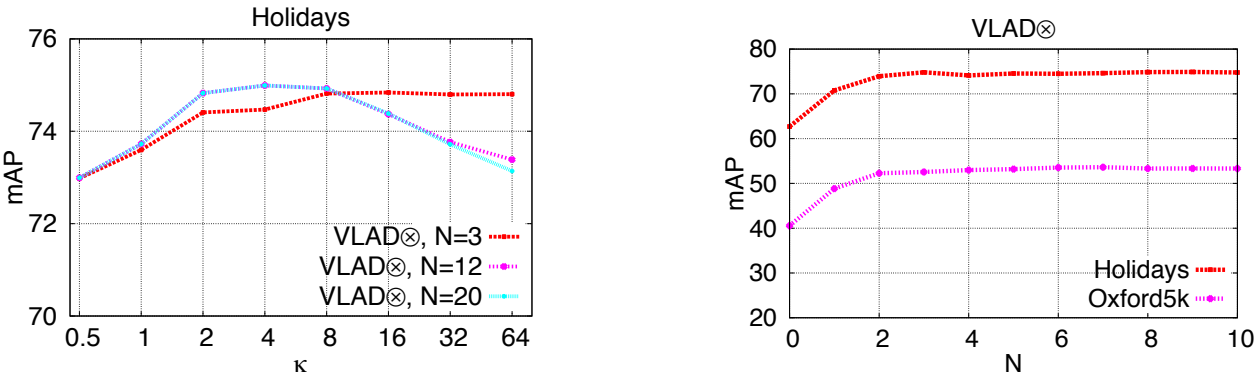


Fig. 4. Left: Performance on Holidays dataset of modulated VLAD for different values of κ and for different approximations. Right: Performance comparison of modulated VLAD for increasing number of components of the angle feature map. Zero corresponds to original VLAD (not modulated).

the sensitivity to eigenvalues (in whitening) when learning PCA with few input data. We refer to this Rotation and Normalization as RN in our experiments.

Optionally, to produce compact representations, we keep only the first few components (the most energetic ones) and ℓ_2-normalize the shortened vector.

Query rotation. In order to obtain rotation invariance jointly with power-law normalization and RN, we apply rotations of the query image and apply individual queries as described in Section 3 (option 1). We apply 8 query rotations on Holidays dataset. On Oxford5k, we rather adopt the common choice of not considering other possible orientations: Possible rotation of the query object is usually not considered since all the images are up-right.

4.2 Impact of the parameters

The impact of the angle modulation is controlled by the function k_θ parametrized by κ and N. As shown in Figure 2, the value κ typically controls the "bandwitdh", *i.e.*, the range of $\Delta\theta$ values with non-zero response. The parameter N controls the quality of the approximation, and implicitly constrains the achievable bandwidth. It also determines the dimensionality of the output vector.

Figure 4 (left) shows the impact of these parameters on the performance. As to be expected, there is a trade-off between defining too narrow or too large. The optimal performance is achieved with κ in the range $[2, 8]$. Figure 4 (right) shows the performance for increasing number of frequencies, which rapidly converges to a fixed mAP. This is the mAP of the exact evaluation of (7). We set $N = 3$ as a compromise between dimensionality expansion and performance. Therefore the modulation multiplies the input dimensionality by 7.

4.3 Benefit of our approach

Table 1 shows the benefit of modulation when applied to the monomial embeddings φ_1, φ_2 and φ_3. The results are on par with the recent coding techniques like

Method	φ_1	$\varphi_1\otimes$			φ_2		$\varphi_2\otimes$				φ_3	$\varphi_3\otimes$
RN						×			×	×		
N	–	1	3	6		–	1	3	1	3	–	1
#dim	80	240	560	1,040	3240	3,240	9,720	22,680	9,720	22,680	88,560	265,680
mAP	35.4	48.9	59.5	**63.2**	59.7	71.6	68.8	73.7	75.3	**79.9**	60.0	72.5

Table 1. Impact of modulation on monomial embeddings of order 1, 2 and 3. The performance is reported for Holidays dataset. RN = Rotation and Normalization.

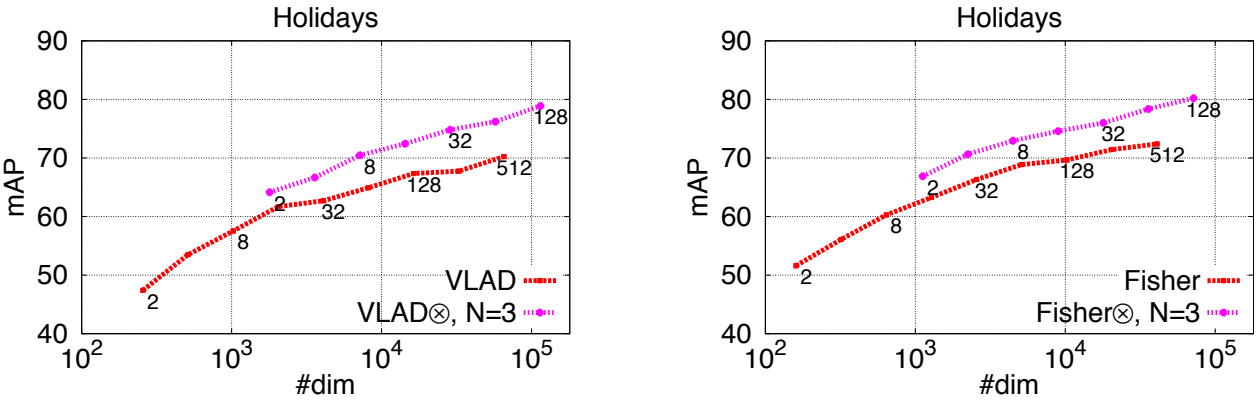


Fig. 5. Impact of modulation on VLAD and Fisher: Performance versus dimensionality of the final vector for VLAD (left) and Fisher (right) compared to their modulated counterparts. Codebook size is shown with text labels. Results for Holidays dataset.

VLAD or Fisher improved with modulation. We consider the obtained performance as one of our main achievements, because the representation is codebook-free and requires no learning. In addition, we further show the benefit of combining monomial embeddings with RN. This significantly boosts performance with the same vector dimensionality and negligible computational overhead.

We compare VLAD, Fisher and monomial embeddings to their modulated counterparts. Figure 5 shows that modulation significantly improves the performance for the same codebook size. However, given that the modulated vector is $\times 7$ larger (with $N = 3$), the comparison focuses on the performance obtained with the same dimensionality. Even in this case, modulated VLAD$\otimes$ and Fisher$\otimes$ offer a significant improvement. We can conclude that it is better to increase the dimensionality by modulation than using a larger codebook.

4.4 Comparison to other methods

We compare our approach, in particular, to CVLAD, as this work also intends to integrate the dominant orientation into a vector representation. We consistently apply 8 query rotations for both CVLAD and our method on Holidays dataset. Figure 6 shows the respective performance measured for different codebooks. The proposed methods appear to consistently outperform CVLAD, both for the same codebook and for the same dimensionality. Noticeably, the modulated embedded monomial $\varphi_2\otimes$ is on par with or better than CVLAD.

We further conduct experiments using oriented dense [19] to compare VLAD$\otimes$ to CVLAD. They achieve 87.2 and 86.5 respectively, on Holidays with codebook of size 512. This score is significantly higher than the one reported in [19]. Corresponding scores on Oxford5k are 50.5 and 50.7, respectively. However, note that it is very costly to densely extract patches aligned with dominant orientation.

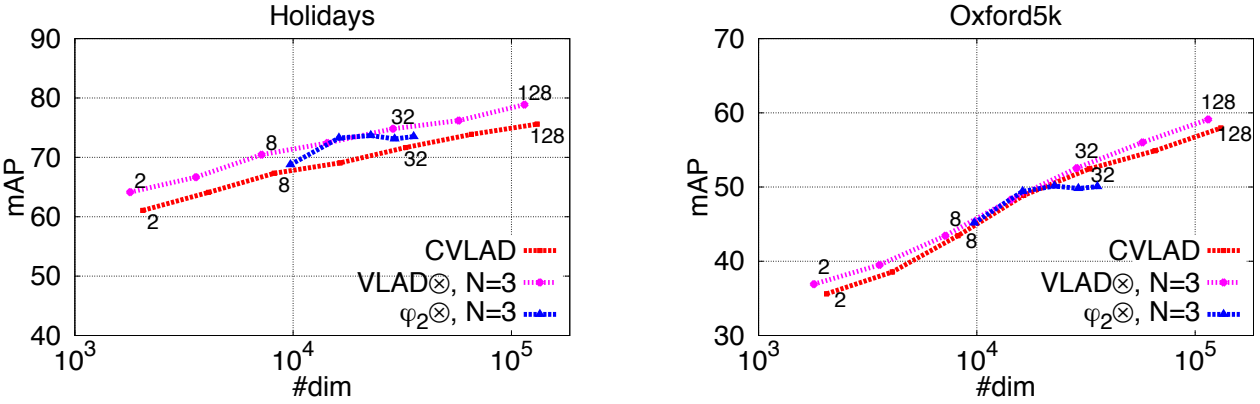


Fig. 6. Comparison to CVLAD. We measure performance on Holidays and Oxford5k for CVLAD and our proposed methods for increasing codebook size. The codebook cardinality is shown with text labels for CVLAD and modulated VLAD, while for φ_2 the number of frequency components (N) used are increased from 1 to 5.

Method	#C	#dim	RN	Holidays	Oxford5k	Oxford105k
VLAD [12]	64	4,096		55.6	37.8	-
Fisher [12]	64	4,096		59.5	41.8	-
VLAD [12]	256	16,384		58.7	-	-
Fisher [12]	256	16,384		62.5	-	-
Arandjelovic [42]	256	32,536		65.3	**55.8**	-
Delhumeau [35]	64	8,192		65.8	51.7	45.6
Zhao [19]	32	32,768		**68.8**	42.7	-
VLAD⊗	32	28,768		74.8	52.5	46.3
VLAD⊗	32	28,768	×	81.0	61.8	**53.9**
Fisher⊗	32	17,920		76.0	51.0	44.9
Fisher⊗	32	17,920	×	81.2	60.7	52.2
Fisher⊗	64	35,840	×	**84.1**	**64.8**	-
φ_2⊗	n/a	22,680		73.7	50.1	44.3
φ_2⊗	n/a	22,680	×	79.9	60.5	51.9
φ_3⊗	n/a	265,680		72.5	53.5	-

Table 2. Performance comparison with state of the art approaches. Results with the use of full vector representation. #C: size of codebook. #dim: Number of components of each vector. Modulation is performed with $N = 3$ for all cases, except to φ_3, where $N = 1$. We do not use any re-ranking or spatial verification in any experiment. VLAD⊗ achieves **87.2** on Holidays and 80.5 on Oxford5k with $\#C$=512 and oriented dense.

Method	#dim	full dim	dim→1024	dim→128
VLAD	4,096	40.3	34.7	24.0
VLAD⊗	28,672	53.9	**40.7** (+7.0)	**27.5** (+3.5)
Fisher	2,560	39.3	37.3	25.2
Fisher⊗	17,920	52.2	39.9 (+2.6)	26.5 (+1.3)
φ_2	3,240	35.8	31.1	20.4
φ_2⊗	22,680	51.9	37.7 (+6.6)	24.0 (+3.6)

Table 3. Oxford105k: Performance comparison (mAP) after dimensionality reduction with PCA into 128 and 1024 components. The results with the full vector representation are with RN. Observe the consistent gain (in parentheses) brought by our approach for a *fixed* output dimensionality of 1,024 or 128 components.

We also compare to other prior works and present results in Table 2 for Holidays, Oxford5k and Oxford105k. We outperform by a large margin the state of the art with full vector representations. Further, our approach is arguably compatible with these concurrent approaches, which may bring further improvement. Note that RN also boosts performance for VLAD and Fisher. In particular with a codebook of size 32, they achieve 50.0 and 48.6 respectively on Oxford5k. Our scores on Holidays with Fisher$\otimes$ and RN are also competitive to those reported by state-of-the-art methods based on large codebooks [9]. To our knowledge, this is the first time that a vector representation compatible with inner product attains such image search performance.

On Oxford5k we do not evaluate multiple query rotations for our method. A simple way to enforce up-right objects for baseline methods is to use up-right features. Performance of VLAD with codebook of size 256 decreases from 51.3 to 49.4 by doing so, presumably because of small object rotations.

Finally, Table 3 reports the performance after dimensionality reduction to 128 or 1024 components. The same set of local features and codebooks are used for all methods. We observe a consistent improvement over the original encoding.

4.5 Timings

The image representation created by modulating the monomial embedding φ_2 using $N = 3$ takes on average 68 ms for a typical image with 3,000 SIFT descriptors. The resulting aggregated vector representation has 22,680 components. The average query time using cosine similarity on Oxford5k is 44 ms assuming no query rotation and 257 ms with the use of 8 possible fixed rotations (with the naive strategy discussed in Section 3.3). The corresponding timings for Oxford105k and vectors reduced to 128 dimensions are 55 ms and 134 ms, respectively. Note, these timings are better than those achieved by a bag-of-words representation with a large vocabulary, for which the quantization typically takes above 1 second with an approximate nearest neighbor search algorithm like FLANN [43].

5 Conclusion

Our modulation strategy integrates the dominant orientation directly in the coding stage. It is inspired by and builds upon recent works on explicit feature maps and kernel descriptors. Thanks to a generic formulation provided by match kernels, it is compatible with coding strategies such as Fisher vector or VLAD. Our experiments demonstrate that it gives a consistent gain compared to the original coding in all cases, even after dimensionality reduction. Interestingly, it is also very effective with a simple monomial kernel, offering competitive performance for image search with a coding stage not requiring any quantization.

Whatever the coding stage that we use with our approach, the resulting representation is compared with inner product, which suggests that it is compliant with linear classifiers such as those considered in image classification.

Acknowledgments. This work was supported by ERC grant Viamass no. 336054 and ANR project Fire-ID.

References

1. Lowe, D.: Distinctive image features from scale-invariant keypoints. IJCV **60**(2) (Nov. 2004) 91–110
2. Bay, H., Ess, A., Tuytelaars, T., Gool, L.V.: SURF: Speeded up robust features. Computer Vision and Image Understanding **110**(3) (May 2008) 346–359
3. Sivic, J., Zisserman, A.: Video Google: A text retrieval approach to object matching in videos. In: ICCV. (Oct. 2003)
4. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching. In: CVPR. (Jun. 2007)
5. Chum, O., Philbin, J., Sivic, J., Isard, M., Zisserman, A.: Total recall: Automatic query expansion with a generative feature model for object retrieval. In: ICCV. (Oct. 2007)
6. Jégou, H., Douze, M., Schmid, C.: Improving bag-of-features for large scale image search. IJCV **87**(3) (Feb. 2010) 316–336
7. Mikulík, A., Perdoch, M., Chum, O., Matas, J.: Learning a fine vocabulary. In: ECCV. (Sep. 2010)
8. Jégou, H., Douze, M., Schmid, C.: On the burstiness of visual elements. In: CVPR. (Jun. 2009)
9. Tolias, G., Avrithis, Y., Jégou, H.: To aggregate or not to aggregate: Selective match kernels for image search. In: ICCV. (Dec. 2013)
10. Wang, J., Yang, J., K. Yu, F.L., Huang, T., Gong, Y.: Locality-constrained linear coding for image classification. In: CVPR. (Jun. 2010)
11. Perronnin, F., Dance, C.R.: Fisher kernels on visual vocabularies for image categorization. In: CVPR. (Jun. 2007)
12. Jégou, H., Perronnin, F., Douze, M., Sánchez, J., Pérez, P., Schmid, C.: Aggregating local descriptors into compact codes. In: Trans. PAMI. (Sep. 2012)
13. Perronnin, F., Liu, Y., Sanchez, J., Poirier, H.: Large-scale image retrieval with compressed Fisher vectors. In: CVPR. (Jun. 2010)
14. Charikar, M.: Similarity estimation techniques from rounding algorithms. In: STOC. (May 2002)
15. Jégou, H., Douze, M., Schmid, C.: Product quantization for nearest neighbor search. Trans. PAMI **33**(1) (Jan. 2011) 117–128
16. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In: CVPR. (Jun. 2006)
17. Douze, M., Jégou, H., Singh, H., Amsaleg, L., Schmid, C.: Evaluation of GIST descriptors for web-scale image search. In: CIVR. (July 2009)
18. Koniusz, P., Yan, F., Mikolajczyk, K.: Comparison of mid-level feature coding approaches and pooling strategies in visual concept detection. Computer Vision and Image Understanding **17**(5) (May 2013) 479–492
19. Zhao, W., Jégou, H., Gravier, G.: Oriented pooling for dense and non-dense rotation-invariant features. In: BMVC. (Sep. 2013)
20. Bo, L., Sminchisescu, C.: Efficient match kernel between sets of features for visual recognition. In: NIPS. (Dec. 2009)
21. Bo, L., Ren, X., Fox, D.: Kernel descriptors for visual recognition. In: NIPS. (Dec. 2010)
22. Vedaldi, A., Zisserman, A.: Efficient additive kernels via explicit feature maps. Trans. PAMI **34**(3) (Mar. 2012) 480–492
23. Jégou, H., Douze, M., Schmid, C.: Hamming embedding and weak geometric consistency for large scale image search. In: ECCV. (Oct. 2008)

24. Lyu, S.: Mercer kernels for object recognition with local features. In: CVPR. (Jun. 2005)
25. Csurka, G., Dance, C.R., Fan, L., Willamowski, J., Bray, C.: Visual categorization with bags of keypoints. In: ECCV Workshop Statistical Learning in Computer Vision. (May 2004)
26. Picard, D., Gosselin, P.H.: Efficient image signatures and similarities using tensor products of local descriptors. Computer Vision and Image Understanding **117** (Jun. 2013)
27. Perronnin, F., Sánchez, J., Mensink, T.: Improving the Fisher kernel for large-scale image classification. In: ECCV. (Sep. 2010)
28. Caseiro, J.C.R., Batista, J., Sminchisescu, C.: Semantic segmentation with second-order pooling. In: ECCV. (Oct. 2012)
29. Abramowitz, M., Stegun, I.A.: Handbook of mathematical functions with formulas, graphs, and mathematical tables. Volume 55 of National Bureau of Standards Applied Mathematics Series. U.S. Government Printing Office (1964)
30. Bazavan, E.G., Li, F., , Sminchisescu, C.: Fourier kernel learning. In: ECCV. (Oct. 2012)
31. Jaakkola, T., Haussler, D.: Exploiting generative models in discriminative classifiers. In: NIPS. (Dec. 1998)
32. Chatfield, K., Lempitsky, V., Vedaldi, A., Zisserman, A.: The devil is in the details: an evaluation of recent feature encoding methods. In: BMVC. (Sep. 2011)
33. Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., Gool, L.V.: A comparison of affine region detectors. IJCV **65**(1/2) (Nov. 2005) 43–72
34. Arandjelovic, R., Zisserman, A.: Three things everyone should know to improve object retrieval. In: CVPR. (Jun. 2012)
35. Delhumeau, J., Gosselin, P.H., Jégou, H., Pérez, P.: Revisiting the VLAD image representation. In: ACM Multimedia. (Oct. 2013)
36. Perdoch, M., Chum, O., Matas, J.: Efficient representation of local geometry for large scale object retrieval. In: CVPR. (Jun. 2009)
37. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Lost in quantization: Improving particular object retrieval in large scale image databases. In: CVPR. (Jun. 2008)
38. Douze, M., Jégou, H.: The Yael library. In: ACM Multimedia. (Nov. 2014)
39. Torii, A., Sivic, J., Pajdla, T., Okutomi, M.: Visual place recognition with repetitive structures. In: CVPR. (Jun. 2013)
40. Jégou, H., Chum, O.: Negative evidences and co-occurences in image retrieval: The benefit of PCA and whitening. In: ECCV. (Oct. 2012)
41. Safadi, B., Quenot, G.: Descriptor optimization for multimedia indexing and retrieval. In: CBMI. (Jun. 2013)
42. Arandjelovic, R., Zisserman, A.: All about VLAD. In: CVPR. (Jun. 2013)
43. Muja, M., Lowe, D.G.: Fast approximate nearest neighbors with automatic algorithm configuration. In: VISAPP. (Feb. 2009)

IV Rotation and translation covariant match kernels for image retrieval

Title: Rotation and translation covariant match kernels for image retrieval
Authors: G. Tolias, A. Bursuc, T. Furon, H. Jégou
Published at: CVIU 2015

Computer Vision and Image Understanding 140 (2015) 9–20

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu

Rotation and translation covariant match kernels for image retrieval☆

Giorgos Tolias*, Andrei Bursuc, Teddy Furon, Hervé Jégou

Inria, Rennes, France

ARTICLE INFO

Article history:
Received 2 February 2015
Accepted 15 June 2015
Available online 23 June 2015

Keywords:
Image retrieval
Geometry aware aggregation
Match kernels
Monomial embedding

ABSTRACT

Most image encodings achieve orientation invariance by aligning the patches to their dominant orientations and translation invariance by completely ignoring patch position or by max-pooling. Albeit successful, such choices introduce too much invariance because they do not guarantee that the patches are rotated or translated consistently. In this paper, we propose a geometric-aware aggregation strategy, which jointly encodes the local descriptors together with their patch dominant angle or location. The geometric attributes are encoded in a continuous manner by leveraging explicit feature maps. Our technique is compatible with generic match kernel formulation and can be employed along with several popular encoding methods, in particular Bag-of-Words, VLAD and the Fisher vector. The method is further combined with an efficient monomial embedding to provide a codebook-free method aggregating local descriptors into a single vector representation. Invariance is achieved by efficient similarity estimation of multiple rotations or translations, offered by a simple trigonometric polynomial. This strategy is effective for image search, as shown by experiments performed on standard benchmarks for image and particular object retrieval, namely Holidays and Oxford buildings.

1. Introduction

This paper considers the problem of particular image or particular object retrieval. This subject has received a sustained attention over the last decade. Many of the recent works employ local descriptors such as SIFT [1] or variants [2,3] for the low-level description of the images. In particular, approaches derived from the Bag-of-Words framework [4] are especially successful to solve problems like recognizing buildings. They are typically combined with spatial verification [5,6] or other re-ranking strategies such as query expansion [7,8].

Our objective is to improve the quality of the first retrieval stage, before any re-ranking is performed. This is critical when considering large datasets, as re-ranking methods depend on the quality of the initial short-list, which typically consists of a few hundred images. The initial stage is improved by better matching rules, for instance with Hamming embedding [9], by learning a fine vocabulary [10], or weighting the distances [11,12]. Moreover, it is useful to employ some geometrical information associated with the region of interest [9]. All these approaches rely on matching individual descriptors and therefore store some data on a per descriptor basis. Moreover, the quantization of the query's descriptors on a large vocabulary causes delays.

Recently, very short yet effective representations have been proposed based on alternative encoding strategies, such as local linear coding [13], the Fisher vector [14] or VLAD [15]. Most of these representations have been proposed first for image classification, yet also offer very effective properties in the context of extremely large-scale image search. A feature of utmost importance is that they offer vector representations compatible with cosine similarity. The representation can then be effectively binarized [16] with cosine sketches, such as those proposed by Charikar [17] (also known as LSH), or aggressively compressed to very short vectors with principal component dimensionality reduction (PCA). Product quantization [18] is another example achieving a very compact representation of a few dozens to hundreds of bytes as well as an efficient search because the comparison is done in the compressed domain.

This paper focuses on such short- and mid-sized vector representations of images. Our objective is to exploit some geometrical information associated with the regions of interest. A popular work in this context is the spatial pyramid kernel [19], which is widely adopted for image classification. However, it is ineffective for particular image retrieval as the grid is too rigid and the resulting representation is not invariant enough, as shown by Douze et al. [20].

Here, we aim at incorporating some relative angle information to ensure that the patches are consistently rotated. In other terms, we want to achieve a covariant property similar to that offered by Weak Geometry Consistency (WGC) [9], but directly implemented into the coding stage of image vector representations like Fisher, or VLAD. We achieve that by jointly encoding the local descriptor with the

☆ This paper has been recommended for acceptance by Barbara Caputo.

* Corresponding author.
E-mail address: giorgos.tolias@inria.fr (G. Tolias).

http://dx.doi.org/10.1016/j.cviu.2015.06.007
1077-3142/

dominant angle in a continuous way. Some recent works in classification [21] and image search [22] consider a similar objective and proceed by rotation quantization. Encoding of such a rough approximation is not straightforwardly compatible with generic match kernels.

In contrast, we achieve the covariant property for any method provided that it can be written as a match kernel. This holds for the Fisher vector, LLC, Bag-of-Words and efficient match kernels listed in [23]. Our method initially assumes aligned objects and image similarity is computed efficiently for multiple rotations thanks to simple trigonometric identities. Finally, the same methodology yields a continuous alternative to spatial pyramid match kernel by encoding patch positions.

This work is the continuation of our previous work [24]. The new contribution consists of the extension to the translation covariant match kernel and the exploitation of a trigonometric polynomial for efficient similarity computation. The latter was only discussed in our previous work, but not exploited.

This paper is organized as follows. Section 2 discusses related works, while Section 3 introduces notation for generic match kernels. Our approach is presented in Section 4 and Section 5 describes the extension to position-translation. Evaluation is presented in Section 6 on several popular benchmarks for image search, namely Oxford5k [5], Oxford105k and Inria Holidays [25]. These experiments show that our approach gives a significant improvement over the state of the art on image search with vector representations. Interestingly, we further achieve competitive results by combining our approach with monomial embeddings, *i.e.*, with a codebook-free approach, as opposed to coding approaches like VLAD.[1]

2. Related work

Our method is inspired by the kernel descriptor of Bo et al. [26] but it departs from this in several ways. First, we are interested in aggregating local descriptors to produce a vector image representation, whereas they construct new local descriptors. Our objective is not to encode the pixel gradient orientation but to achieve the property that the patch representation is covariant. Therefore, we encode the dominant orientation or the spatial coordinates of the region of interest jointly with the corresponding SIFT descriptor. Finally, we rely on explicit feature maps [27] to encode the angle, which provides a much better approximation than efficient match kernel [23] for a given number of components.

The well known aggregated representations, such as Bag-of-Words, VLAD and Fisher vectors, only encode appearance and completely discard spatial information. The most popular attempt surpassing this limitation is the spatial pyramid match [19] (SPM). Patch position is quantized and used as a pooling variable. In this fashion, invariance to any geometric transformation is lost, and only a restricted amount of tolerance is attained.

Regarding position encoding, Arandjelovic and Zisserman [28] extract multiple VLAD descriptors per image from horizontal and vertical tiles, aiming at localizing the searched object in the image. This approach is effective for retrieving small objects, but does not solve the aforementioned shortcomings of image level aggregated descriptors. Recent works in image classification [29–31] provide lower-dimensional alternatives for SPM *via* different encodings of spatial information. Krapac et al. [29] define a Fisher Kernel integrating location prior. Both appearance and spatial layout of patches are encoded. Spatial Coordinate Coding [30] augments the SIFT descriptors with the corresponding spatial coordinates. Quantization, encoding and pooling take place in the augmented feature space. In a similar work [31], feature scale is encoded in addition to position, leading to encouraging results on PASCAL VOC and ImageNet fine-grained classification benchmarks [32].

The hierarchical kernel descriptor of Bo et al. [33] encodes position information at multiple levels. Patch location proximity is evaluated *via* a Gaussian kernel. In order to keep the representation compact, the positions are expressed as projections on 25 basis vectors uniformly sampled from a 5×5 grid. In contrast, we encode positions in a continuous manner, leading to a richer representation and to reduced quantization artifacts.

Following a similar principle to that of SPM but at a single level, the dominant angle is quantized and considered as a pooling variable in recent works [21,22]. CVLAD [22] in particular shifts the subvectors corresponding to each angular cell in order to mimic query image rotation and provides some rotation invariance. This strategy increases complexity proportionally to the number of rotations taken into account. In contrast, along with our method we propose a very efficient way to compute similarity for multiple image rotations.

WGC applies geometric constraints on the whole database of images, and not only on a short-list [5,6]. We achieve the same property with aggregated representations; thus individual descriptors are not indexed. Moreover, we do not need to explicitly form patch correspondences and compute relative angles for each.

3. Background: match kernels and embeddings

We consider the context of match kernels. An image is typically described by a set of local descriptors $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_i, \dots\}$, $\mathbf{x}_i \in \mathbb{R}^d$, $\|\mathbf{x}_i\| = 1$. Similar to other works [9,23,34], two images described by $\mathcal{X}$ and $\mathcal{Y}$ are compared with a match kernel K of the form

$$K(\mathcal{X}, \mathcal{Y}) = \beta(\mathcal{X})\beta(\mathcal{Y}) \sum_{\mathbf{x}\in\mathcal{X}} \sum_{\mathbf{y}\in\mathcal{Y}} k(\mathbf{x}, \mathbf{y}), \tag{1}$$

where k is referred to as the local kernel and where the proportionality factor β ensures that $K(\mathcal{X}, \mathcal{X}) = K(\mathcal{Y}, \mathcal{Y}) = 1$. A convenient way to obtain such a kernel is to map the vectors $\mathbf{x}$ to a higher-dimensional space with a function $\varphi : \mathbb{R}^d \to \mathbb{R}^D$, such that the inner product similarity evaluates the local kernel $k(\mathbf{x}, \mathbf{y}) = \langle \varphi(\mathbf{x}) | \varphi(\mathbf{y}) \rangle$. This approach then represents a set of local descriptors by a single vector

$$\mathbf{X} = \beta(\mathcal{X}) \sum_{\mathbf{x}\in\mathcal{X}} \varphi(\mathbf{x}), \qquad \text{(such that } \|\mathbf{X}\| = 1\text{)} \tag{2}$$

because the match kernel is computed with a simple inner product as

$$K(\mathcal{X}, \mathcal{Y}) = \beta(\mathcal{X})\beta(\mathcal{Y}) \sum_{\mathbf{x}\in\mathcal{X}} \sum_{\mathbf{y}\in\mathcal{Y}} \langle \varphi(\mathbf{x}) | \varphi(\mathbf{y}) \rangle = \langle \mathbf{X} | \mathbf{Y} \rangle. \tag{3}$$

This kernelized view encompasses many approaches for aggregating local image descriptors such as Bag-of-Words [4,35], LLC [13], Fisher vector [14], VLAD [15], or VLAT [36]. A desirable property of k is to have $k(\mathbf{x}, \mathbf{y}) \approx 0$ for unrelated features, so that they do not interfere with the measurements between the true matches. It is somehow satisfied with the classical inner product $k(\mathbf{x}, \mathbf{y}) = \langle \mathbf{x} | \mathbf{y} \rangle$. Several authors [12,34,36,56] propose to increase the contrast between related and unrelated features with a monomial match kernel of degree p of the form

$$K(\mathcal{X}, \mathcal{Y}) = \beta(\mathcal{X})\beta(\mathcal{Y}) \sum_{\mathbf{x}\in\mathcal{X}} \sum_{\mathbf{y}\in\mathcal{Y}} \langle \mathbf{x} | \mathbf{y} \rangle^p. \tag{4}$$

All monomial (and polynomial) embeddings admit exact finite-dimensional feature maps whose length rapidly increases with degree p (in $\mathcal{O}(d^p/p!)$). The order $p = 2$ has already demonstrated some benefit, for instance in semantic segmentation [37] or in image classification [36]. In this case, the kernel is equivalent to comparing the

[1] Code is available online at: https://gforge.inria.fr/frs/download.php/latestzip/4895/PkgAngularmodulation-latest.zip .

set of features based on their covariance matrix [36]. Equivalently, by observing that some components are identical, we can define the embedding $\varphi_2 : \mathbb{R}^d \to \mathbb{R}^{d(d+1)/2}$ mapping $\mathbf{x} = [x_1, \dots, x_d]^\top$ to

$$\varphi_2(\mathbf{x}) = [x_1^2, \dots, x_d^2, x_1 x_2\sqrt{2}, \dots, x_{d-1}x_d\sqrt{2}]^\top. \quad (5)$$

Similarly, the simplified exact monomial embedding associated with $p = 3$ is the function $\varphi_3 : \mathbb{R}^d \to \mathbb{R}^{(d^3+3d^2+2d)/6}$ defined as

$$\varphi_3(\mathbf{x}) = [x_1^3, \dots, x_d^3, x_1^2 x_2\sqrt{3}, \dots, x_d^2 x_{d-1}\sqrt{3}, x_1 x_2 x_3\sqrt{6}, \dots, x_{d-2}x_{d-1}x_d\sqrt{6}]^\top. \quad (6)$$

4. Covariant aggregation of local descriptors

The core idea of the proposed method is to exploit jointly the SIFT descriptors and the dominant orientation θ_x associated with a region of interest. For this purpose, we now assume that an image is represented by a set $\mathcal{X}^\star$ of tuples, each of the form $(\mathbf{x}, \theta_x)$, where $\mathbf{x}$ is a SIFT descriptor and $\theta_x \in [-\pi, \pi]$ is the dominant orientation. Our objective is to obtain an approximation of a match kernel of the form

$$K^\star(\mathcal{X}^\star, \mathcal{Y}^\star) = \beta(\mathcal{X}^\star)\beta(\mathcal{Y}^\star) \sum_{\substack{(\mathbf{x},\theta_x)\in\mathcal{X}^\star \\ (\mathbf{y},\theta_y)\in\mathcal{Y}^\star}} k(\mathbf{x},\mathbf{y})\, k_\theta(\theta_x, \theta_y) \quad (7)$$

$$= \langle \mathbf{X}^\star | \mathbf{Y}^\star \rangle, \quad (8)$$

where k is a local kernel identical to that considered in Section 2 and k_θ reflects the similarity between angles. The interest of enriching this match kernel with orientation is illustrated in Fig. 1, where we show that several incorrect matches are downweighted thanks to this information.

The kernel in (7) resembles that implemented in WGC [9] with a voting approach. In contrast, we intend to approximate this kernel with an inner product between two vectors as in (8), similar to the linear match kernel simplification in (3). Our work is inspired by the kernel descriptors [26] of Bo et al., who also consider a kernel of a similar form, but at the patch level, to construct a local descriptor from pixel attributes, such as gradient and position.

In our case, we consider the coding stage and employ a better approximation technique, namely explicit feature maps [27], to encode $\mathcal{X}^\star$. This section first explains the feature map of the angle, then described how the descriptors and angles are jointly represented, and finally discusses the match kernel design and properties.

4.1. A feature map for the angle

The first step is to find a mapping $\boldsymbol{\alpha} : [-\pi, \pi] \to \mathbb{R}^M$ from an angle θ to a vector $\boldsymbol{\alpha}(\theta)$ such that $\boldsymbol{\alpha}(\theta_1)^\top \boldsymbol{\alpha}(\theta_2) = k_\theta(\theta_1 - \theta_2)$. The function $k_\theta : \mathbb{R} \to [0, 1]$ is a shift invariant kernel which should be symmetric ($k_\theta(\Delta\theta) = k_\theta(-\Delta\theta)$), pseudo-periodic with period of 2π and monotonically decreasing over $[0, \pi]$. The function k_θ is scalar and it allows us to model the behaviour of the match kernel and to design the feature map accordingly. In the work of Vedaldi and Zisserman [27] it is termed as kernel signature. We consider in particular the following function:

$$k_{\mathsf{VM}}(\Delta\theta) = \frac{\exp(\kappa\cos(\Delta\theta)) - \exp(-\kappa)}{2\sinh(\kappa)}. \quad (9)$$

It is derived from Von Mises distribution $f(\Delta\theta; \kappa)$, which is often considered as the probability density distribution of the noise of the measure of an angle, and therefore regarded as the equivalent Gaussian distribution for angles. Our function k_{VM} is a shifted and scaled variant of Von Mises, designed such that its range is $[0, 1]$, which ensures that $k_{\mathsf{VM}}(\pi) = 0$.

The periodic function k_{VM} can be expressed as a Fourier series whose coefficients are (see [38, Eq. (9.6.19)]

$$k_{\mathsf{VM}}(\Delta\theta) = \frac{I_0(\kappa) - e^{-\kappa} + 2\sum_{n=1}^{\infty} I_n(\kappa)\cos(n\Delta\theta)}{2\sinh(\kappa)}, \quad (10)$$

where $I_n(\kappa)$ is the modified Bessel function of the first kind of order n. We now consider the truncation $\bar{k}^N_{\mathsf{VM}}$ of the series to the first N terms

$$\bar{k}^N_{\mathsf{VM}}(\Delta\theta) = \sum_{n=0}^{N} \gamma_n \cos(n\Delta\theta) \quad (11)$$

$$\text{with} \quad \gamma_0 = \frac{(I_0(\kappa) - e^{-\kappa})}{2\sinh(\kappa)} \quad \text{and} \quad \gamma_n = \frac{I_n(\kappa)}{\sinh(\kappa)} \quad \text{if } n > 0. \quad (12)$$

We design the feature map $\boldsymbol{\alpha}(\theta)$, mappping an angle θ to a vector, as follows:

$$\boldsymbol{\alpha}(\theta) = (\sqrt{\gamma_0}, \sqrt{\gamma_1}\cos(\theta), \sqrt{\gamma_1}\sin(\theta), \dots, \sqrt{\gamma_N}\cos(N\theta), \sqrt{\gamma_N}\sin(N\theta))^\top. \quad (13)$$

This vector has $2N + 1$ components. Moreover

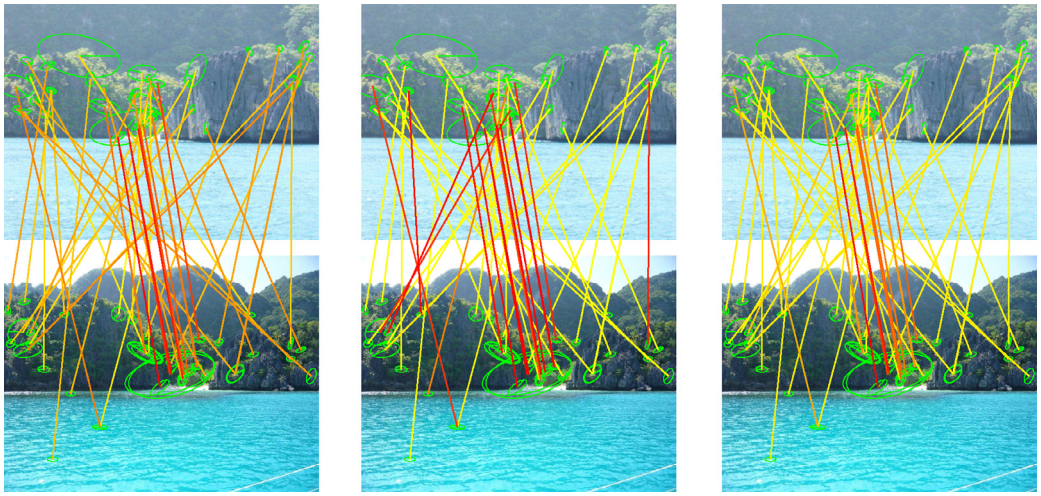


Fig. 1. Similarities between regions of interest, based on SIFT kernel k (left), angle consistency kernel k_θ (middle) and both (right). For each local region, we visualize the values $k(\mathbf{x}, \mathbf{y})$, $k_\theta(\Delta\theta)$ and their product by the colors of the link (red=1). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

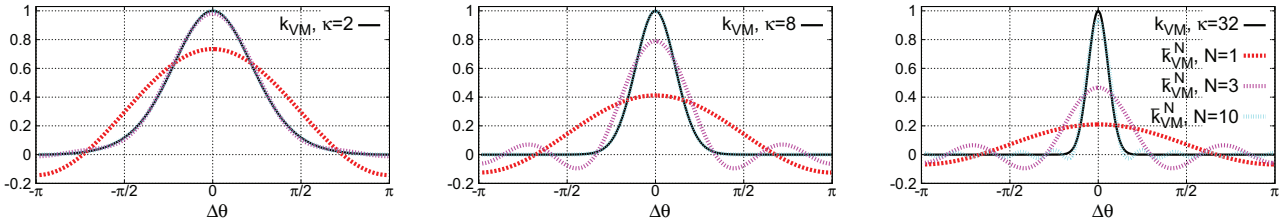


Fig. 2. Function $k_{\mathbf{VM}}$ for different values of κ and its approximation $\bar{k}^N_{\mathbf{VM}}$ using 1, 3 and 10 frequencies, as implicitly defined by the corresponding mapping $\boldsymbol{\alpha} : [\pi, \pi] \rightarrow \mathbb{R}^{2N+1}$.

$$\begin{aligned}\boldsymbol{\alpha}(\theta_1)^\top \boldsymbol{\alpha}(\theta_2) &= \gamma_0 + \sum_{n=1}^{N} \gamma_n (\cos(n\theta_1)\cos(n\theta_2) \\ &\quad + \sin(n\theta_1)\sin(n\theta_2)) \\ &= \sum_{n=0}^{N} \gamma_n \cos(n(\theta_1 - \theta_2)) \\ &= \bar{k}^N_{\mathbf{VM}}(\theta_1 - \theta_2) \approx k_{\mathbf{VM}}(\theta_1 - \theta_2) \end{aligned} \tag{14}$$

The design of a feature map is explained in full details by Vedaldi and Zisserman [27]. This feature map gives an approximation of the target function $k_{\mathbf{VM}}$, which is more accurate as N is bigger.

Fig. 2 illustrates the function $k_{\mathbf{VM}}$ for several values of the parameter κ and its approximation $\bar{k}^N_{\mathbf{VM}}$ for different values of N. First note that $\bar{k}^N_{\mathbf{VM}}$ may not fulfill the original requirements: its range might be wider than [0, 1] and it might not be monotonically decreasing over $[0, \pi]$. Larger values of κ produce a more "selective" function of the angle, yet require more components (larger value of N) to obtain an accurate estimation. Importantly, the approximation stemming from this explicit angle mapping is better than that based on efficient match kernels [23], which converges slowly with the number of components. Efficient match kernels are more intended to approximate kernels on vectors than on scalar values. As a trade-off between selectivity and the number of components, we set κ=8 and N=3 (see Section 6). Accordingly, we use $\bar{k}^3_{\mathbf{VM}}$ as k_θ in the sequel. The corresponding embedding $\boldsymbol{\alpha} : \mathbb{R} \rightarrow \mathbb{R}^7$ maps any angle to a 7-dimensional vector.

Exact estimation of kernel signature. Instead of approximating a kernel on angles with finite Fourier series, one may rather consider directly designing a function satisfying our initial requirements (pseudo-period, symmetric, decreasing over $[0, \pi]$), such as

$$k_P(\Delta\theta) = \cos(\Delta\theta/2)^P \text{ with } P \text{ even}. \tag{15}$$

Thanks to power reduction trigonometric identities, for even P this function is re-written as

$$k_P(\Delta\theta) = \sum_{p=0}^{P/2} \gamma_p \cos(p\Delta\theta) \tag{16}$$

with

$$\gamma_0 = \frac{1}{2^P}\binom{P}{P/2}, \gamma_p = \frac{1}{2^{P-1}}\binom{P}{P/2-p} \quad 0 < p \leq P/2. \tag{17}$$

Now, applying (13) leads to a feature map $\boldsymbol{\alpha}(\theta)$ with $P+1$ components such that $\boldsymbol{\alpha}(\theta_1)^\top \boldsymbol{\alpha}(\theta_2) = k_P(\theta_1 - \theta_2)$. For this function, the interesting property is that the scalar product is exactly equal to the target kernel value $k_P(\theta_1 - \theta_2)$, and that the original requirements now hold. From our experiments, this function gives reasonable results, but requires more components than $\bar{k}_{\mathbf{VM}}$ to achieve a shape narrow around $\Delta\theta = 0$ and close to 0 otherwise. The results for our image search application task using this function are slightly below our Von Mises variant for a given dimensionality. So, despite its theoretical interest we do not use it in our experiments. Ultimately, one would rather directly learn a Fourier embedding for a targeted task (*e.g.* an embedding per classifier), in the spirit of Fourier kernel learning [39].

4.2. Modulation and covariant match kernel

The vector $\boldsymbol{\alpha}$ encoding the angle θ "modulates"[2] any vector $\mathbf{x}$ (or pre-mapped descriptor $\varphi(\mathbf{x})$) with a function $m : \mathbb{R}^{2N+1} \times \mathbb{R}^D \rightarrow \mathbb{R}^{(2N+1)D}$. Thanks to classical properties of the Kronecker product $\otimes$, we have

$$\begin{aligned} m(\mathbf{x}, \boldsymbol{\alpha}(\theta)) &= \mathbf{x} \otimes \boldsymbol{\alpha}(\theta) \\ &= (x_1\boldsymbol{\alpha}(\theta)^\top, x_2\boldsymbol{\alpha}(\theta)^\top \ldots, x_d\boldsymbol{\alpha}(\theta)^\top)^\top. \end{aligned} \tag{18}$$

We now consider two pairs of vectors and angle, $(\mathbf{x}, \theta_x)$ and $(\mathbf{y}, \theta_y)$, and their modulated descriptors $m(\mathbf{x}, \boldsymbol{\alpha}(\theta_x))$ and $m(\mathbf{y}, \boldsymbol{\alpha}(\theta_y))$, respectively. In the inner product space $\mathbb{R}^{(2N+1)D}$, the following holds:

$$\begin{aligned} m(\mathbf{x}, \boldsymbol{\alpha}(\theta_x))^\top m(\mathbf{y}, \boldsymbol{\alpha}(\theta_y) &= (\mathbf{x} \otimes \boldsymbol{\alpha}(\theta_x))^\top (\mathbf{y} \otimes \boldsymbol{\alpha}(\theta_y)) \\ &= (\mathbf{x}^\top \otimes \boldsymbol{\alpha}(\theta_x)^\top)(\mathbf{y} \otimes \boldsymbol{\alpha}(\theta_y)) \\ &= (\mathbf{x}^\top \mathbf{y}) \otimes (\boldsymbol{\alpha}(\theta_x)^\top \boldsymbol{\alpha}(\theta_y)) \\ &= (\mathbf{x}^\top \mathbf{y}) k_\theta(\theta_x - \theta_y). \end{aligned} \tag{19}$$

Fig. 3 shows the distribution of the similarities between regions of interest before and after modulation, as a function of the difference of angles. Interestingly, there is no obvious correlation between the difference of angle and the SIFT: the similarity distribution based on SIFT is similar for all angles. This suggests that the modulation with angle provides complementary information.

Combination with coding/pooling techniques. Consider any coding method φ that can be written as match kernel (Fisher, LLC, Bag-of-Words, VLAD, *etc.*). The match kernel in (7), with our k_θ approximation, is re-written as

$$\begin{aligned} K^\star(\mathcal{X}^\star, \mathcal{Y}^\star) &\propto \sum_{\substack{(\mathbf{x},\theta_x)\in\mathcal{X}^\star \\ (\mathbf{y},\theta_y)\in\mathcal{Y}^\star}} m(\varphi(\mathbf{x}), \boldsymbol{\alpha}(\theta_x))^\top m(\varphi(\mathbf{y}), \boldsymbol{\alpha}(\theta_y)) \\ &\propto \sum_{(\mathbf{x},\theta_x)} m(\varphi(\mathbf{x}), \boldsymbol{\alpha}(\theta_x))^\top \sum_{(\mathbf{y},\theta_y)} m(\varphi(\mathbf{y}), \boldsymbol{\alpha}(\theta_y)), \end{aligned} \tag{20}$$

where we observe that the image can be represented as the summation $\mathbf{X}^\star$ of the embedded descriptors modulated by their corresponding dominant orientation, as

$$\mathbf{X}^\star = \beta(\mathcal{X}^\star) \sum_{(\mathbf{x},\theta_x)\in\mathcal{X}^\star} m(\varphi(\mathbf{x}), \boldsymbol{\alpha}(\theta_x)). \tag{21}$$

This representation encodes the relative angles and is already more discriminative than an aggregation that does not consider them. However, at this stage, the comparison assumes that the images have the same global orientation. This is the case on benchmarks like Oxford5k building, where all images are orientated upright, but this is not true in general for particular object recognition.

[2] By analogy to communications, where modulation refers to the process of encoding information over periodic waveforms.

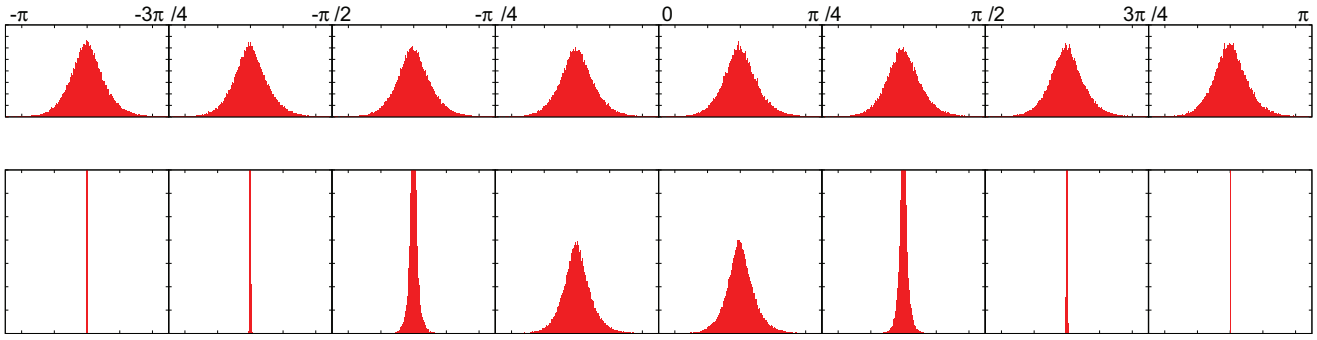


Fig. 3. Distribution of patch similarity for different values of orientation difference. In this figure, we split the angular space into 8 equally-sized bins and present the similarity distribution separately for each of these bins. Horizontal axis represents the similarity value between matching features. Top: distribution of similarities with kernel on SIFTs. Bottom: distribution after modulation with α.

4.3. Rotation invariance

So far our image representation is rotation covariant. Herein we propose how to achieve rotation invariance *via* efficient similarity computation for multiple image rotations. Up to now we have assumed that objects are aligned with respect to orientation or, more particularly, that objects are up-right. This implies that true corresponding patches should have similar orientation. We now describe how to produce a similarity score when the orientations of related images may be different. We represent the image vector $\mathbf{X}^\star$ as the concatenation of $2N+1$ D-dimensional subvectors associated to one term of the finite Fourier series: $\mathbf{X}^\star = [\mathbf{X}_0^{\star\top}, \mathbf{X}_{1,c}^{\star\top}, \mathbf{X}_{1,s}^{\star\top}, \ldots, \mathbf{X}_{N,c}^{\star\top}, \mathbf{X}_{N,s}^{\star\top}]^\top$. The vector $\mathbf{X}_0^\star$ is associated with the constant term in the Fourier expansion, $\mathbf{X}_{n,c}^\star$ and $\mathbf{X}_{n,s}^\star$, $1 \leq n \leq N$, correspond to the cosine and sine terms, respectively.

Imagine now that this image undergoes a global rotation of angle θ. Denote $\breve{\mathcal{X}}^\star$ the new set of pairs $(\mathbf{x}, \breve{\theta}_x)$. Since descriptor $\mathbf{x}$ is by nature rotation invariant, we obtain $\breve{\mathcal{X}}^\star$ by simply shifting all dominant angles by θ, that is $\breve{\theta}_x = \theta_x - \theta$. Denote $\breve{\mathbf{X}}^\star$ the new image vector derived from these local descriptors. It occurs that $\breve{\mathbf{X}}_0^\star = \mathbf{X}_0^\star$ because this term does not depend on the angle, and that, for a given frequency bin n, elementary trigonometry identities lead to

$$\breve{\mathbf{X}}_{n,c}^\star = \mathbf{X}_{n,c}^\star \cos n\theta + \mathbf{X}_{n,s}^\star \sin n\theta \tag{22}$$

$$\breve{\mathbf{X}}_{n,s}^\star = -\mathbf{X}_{n,c}^\star \sin n\theta + \mathbf{X}_{n,s}^\star \cos n\theta. \tag{23}$$

Therefore, we do not need to recompute the image representation of the rotated image. It is efficiently derived by component wise multiplications of the vector describing the original image. It also turns out that $\|\breve{\mathbf{X}}^\star\| = \|\mathbf{X}^\star\|$, meaning that rotation has no effect on the global normalization factor $\beta(\mathcal{X}^\star)$.

When comparing two images with such vectors, the linearity of the inner product ensures that

$$\langle \breve{\mathbf{X}}^\star | \mathbf{Y}^\star \rangle = \langle \mathbf{X}_0^\star | \mathbf{Y}_0^\star \rangle + \sum_{n=1}^{N} \cos n\theta \left(\langle \mathbf{X}_{n,c}^\star | \mathbf{Y}_{n,c}^\star \rangle + \langle \mathbf{X}_{n,s}^\star | \mathbf{Y}_{n,s}^\star \rangle \right) + \sum_{n=1}^{N} \sin n\theta \left(-\langle \mathbf{X}_{n,c}^\star | \mathbf{Y}_{n,s}^\star \rangle + \langle \mathbf{X}_{n,s}^\star | \mathbf{Y}_{n,c}^\star \rangle \right). \tag{24}$$

Here, we stress that the similarity between two images is a real trigonometric polynomial in θ (image rotation angle) of degree N. Its $2N+1$ components are fully determined by computing $\langle \mathbf{X}_0^\star | \mathbf{Y}_0^\star \rangle$ and the inner products between the subvectors associated with each frequency, *i.e.*, $\langle \mathbf{X}_{n,c}^\star | \mathbf{Y}_{n,c}^\star \rangle$, $\langle \mathbf{X}_{n,s}^\star | \mathbf{Y}_{n,s}^\star \rangle$, $\langle \mathbf{X}_{n,c}^\star | \mathbf{Y}_{n,s}^\star \rangle$ and $\langle \mathbf{X}_{n,s}^\star | \mathbf{Y}_{n,c}^\star \rangle$. Finding the maximum of this polynomial amounts to finding the rotation maximizing the image similarity.

Computing the coefficients of this polynomial requires a total of $D \times (1+4N)$ elementary operations for a vector representation of dimensionality $D \times (1+2N)$, that is, less than twice the cost of the inner product between $\mathbf{X}^\star$ and $\mathbf{Y}^\star$. Once these components are obtained, the cost of finding the maximum value achieved by this polynomial is negligible for large values of D, for instance by simply sampling a few values of θ. Therefore, if we offer the orientation invariant property, the complexity of similarity computation is typically twice the cost of that of a regular vector representation (whose complexity is equal to the number of dimensions).

Our trigonometric polynomial of similarity scores can be rewritten as

$$K^\star(\mathcal{X}^\star, \mathcal{Y}^\star, \theta) = c + \sum_{n=1}^{N} a_n \cos n\theta + \sum_{n=1}^{N} b_n \sin n\theta, \tag{25}$$

with coefficients c, a_n, b_n given by (24). Note that in our experiment it turns out that retrieval performance already saturates at $N = 3$.

This strategy for computing the scores for all possible orientations of the query is not directly compatible with non-linear post-processing of $\mathbf{X}^\star$ such as component-wise power-law normalization [40], except for the subvector $\mathbf{X}_0^\star$. We adapt the power-law normalization to become compatible with our strategy: we compute the modulus of the complex number represented by two components (sin and cos) associated with the same frequency n and the same original component in $\varphi(\mathbf{x})$. These two components are then divided by the square-root (or any power) of this modulus.

In detail, let $\mathbf{X}_{n,c,i}^\star$ and $\mathbf{X}_{n,s,i}^\star$ be the ith component of subvectors $\mathbf{X}_{n,c}^\star$ and $\mathbf{X}_{n,s}^\star$, respectively. The modified scheme considers the modulus of those components. The power-law normalized version of the former turns out to be equal to $\frac{\mathbf{X}_{n,c,i}^\star}{(\mathbf{X}_{n,c,i}^{\star 2} + \mathbf{X}_{n,s,i}^{\star 2})^{\frac{1-l}{2}}}$, where $l \in [0,1]$ is the power-law exponent. The counterpart sine component is obtained similarly, and now the representation of the rotated image is factorized equivalently to (22) and (23).

In our experiments we reduce the dimensionality of the modulated image vector by PCA, as typically done with aggregated representations. It is then not possible to use the efficient polynomial of scores. In this case, we follow the naive strategy, which is to compute the query representation for several hypotheses of angle rotation, typically 8. In theory, this multiplies the query complexity by the same factor 8. However, in practice, it is faster to perform the matrix–matrix multiplication, with the right matrix representing 8 queries, than computing separately the corresponding 8 matrix–vector multiplications. In our former work [24], the naive approach was used in all cases, while now we explore the proposed polynomial on the full vectors.

Fig. 4 presents the evaluation of (25) for a pair of images. More frequencies improve the approximation. However, maximum similarity value is observed at a similar point in all cases.

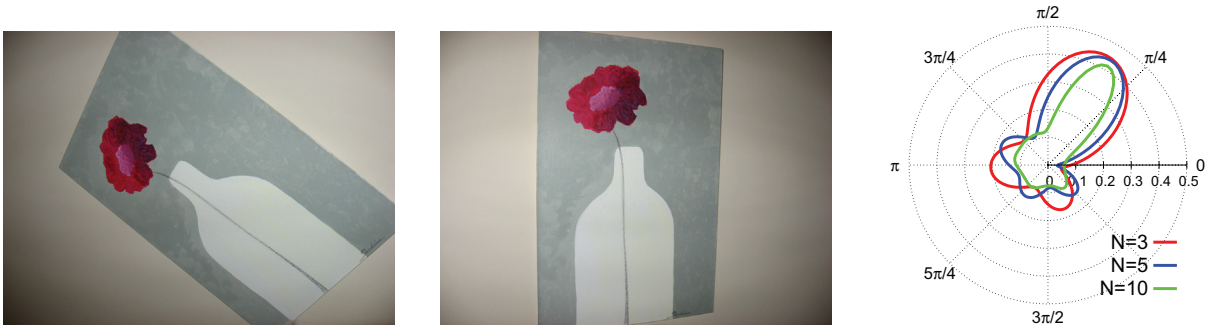


Fig. 4. Matching example of two images and similarity estimation for all possible image rotations. The image on the left undergoes rotation. Image similarity *versus* rotation is shown in polar coordinates, with the angular direction corresponding to the image rotation, and the radial to the image similarity score. This example is computed with angular modulation of VLAD, while using 3, 5 or 10 frequencies.

5. Translation covariant aggregation

5.1. Encoding the position

Following the objective of jointly encoding local description and geometry, we now deal with the location of local features. We assume that an image is represented by a set $\mathcal{X}^\diamond$ of triples of the form $(\mathbf{x}, u_x, v_x)$. Local descriptor $\mathbf{x}$ is now accompanied by position coordinates u_x and v_x.

Depending on the use-case and on the desired invariance, u_x and v_x can be cartesian or polar coordinates. Whatever the coordinate system is, our position encoding is performed in a continuous manner, unlike SPM [19] and *visual phrases* [41], where positions, respectively position differences, are quantized to a uniform grid. In the following we consider cartesian coordinates.

We employ once more the angular embedding proposed in Section 4.1, by mapping a spatial coordinate to an angle. The kernel function k_θ defined for angles is periodic, while the spatial coordinates of a local feature are not. Mapping positions directly to $[-\pi, \pi]$ would practically convert the position domain into a torus. In such a case, patches located at opposite edges of the image would be considered close to each other. We simply handle this by mapping to $[-\pi/2, \pi/2]$. Still, when employing multiple translations this undesired effect emerges, but it does not seem to harm the effectiveness of the method in our experiments. Therefore, we convert the position coordinates to angles by

$$\hat{u} = \frac{u - \frac{h}{2}}{\max\{h, w\}}\,\pi\,,\quad \hat{v} = \frac{v - \frac{w}{2}}{\max\{h, w\}}\,\pi\,, \tag{26}$$

where h and w are the height and width of the image, respectively.

We now employ the procedure of Section 4. We map local coordinates u_x and v_x to $\boldsymbol{\alpha}(\hat{u}_x)$ and $\boldsymbol{\alpha}(\hat{v}_x)$, respectively. Then, each one of them can be encoded jointly with descriptor $\mathbf{x}$ by $m(\mathbf{x}, \boldsymbol{\alpha}(\hat{u}_x))$ and $m(\mathbf{x}, \boldsymbol{\alpha}(\hat{v}_x))$. We obtain two new match kernels $K_u(\mathcal{X}^\diamond, \mathcal{Y}^\diamond)$ and $K_v(\mathcal{X}^\diamond, \mathcal{Y}^\diamond)$, down-weighting or up-weighting matches by the consistency of their u or v coordinate, respectively.

We are not limited to modulation only by u or v (single modulation), we can further encode both coordinates by a double modulation. This is achieved by modulating descriptor $\mathbf{x}$ by both $\boldsymbol{\alpha}(\hat{u}_x)$ and $\boldsymbol{\alpha}(\hat{v}_x)$ with a function $m_{u,v} : \mathbb{R}^{2N+1} \times \mathbb{R}^{2N+1} \times \mathbb{R}^D \to \mathbb{R}^{(2N+1)^2 D}$, where

$$m_{u,v}(\mathbf{x}, \boldsymbol{\alpha}(\hat{u}_x), \boldsymbol{\alpha}(\hat{v}_x)) = \mathbf{x} \otimes \boldsymbol{\alpha}(\hat{u}_x) \otimes \boldsymbol{\alpha}(\hat{v}_x). \tag{27}$$

The match kernel for two sets of local descriptors can be then written as

$$\begin{aligned} K_{u,v}(\mathcal{X}^\diamond, \mathcal{Y}^\diamond) &\propto \sum_{\substack{(\mathbf{x}, u_x, v_x) \in \mathcal{X}^\diamond \\ (\mathbf{y}, u_y, v_y) \in \mathcal{Y}^\diamond}} k(\mathbf{x}, \mathbf{y}) k_\theta(\hat{u}_x, \hat{u}_y) k_\theta(\hat{v}_x, \hat{v}_y) \\ &\propto \langle \mathbf{X}^\diamond | \mathbf{Y}^\diamond \rangle, \end{aligned} \tag{28}$$

where each image is represented by vector $\mathbf{X}^\diamond$. This is the vector of aggregated double modulated local descriptors

$$\mathbf{X}^\diamond = \beta(\mathcal{X}^\diamond) \sum_{(\mathbf{x}, u_x, v_x) \in \mathcal{X}^\diamond} m_{u,v}(\mathbf{x}, \boldsymbol{\alpha}(\hat{u}_x), \boldsymbol{\alpha}(\hat{v}_x)). \tag{29}$$

Fig. 5 illustrates the function $k_{\mathsf{VM}}(\Delta\hat{u}) \times k_{\mathsf{VM}}(\Delta\hat{v})$ along with it approximation for 2 and 3 frequencies. We want a "selective" kernel function in order to weight up pairs of similar patches placed at similar locations. The double modulation increases dimensionality too fast with respect to the number of frequencies. As a trade-off between selectivity and the number of components we set κ=8 and N=2. In this case, coordinates (u, v) are mapped to a 25 dimensional vector.

5.2. Translation invariance

We have assumed, up to now, that objects are aligned. Next, we follow the same approach proposed for dominant orientation, in order to offer translation invariance. Note that rotation and scale invariance are lost in this case, but there is some tolerance introduced by the continuous encoding and by the employed similarity function.

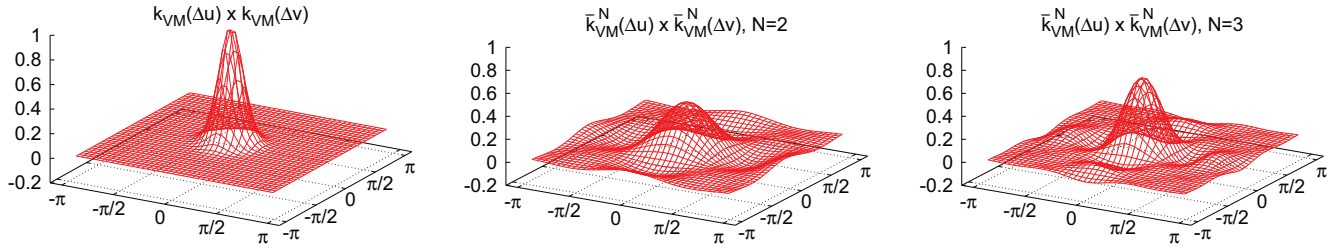


Fig. 5. Function $k_{\mathbf{VM}}(\Delta\theta_u) \times k_{\mathbf{VM}}(\Delta\theta_v)$ for $\kappa = 8$ and its approximation $\bar{k}^N_{\mathbf{VM}}(\Delta\theta_u) \times \bar{k}^N_{\mathbf{VM}}(\Delta\theta_v)$ using 2 and 3 frequencies, as implicitly defined by the corresponding mapping $\boldsymbol{\alpha} : [\pi, \pi] \to \mathbb{R}^{2N+1}$. The higher the number of frequencies, the better the approximation of the signature.

Fig. 6. Matching example of two images and similarity estimation for all possible image translations. Image on the left undergoes translation independently to the horizontal and the vertical direction. Modulated VLAD is used for the example.

In the case of single modulation of u or v, we are able to efficiently evaluate for multiple 1D translations with the same trigonometric polynomial introduced before (25). Detecting maximum similarity aligns objects with respect to u or v, independently. An example is shown in Fig. 6. Another choice is to maximize independently and to keep best alignment of both. This is achieved by simply keeping the maximum similarity score of the two.

One step further, we allow for 2D translation along with the double modulation by u and v. Factorization similar to that of (22)–(23) is still possible, resulting into another trigonometric polynomial for efficient 2D translation

$$\begin{aligned} K_{u,v}(\mathcal{X}^\diamond, \mathcal{Y}^\diamond, \hat{u}, \hat{v}) = a^0 &+ \sum_{n=1}^{N} a_n^1 \cos n\hat{u} + \sum_{n=1}^{N} a_n^2 \sin n\hat{u} \\ &+ \sum_{t=1}^{N} a_t^3 \cos t\hat{v} + \sum_{t=1}^{N} a_t^4 \sin t\hat{v} \\ &+ \sum_{n=1}^{N}\sum_{t=1}^{N} a_{n,t}^5 \cos n\hat{u} \cos t\hat{v} \\ &+ \sum_{n=1}^{N}\sum_{t=1}^{N} a_{n,t}^6 \cos n\hat{u} \sin t\hat{v} \\ &+ \sum_{n=1}^{N}\sum_{t=1}^{N} a_{n,t}^7 \sin n\hat{u} \cos t\hat{v} \\ &+ \sum_{n=1}^{N}\sum_{t=1}^{N} a_{n,t}^8 \sin n\hat{u} \sin t\hat{v}. \end{aligned} \tag{30}$$

The image translation is denoted by $(\hat{u}, \hat{v})$. Coefficients $a^0 \dots a^8$ are given by inner products of particular sub-vectors of the two image representation vectors. We skip the details which are in analogy to those of (24).

Its computational cost is, once more, very small comparing to performing the translations in a naive way. Compatibility with power-law normalization is achieved in a similar fashion to that of Section 4, but with groups of 4 components in this case.

Our achievement of fast similarity computation for multiple translations resembles the work of Henriques et al. [42] who speed-up learning with multiple shifted versions of negative samples. They do this instead of performing costly sliding window based hard negative mining. In our case, we obtain the translated models through latent variables, *i.e.* translation on a given direction, parameterizing a trigonometric polynomial of similarity scores.

6. Experiments

Datasets. We evaluate the performance of the proposed approaches and compare with state-of-the-art methods on two publicly available datasets for image and particular object retrieval, namely Inria Holidays [25] and Oxford Buildings 5k [5]. We also combine the latter with 100k distractor images to measure the performance on a larger scale. The merged dataset is referred to as Oxford105k. Performance is measured with mean Average Precision (mAP) [5].

We further employ the rotated Holidays dataset [43], with images rotated to their natural orientation, in order to evaluate our position covariant kernels. This is necessary since this method is not rotation invariant. It is therefore not applicable when object rotations exist. Note that the rotation covariant kernel does not have such limitations. We refer to the upright oriented Holidays dataset as Holidays^.

Our approach modulates any coding/pooling technique operating as a match kernel. Therefore, we evaluate the benefit of our approach combined with several coding techniques, namely

- VLAD [15], which encodes a SIFT descriptor by considering the residual vector to the centroid.
- The Fisher vector [14,40,44]. For image classification, Chatfield et al. [45] show that it outperforms concurrent coding techniques, in particular LLC [13]. We adopt the standard choice for image retrieval and use only the gradient with respect to the mean [15].
- Monomomial embeddings of order 2 and 3 applied on local descriptors (see below for pre-processing), *i.e.*, the functions φ_2 in (5) and φ_3 in (6). For the sake of consistency, we also denote by φ_1 the function $\varphi_1 : x \to x$.

We refer to these methods combined with single modulation with the symbol "$\otimes$": VLAD$\otimes$, Fisher$\otimes$, $\varphi_1\otimes$, $\varphi_2\otimes$ and $\varphi_3\otimes$ for the angle modulation. Single position modulations are denoted by VLAD$\otimes_u$ and VLAD$\otimes_v$ and double modulation by VLAD$\otimes_{u,v}$. The particular case for which we independently encode u and v and keep the maximum score of both is referred as VLAD$\otimes_{u/v}$. The same notation is followed for the Fisher and monomial embeddings.

In addition, we compare against the most related work, namely the recent CVLAD [22] method, which also aims at producing an image vector representation integrating the dominant orientations of the patches. Whenever the prior work is not referenced, results are produced using our own implementations of VLAD, Fisher and CVLAD, so that the results are directly comparable with the same features.

6.1. Implementation details

Local descriptors. We use the Hessian-Affine detector [46] to extract the regions of interest, that are subsequently described by SIFT descriptors [1] post-processed with RootSIFT [47]. Then, we apply PCA and the resulting vector is subsequently ℓ_2-normalized. Following the typical procedure for the Fisher vector [14,15,40], when applying PCA we reduce the vector to 80 components. The same stands for monomial embeddings. An exception is done for VLAD and CVLAD with which we only use the PCA basis to center and rotate descriptors as suggested by Delhumeau [48], without dimensionality reduction.

The optimized Hessian-Affine detector of Perdoch et al. [43] improves the retrieval performance. However, it is not compatible with

our angular encoding (rotation covariant kernel) by discarding rotations and enforcing the gravity vector assumption (up-right features). For the needs of the angular modulation we use the original Hessian-Affine detector [46], but modify it so that it has similar parameters (enlarged measurement region by a factor of 2) and use a lower detector threshold. In addition, we use the detector of Perdoch et al. [43] for evaluating the position encoding. The translation covariant kernel can benefit from the advantages of up-right features when all depicted objects are aligned with respect to rotation. The use of the latter is explicitly stated in each case.

Codebook. For all methods based on codebooks, we only consider distinct datasets for learning. More precisely and following common practice, the k-means and GMM (for VLAD and Fisher, respectively) are learned on Flickr60k for Inria Holidays and Paris6k [49] for Oxford buildings. We rely on the Yael library [50] for codebook construction and VLAD and Fisher encoding.

Post-processing. The final image vector obtained by each method is power-law normalized [11,15,40]. This processing improves the performance by efficiently handling the burstiness phenomenon. Exploiting the dominant orientation in our covariant match kernel provides a complementary way to further handle the same problem. We mention that using the dominant orientation is shown effective in a recent work by Torii et al. [51]. This post-processing, when applied to the modulated vectors, inherently captures and down weights patches with similar orientation or position.

In addition to power-law normalization, we rotate the aggregated vector representation with a learned PCA rotation matrix [52,53]. This aims at capturing the co-occurrences to down-weight them either by whitening [52] or a second power-law normalization [53]. We adopt the latter choice (with exponent 0.5) to avoid the sensitivity to eigenvalues (in whitening) when learning PCA with a few input data. We refer to this Rotation and Normalization [54] as RN.

Optionally, to produce compact representations, we keep only the first few components (the most energetic ones) and ℓ_2-normalize the shortened vector.

Query rotation. In order to obtain rotation invariance jointly with power-law normalization we exploit the trigonometric polynomial presented in Section 4, along with our modified scheme for power-law normalization. Image similarity is evaluated for multiple query image rotations.

Since the aforementioned technique is not compatible with RN or dimensionality reduction, in that case, we follow the naive approach and apply rotations of the query image and perform individual queries.

We apply 8 query rotations on Holidays dataset. On Oxford5k, we rather adopt the common choice of not considering other possible orientations, since images are up-right.

Query translation. Our methods for position encoding are evaluated on the upright Holidays^ dataset. On Oxford5k we follow the standard protocol and use the cropped queries. However, we also consider the position of the bonding box in the image as known, in order to properly normalize the patch coordinates.

Once more, we use the trigonometric polynomial for computation of the image similarity score. In the case of 1D translations, we evaluate 25 possible translations on each direction and a step of 10 pixels, that is $1 + 25 \times 2$ translations in total. While for 2D translations, we evaluate 20 translations per direction leading to a total of $(20 + 20 + 1)^2$ translations. The chosen step is also set to 10 pixels.

6.2. Impact of power-law normalization

In Fig. 7 we present performance for power-law normalization of different exponents. The non-modulated representations appear to have optimal performance around $l = 0.2$, which is in accordance with previous results [48] in the case that the local descriptors are rotated with PCA. The behavior is different for the modulated vectors, where optimal performance appears for $l = 0$. Note that in contrast to the standard power-law normalization scheme, the modified scheme proposed in Section 4.3 does not produce binary vectors for such a choice.

In the rest of our experiments we adopt an exponent equal to 0 and 0.2 for the modulated and non-modulated vectors correspondingly. Such choices are not optimal while learning the PCA rotation matrix for RN or for dimensionality reduction, where we apply square-rooting. Any difference observed in the reported performance compared to our previous work [24] is attributed to an optimal power-law value used in this work and to the modified power-law normalization scheme.

6.3. Impact of the parameters

The impact of the angle modulation is controlled by the function k_θ parametrized by κ and N. As shown in Fig. 2, the value κ typically controls the "bandwidth", *i.e.*, the range of $\Delta\theta$ values with non-zero response. The parameter N controls the quality of the approximation, and implicitly constrains the achievable bandwidth. It also determines the dimensionality of the output vector.

Fig. 8 shows the impact of the selectivity of the kernel signature on the performance. As to be expected, there is a trade-off between defining too narrow or too large. The optimal performance is achieved with κ in the range [2, 8].

Fig. 9 shows the performance for increasing number of frequencies, which rapidly converges to a fixed mAP. This is the mAP of the exact evaluation of (7). We set $N = 3$ as a compromise between dimensionality expansion and performance. Therefore the modulation multiplies the input dimensionality by 7.

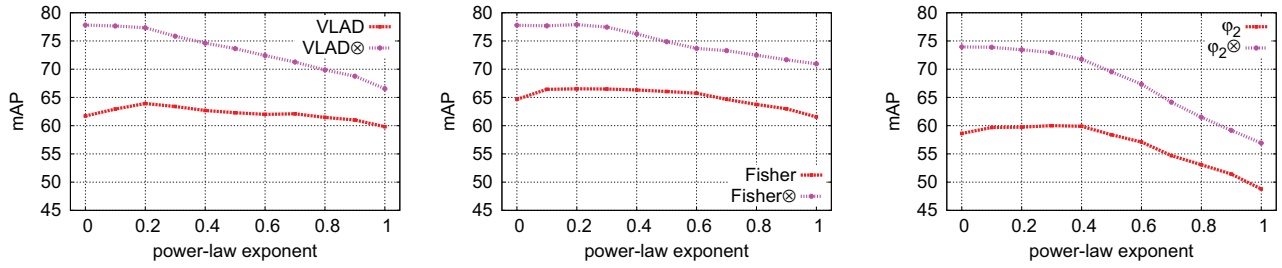


Fig. 7. Impact of powerlaw for modulated and non-modulated image representations. Results on Holidays dataset with a codebook of 32 visual words for VLAD and Fisher vectors. We follow the modified power-law normalization for the modulated vectors.

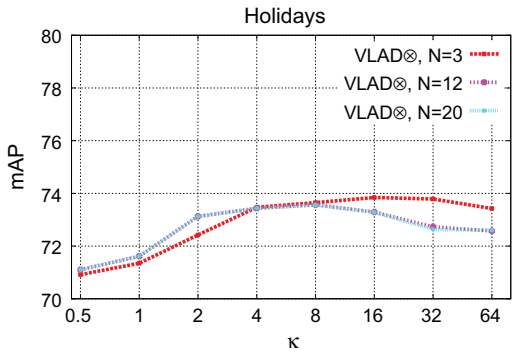


Fig. 8. Performance on Holidays dataset of modulated VLAD for different values of κ and for different approximations. Results shown with RN. A codebook of 32 visual words is used.

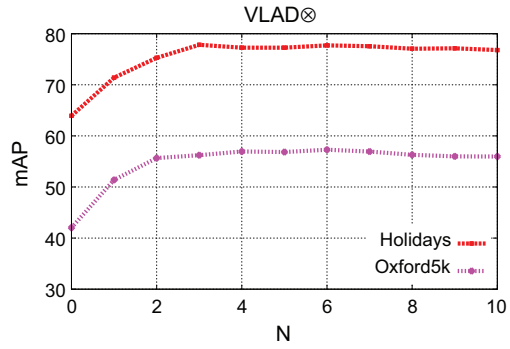


Fig. 9. Performance comparison of modulated VLAD for increasing number of components of the angle feature map. Zero corresponds to original VLAD (not modulated). A codebook of 32 visual words is used.

6.4. Benefit of angular modulation

Table 1 shows the benefit of modulation when applied to the monomial embeddings φ_1, φ_2 and φ_3. The results are on par with the recent coding techniques like VLAD or Fisher improved with modulation. We consider the obtained performance as one of our main achievements, because the representation is codebook-free and requires no learning. In addition, we further show the benefit of combining monomial embeddings with RN. This significantly boosts performance with the same vector dimensionality and negligible computational overhead.

To demonstrate the benefit of the proposed method, we compare VLAD, Fisher and monomial embeddings to their modulated counterparts. Fig. 10 shows that modulation significantly improves the performance for the same codebook size. Given that the modulated vector is 7 times larger (with $N = 3$), the comparison focuses on the performance obtained with the same dimensionality. Even in this case, modulated VLAD⊗ and Fisher⊗ offer a significant improvement. We can conclude that it is better to increase the dimensionality by modulation than using a larger codebook.

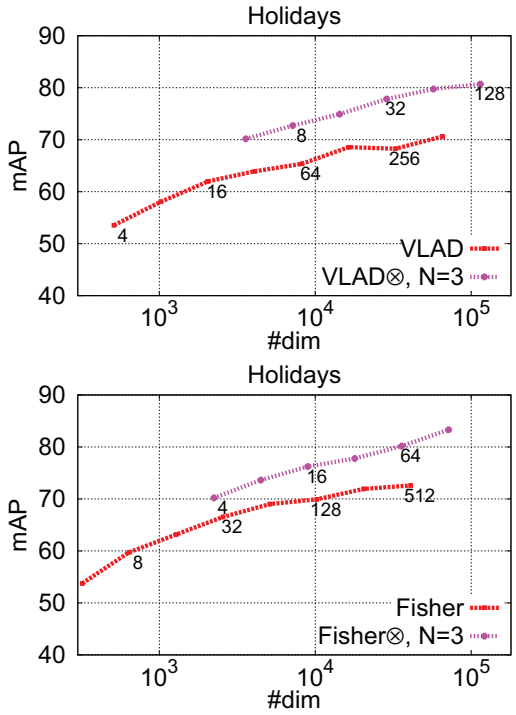


Fig. 10. Impact of modulation on VLAD and Fisher: Performance *versus* dimensionality of the final vector for VLAD (top) and Fisher (bottom) compared to their modulated counterparts. Codebook size is shown with text labels. Results for Holidays dataset.

6.5. Comparison to other methods

We compare our approach, in particular, to CVLAD, as this work also intends to integrate the dominant orientation into a vector representation. We consistently apply 8 query rotations for both CVLAD and our method on Holidays dataset. Fig. 11 shows the respective performance measured for different codebooks. The proposed methods appear to consistently outperform CVLAD, both for the same codebook and for the same dimensionality. Noticeably, the modulated embedded monomial $\varphi_2\otimes$ is on par with or better than CVLAD.

We also compare to other prior works and present results in Table 2 for Holidays, Oxford5k and Oxford105k. We outperform by a large margin the state of the art with full vector representations. Further, our approach is arguably compatible with these concurrent approaches, which may bring further improvement. Note that RN also boosts performance for VLAD and Fisher. In particular with a codebook of size 32, they achieve 50.4 and 49.8 respectively on Oxford5k. Our scores on Holidays with Fisher⊗ and RN are also competitive to those reported by state-of-the-art methods based on large codebooks [12]. To our knowledge, this is the first time that a vector representation compatible with inner product attains such image search performance.

On Oxford5k and Oxford105k we do not evaluate multiple query rotations for our method. A simple way to enforce up-right objects for our baseline methods is to use up-right features. Performance

Table 1
Impact of modulation on monomial embeddings of order 1, 2 and 3. The performance is reported for Holidays dataset. RN = Rotation and Normalization.

Method	φ_1	$\varphi_1\otimes$			φ_2		$\varphi_2\otimes$				φ_3	$\varphi_3\otimes$
RN						×			×	×		
N	–	1	3	6	–	–	1	3	1	3	–	1
#dim	80	240	560	1,040	3,240	3,240	9,720	22,680	9,720	22,680	88,560	265,680
mAP	35.4	47.2	58.5	**62.5**	59.7	74.3	68.7	73.9	76.3	**79.7**	60.0	70.8

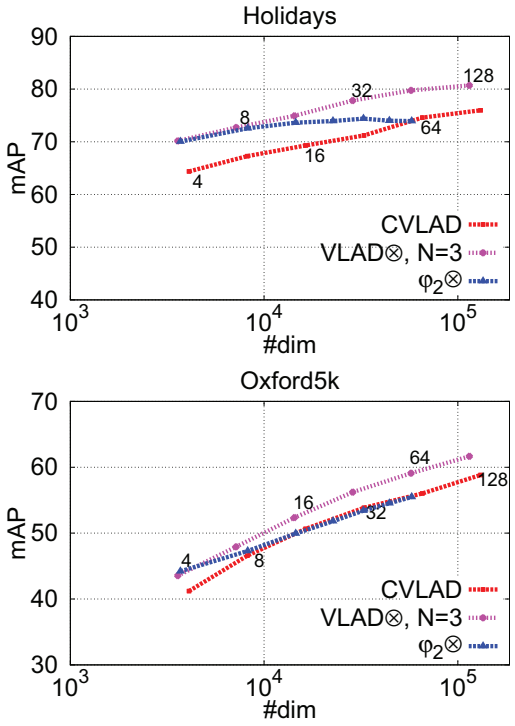


Fig. 11. Comparison to CVLAD. We measure performance on Holidays and Oxford5k for CVLAD and our proposed methods for increasing codebook size. The codebook cardinality is shown with text labels for CVLAD and modulated VLAD, while for φ_2 it is the dimensionality of the input vectors after PCA reduction that we vary.

Table 2
Performance comparison with state of the art approaches. Results with the use of full vector representation. #C: size of codebook. #dim: Number of components of each vector. Modulation is performed with $N = 3$ for all cases, except for φ_3, where $N = 1$. We do not use any re-ranking or spatial verification in any experiment. Results followed by a citation are the ones reported in the original publication.

Method	#C	#dim	RN	Holidays	Oxf5k	Oxf105k
VLAD [15]	64	4,096		55.6	37.8	–
Fisher [15]	64	4,096		59.5	41.8	–
VLAD [15]	256	16,384		58.7	–	–
Fisher [15]	256	16,384		62.5	–	–
Arandjelovic [28]	256	32,536		65.3	**55.8**	–
Delhumeau [48]	64	8,192		65.8	51.7	45.6
Zhao [22]	32	32,768		**68.8**	42.7	–
VLAD⊗	32	28,672		77.8	56.2	51.4
VLAD⊗	32	28,672	×	80.8	62.1	**53.8**
Fisher⊗	32	17,920		77.7	52.3	47.3
Fisher⊗	32	17,920	×	81.3	61.3	52.6
Fisher⊗	64	35,840	×	**83.6**	**65.1**	–
φ_2⊗	n/a	22,680		73.9	51.8	45.7
φ_2⊗	n/a	22,680	×	79.7	60.9	51.8
φ_3⊗	n/a	265,680		70.8	55.0	–

on Oxford5k achieved by VLAD with codebook of size 256 and with up-right features of the same detector is 53.4, while the corresponding score with rotation invariant features is 52.4. Even though switching off rotation when all objects are aligned seems to slightly increase performance, our method appears to be more effective (VLAD⊗ achieves 56.2 with a codebook of size 32). Moreover, note that up-right features are not applicable when object rotation exists, while our rotation covariant match kernel is.

Table 3 reports the performance after dimensionality reduction to 128 or 1024 components. The same set of local features and codebooks are used for all methods. We observe a consistent improvement over the original encoding.

Table 3
Oxford105k: performance comparison (mAP) after dimensionality reduction with PCA into 128 and 1024 components. The results with the full vector representation are with RN. Observe the consistent gain (in parentheses) brought by our approach for a fixed output dimensionality of 1024 or 128 components.

Method	#dim	full dim	dim → 1024	dim → 128
VLAD	4096	40.3	37.3	26.0
VLAD⊗	28,672	53.8	**40.7** (+3.4)	**28.7** (+2.7)
Fisher	2560	39.6	34.6	24.8
Fisher⊗	17,920	52.6	40.3 (+5.7)	27.6 (+2.8)
φ_2	3240	37.2	32.4	21.3
φ_2⊗	22,680	51.8	40.0 (+7.6)	26.8 (+5.5)

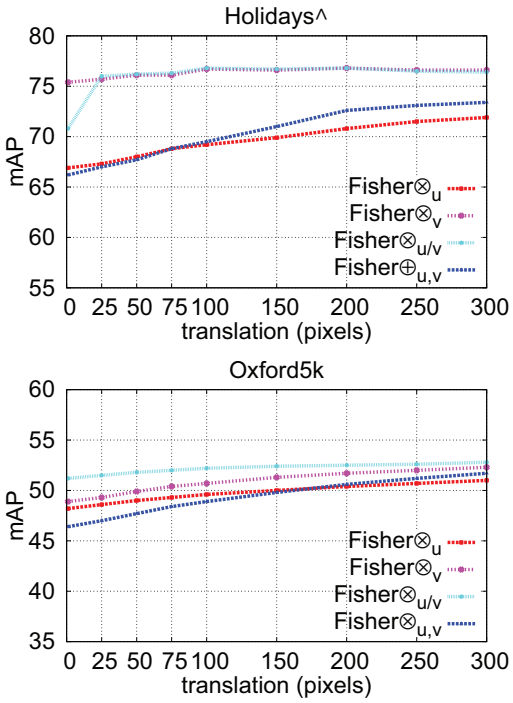


Fig. 12. Impact of multiple translations to the performance of our translation covariant match kernel. Results reported on upright Holidays^ and Oxford5k. A codebook of 32 (8) visual words is used for single (double) modulation. Results are reported using the feature detector with the gravity vector assumption Perdoch et al. [43].

6.6. Position encoding

The selectivity parameter κ is set equal to 8, similarly to the case of dominant orientation. We evaluate the influence of multiple translations applied to the query image in order to identify the best alignment. In Fig. 12 we present the impact of the translation on the performance. Performance improves as we evaluate multiple translations and saturates around 250 pixels. In the rest of our experiments we set the maximum translation value to 250 pixels.

Interestingly, on Holidays^ the vertical coordinate is significantly more informative than the horizontal one. This might be due to the nature of this dataset; images depicting the same landmark have been shot by the same person. Therefore, the objects/scenes have been photographed roughly at the same height, but with horizontal slides in a panoramic manner. On Oxford5k this is not the case.

We evaluate the position encoding for single and double modulation for comparable dimensionality (approximately 30K). The performance increases by taking into account more translations for both methods. Single modulation performs better than the double counterpart for the given dimensionality. The increase of dimensionality for the latter is high, but it uses a much smaller codebook.

Table 4
Comparison of the translation covariant match kernel and baseline VLAD and Fisher vectors. Results are reported using the feature detector with the gravity vector assumption by Perdoch et al. [43]. Note that these results are not directly comparable to the ones of Table 2 due to different features and different versions of Holidays dataset (rotated *versus* original).

Method	N	#C	#dim	Holidays^			Oxf5k		
VLAD	n/a	128	16,384	67.5			53.3		
VLAD	n/a	256	32,768	70.1			56.7		
Fisher	n/a	128	10,240	73.4			54.7		
Fisher	n/a	256	20,480	73.7			**57.6**		
$\chi =$				u	v	u/v	u	v	u/v
VLAD$\otimes_\chi$	2	32	20,480	68.8	72.5	72.4	49.3	50.9	50.8
VLAD$\otimes_\chi$	3	32	28,672	69.8	74.3	74.4	50.4	52.5	53.2
Fisher$\otimes_\chi$	2	32	12,800	71.6	75.8	75.7	49.8	51.0	51.5
Fisher$\otimes_\chi$	3	32	17,920	71.5	**76.6**	76.5	50.9	52.1	53.3
$\chi =$				u, v					
VLAD$\otimes_\chi$	2	16	51,200	66.4			47.2		
VLAD$\otimes_\chi$	3	8	50,176	68.5			46.9		
Fisher$\otimes_\chi$	2	16	32,000	75.7			52.1		
Fisher$\otimes_\chi$	3	8	31,360	73.1			51.2		

In Table 4 we show results for the baseline VLAD, and Fisher vector, and their modulated counterparts with respect to position. By considering 2 instead of 3 frequencies the dimensionality of the modulated descriptor is reduced by 30% with just a small loss (1–2%). We note that, once more, comparing to the baseline the advantage relies on the fact that the visual codebook is smaller and the assignment to that is faster to compute. Recall that now scale invariance is lost, leading to lower performance on Oxford5k. In the case of scale changes the dominant orientation is more distinctive and reliable. Overall, the orientation information brings higher improvement compared to the spatial one.

In order to provide a direct comparison between the two proposed methods we evaluate the translation covariant match kernel on Oxford5k and with the same features as those of the experiments reported in Table 2. VLAD$\otimes_u$ and VLAD$\otimes_v$ achieve 47.1 and 50.1 respectively by computing similarity for 25 translations on each direction with a codebook of size 32. Concerning the position modulated Fisher vectors the corresponding scores are 43.0 and 45.7. It appears that encoding rotation is more effective for this use-case. However, the translation covariant match kernel opens other possible directions, such as application on image classification, as a continuous alternative to spatial pyramid match.

6.7. Timings

The image representation created by modulating the monomial embedding φ_2 using $N = 3$ takes on average 68 ms for a typical image with 3000 SIFT descriptors. The resulting aggregated vector representation has 22,680 components. The average query time with such a representation on Holidays is 5.8 ms, assuming no query rotation. Employing the trigonometric polynomial of scores results in 5.9 ms (6.1 ms) for 8 (64) possible fixed rotations. The corresponding timings for Oxford105k and vectors reduced to 128 dimensions are 55 ms (no rotations) and 134 ms (8 fixed rotations). In the case of reduced vectors, the query rotations are computed with the naive way. Note, these timings are better than those achieved by a Bag-of-Words representation with a large vocabulary, for which the quantization typically takes about 1 s with an approximate nearest neighbor search algorithm like FLANN [55]. Our timings are measured with a single threaded implementation on an Intel Xeon E5-4640@2.40 GHz.

7. Conclusion

Our modulation strategy integrates geometric information directly in the coding stage. Dominant orientation of local features or their position is jointly encoded with the local descriptor, in a continuous manner. Our method is inspired by and builds upon recent works on explicit feature maps and kernel descriptors. Thanks to a generic formulation provided by match kernels, it is compatible with coding strategies such as Fisher vector or VLAD.

Invariance is offered by estimating maximum similarity for multiple image rotations or translations. The nature of our representation enables this very efficiently with a simple trigonometric polynomial. Our context (datasets) simply demands just a few sampling points on the rotation or translation domain. However, note that our methodology provides high efficiency even for denser search.

Our experiments demonstrate a consistent gain compared to the original coding in all cases. Angular modulation appears to be more promising than that of position for the task that we examine. Interestingly, our method is also very effective with a simple monomial kernel, offering competitive performance for image search with a coding stage not requiring any quantization.

Whatever the coding stage that we use with our approach, the resulting representation is compared with inner product, which suggests that it is compliant with linear classifiers such as those considered in image classification.

Acknowledgments

This work was supported by ERC grant VIAMASS No. 336054 and ANR project Fire-ID.

References

[1] D. Lowe, Distinctive image features from scale-invariant keypoints, Int. J. Comput. Vis. 60 (2) (2004) 91–110.
[2] H. Bay, A. Ess, T. Tuytelaars, L.V. Gool, SURF: speeded up robust features, Comput. Vis. Image Underst. 110 (3) (2008) 346–359.
[3] T. Ojala, M. Pietikainen, T. Maenpaa, Multiresolution gray-scale and rotation invariant texture classification with local binary patterns, Trans. Pattern Anal. Mach. Intell. 24 (7) (2002) 971–987.
[4] J. Sivic, A. Zisserman, Video Google: A text retrieval approach to object matching in videos, in: Proceedings of International Conference on Computer Vision, ICCV, 2003.
[5] J. Philbin, O. Chum, M. Isard, J. Sivic, A. Zisserman, Object retrieval with large vocabularies and fast spatial matching, in: Proceedings of Conference on Computer Vision and Pattern Recognition, CVPR, 2007.
[6] G. Tolias, Y. Avrithis, Speeded-up, relaxed spatial matching, in: Proceedings of International Conference on Computer Vision, ICCV, 2011.
[7] O. Chum, J. Philbin, J. Sivic, M. Isard, A. Zisserman, Total recall: Automatic query expansion with a generative feature model for object retrieval, in: Proceedings of International Conference on Computer Vision, ICCV, 2007.
[8] G. Tolias, H. Jégou, Visual query expansion with or without geometry: refining local descriptors by feature aggregation, Pattern Recognition (2014).

[9] H. Jégou, M. Douze, C. Schmid, Improving bag-of-features for large scale image search, Int. J. Comput. Vis. 87 (3) (2010) 316–336.
[10] A. Mikulík, M. Perdoch, O. Chum, J. Matas, Learning a fine vocabulary, in: Proceedings of European Conference on Computer Vision, ECCV, 2010.
[11] H. Jégou, M. Douze, C. Schmid, On the burstiness of visual elements, in: Proceedings of Conference on Computer Vision and Pattern Recognition, CVPR, 2009.
[12] G. Tolias, Y. Avrithis, H. Jégou, To aggregate or not to aggregate: Selective match kernels for image search, in: Proceedings of International Conference on Computer Vision, ICCV, 2013.
[13] J. Wang, J. Yang, F.L.K. Yu, T. Huang, Y. Gong, Locality-constrained linear coding for image classification, in: Proceedings of Conference on Computer Vision and Pattern Recognition, CVPR, 2010.
[14] F. Perronnin, C.R. Dance, Fisher kernels on visual vocabularies for image categorization, in: Proceedings of Conference on Computer Vision and Pattern Recognition, CVPR, 2007.
[15] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, C. Schmid, Aggregating local descriptors into compact codes, in: Trans. Pattern Anal. Mach. Intell., 2012.
[16] F. Perronnin, Y. Liu, J. Sánchez, H. Poirier, Large-scale image retrieval with compressed Fisher vectors, in: Proceedings of Conference on Computer Vision and Pattern Recognition, CVPR, 2010.
[17] M. Charikar, Similarity estimation techniques from rounding algorithms, in: Proceedings of Annual ACM Symposium on Theory of Computing, STOC, 2002.
[18] H. Jégou, M. Douze, C. Schmid, Product quantization for nearest neighbor search, Trans. Pattern Anal. Mach. Intell. 33 (1) (2011) 117–128.
[19] S. Lazebnik, C. Schmid, J. Ponce, Beyond bags of features: spatial pyramid matching for recognizing natural scene categories, in: Proceedings of Conference on Computer Vision and Pattern Recognition, CVPR, 2006.
[20] M. Douze, H. Jégou, H. Singh, L. Amsaleg, C. Schmid, Evaluation of GIST descriptors for web-scale image search, in: Proceedings of Conference on Computer Vision and Pattern Recognition, CVPR, 2009.
[21] P. Koniusz, F. Yan, K. Mikolajczyk, Comparison of mid-level feature coding approaches and pooling strategies in visual concept detection, Computer Vision and Image Understanding 17 (5) (2013) 479–492.
[22] W. Zhao, H. Jégou, G. Gravier, Oriented pooling for dense and non-dense rotation-invariant features, in: Proceedings of the British Machine Vision Conference, BMVC, 2013.
[23] L. Bo, C. Sminchisescu, Efficient match kernel between sets of features for visual recognition, in: Proceedings of Neural Information Processing Systems, NIPS, 2009.
[24] G. Tolias, T. Furon, H. Jégou, Orientation covariant aggregation of local descriptors with embeddings, in: Proceedings of European Conference on Computer Vision, ECCV, 2014, pp. 382–397.
[25] H. Jégou, M. Douze, C. Schmid, Hamming embedding and weak geometric consistency for large scale image search, in: Proceedings of European Conference on Computer Vision, ECCV, 2008.
[26] L. Bo, X. Ren, D. Fox, Kernel descriptors for visual recognition, in: Proceedings of Neural Information Processing Systems, NIPS, 2010.
[27] A. Vedaldi, A. Zisserman, Efficient additive kernels via explicit feature maps, Trans. Pattern Anal. Mach. Intell. 34 (3) (2012) 480–492.
[28] R. Arandjelovic, A. Zisserman, All about VLAD, in: Proceedings of Conference on Computer Vision and Pattern Recognition, CVPR, 2013.
[29] J. Krapac, J. Verbeek, F. Jurie, Modeling spatial layout with fisher vectors for image categorization, in: Proceedings of International Conference on Computer Vision, ICCV, IEEE, Barcelona, Spain, 2011, pp. 1487–1494.
[30] P. Koniusz, K. Mikolajczyk, Spatial coordinate coding to reduce histogram representations, dominant angle and colour pyramid match, in: Proceedings of International Conference on Image Processing, ICIP, 2011, pp. 661–664.
[31] J. Sánchez, F. Perronnin, T. de Campos, Modeling the spatial layout of images beyond spatial pyramids, Pattern Recognit. Lett. 33 (16) (2012) 2216–2223.
[32] P.-H. Gosselin, N. Murray, H. Jégou, F. Perronnin, Revisiting the Fisher vector for fine-grained classification, Pattern Recognit. Lett. 49 (2014) 92–98.
[33] L. Bo, K. Lai, X. Ren, D. Fox, Object recognition with hierarchical kernel descriptors, in: Proceedings of Conference on Computer Vision and Pattern Recognition, CVPR, 2011.
[34] S. Lyu, Mercer kernels for object recognition with local features, in: Proceedings of Conference on Computer Vision and Pattern Recognition, CVPR, 2005.
[35] G. Csurka, C.R. Dance, L. Fan, J. Willamowski, C. Bray, Visual categorization with bags of keypoints, in: Proceedings of the ECCV Workshop Statistical Learning in Computer Vision, 2004.
[36] D. Picard, P.-H. Gosselin, Efficient image signatures and similarities using tensor products of local descriptors, Comput. Vis. Image Underst. 117 (2013).
[37] J.C.R. Caseiro, J. Batista, C. Sminchisescu, Semantic segmentation with second-order pooling, in: Proceedings of European Conference on Computer Vision, ECCV, 2012.
[38] M. Abramowitz, I.A. Stegun, Handbook of Mathematical Functions With Formulas, Graphs, and Mathematical Tables, Volume 55 of National Bureau of Standards Applied Mathematics Series, U.S. Government Printing Office, 1964.
[39] E.G. Bazavan, F. Li, C. Sminchisescu, Fourier kernel learning, in: Proceedings of European Conference on Computer Vision, ECCV, 2012.
[40] F. Perronnin, J. Sánchez, T. Mensink, Improving the Fisher kernel for large-scale image classification, in: Proceedings of European Conference on Computer Vision, ECCV, 2010.
[41] Y. Zhang, Z. Jia, T. Chen, Image retrieval with geometry-preserving visual phrases, in: Proceedings of Conference on Computer Vision and Pattern Recognition, CVPR, 2011, pp. 809–816.
[42] J.F. Henriques, R. Caseiro, P. Martins, J. Batista, High-speed tracking with kernelized correlation filters, Trans. Pattern Anal. Mach. Intell. (2015).
[43] M. Perdoch, O. Chum, J. Matas, Efficient representation of local geometry for large scale object retrieval, in: Proceedings of Conference on Computer Vision and Pattern Recognition, CVPR, 2009.
[44] T. Jaakkola, D. Haussler, Exploiting generative models in discriminative classifiers, in: Proceedings of Neural Information Processing Systems, NIPS, 1998.
[45] K. Chatfield, V. Lempitsky, A. Vedaldi, A. Zisserman, The devil is in the details: an evaluation of recent feature encoding methods, in: Proceedings of the British Machine Vision Conference, BMVC, 2011.
[46] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, L.V. Gool, A comparison of affine region detectors, IJCV 65 (1/2) (2005) 43–72.
[47] R. Arandjelovic, A. Zisserman, Three things everyone should know to improve object retrieval, in: Proceedings of Conference on Computer Vision and Pattern Recognition, CVPR, 2012.
[48] J. Delhumeau, P.-H. Gosselin, H. Jégou, P. Pérez, Revisiting the VLAD image representation, in: ACM Multimed., 2013.
[49] J. Philbin, O. Chum, M. Isard, J. Sivic, A. Zisserman, Lost in quantization: improving particular object retrieval in large scale image databases, in: Proceedings of Conference on Computer Vision and Pattern Recognition, CVPR, 2008.
[50] M. Douze, H. Jégou, The Yael library, in: ACM Multimed., 2014.
[51] A. Torii, J. Sivic, T. Pajdla, M. Okutomi, Visual place recognition with repetitive structures, in: Proceedings of Conference on Computer Vision and Pattern Recognition, CVPR, 2013.
[52] H. Jégou, O. Chum, Negative evidences and co-occurences in image retrieval: The benefit of PCA and whitening, in: Proceedings of European Conference on Computer Vision, ECCV, 2012.
[53] B. Safadi, G. Quenot, Descriptor optimization for multimedia indexing and retrieval, in: Proceedings of Content-Based Multimedia Indexing, CBMI, 2013.
[54] H. Jégou, A. Zisserman, et al., Triangulation embedding and democratic aggregation for image search, in: Proceedings of Conference on Computer Vision and Pattern Recognition, CVPR, 2014.
[55] M. Muja, D.G. Lowe, Fast approximate nearest neighbors with automatic algorithm configuration, in: Proceedings of International Conference on Computer Vision Theory and Applications, VISAPP, 2009.
[56] P. Koniusz, F. Yan, P.-H. Gosselin, K. Mikolajczyk, Higher-order Occurrence Pooling on Mid-and Low-level Features: Visual Concept Detection, INRIA, 2013.

V Image search with selective match kernels - aggregation across single and multiple images

Title: Image search with selective match kernels - aggregation across single and multiple images
Authors: G. Tolias, Y. Avrithis, H. Jégou
Published at: IJCV 2015

Noname manuscript No.
(will be inserted by the editor)

Image search with selective match kernels: aggregation across single and multiple images

Giorgos Tolias · Yannis Avrithis · Hervé Jégou
INRIA NTUA INRIA

the date of receipt and acceptance should be inserted later

Abstract This paper considers a family of metrics to compare images based on their local descriptors. It encompasses the VLAD descriptor and matching techniques such as Hamming Embedding. Making the bridge between these approaches leads us to propose a match kernel that takes the best of existing techniques by combining an aggregation procedure with a selective match kernel. The representation underpinning this kernel is approximated, providing a large scale image search both precise and scalable, as shown by our experiments on several benchmarks.

We show that the same aggregation procedure, originally applied per image, can effectively operate on groups of similar features found across multiple images. This method implicitly performs feature set augmentation, while enjoying savings in memory requirements at the same time. Finally, the proposed method is shown effective for place recognition, outperforming state of the art methods on a large scale landmark recognition benchmark.

1 Introduction

This paper is interested in improving visual recognition of objects, locations and scenes. The best existing approaches rely on local descriptors [28,29]. Most of them inherit from the seminal Bag-of-Words (BOW) representation [42,11]. It employs a visual vocabulary to quantize a set of local descriptors and to produce a single vector that represents the image. This offers several desirable properties. For image classification [11], it is compatible with powerful machine learning techniques such as support vectors machines. In this case, it is usually employed with relatively small visual vocabularies. In a query by content scenario [42], which is the focus of our paper, large vocabularies make the search efficient [31,36,30], thanks to inverted file structures [39] that exploit the sparsity of the representation. The methods relying on these ingredients are typically able to search in millions of images in a few seconds or less.

Several researchers have built upon this approach to design better retrieval systems. In particular, the search is advantageously refined by re-ranking approaches, which operate on an initial short-list of images. This is done by exploiting additional geometrical information [36,32,44] or applying query expansion techniques [10,46].

Another important improvement is obtained by reducing the quantization noise. This is done by multiple assignment [37,20], or by exploiting a more precise representation of the individual local descriptors, such as binary codes in the so-called Hamming Embedding (HE) method [20], or by integrating some information about the neighborhood of the descriptor [51]. All these approaches implicitly rely on approximate pair-wise matching of the query descriptors with those of the database images.

In a concurrent effort to scale to even larger databases, recent encoding techniques such as Fisher kernels [33,35], local linear coding [50] or the "vector of locally aggregated descriptors" (VLAD) [22], depart from the BOW framework by introducing alternative encoding schemes. By compressing the resulting vector representation [22,34], the local descriptors are not considered individually. Images can be represented by a small number of bytes, similar to coded global descriptors [48], but with the advantage of preserving some key properties inherited from local descriptors, such as rotation and scale invariance.

Our paper introduces a framework to bridge the gap between the "matching-based" approaches, such as HE, and the recent aggregated representations, in particular VLAD. For this purpose, we introduce in Section 3 a class of match

This work was supported by ERC grant VIAMASS no. 336054 and ANR project Fire-ID.

Address(es) of author(s) should be given

kernels that includes both matching-based and aggregated methods for unsupervised image search.

We then discuss and analyze in Section 4 two key differences between matching-based and aggregated approaches. First, we consider the *selectivity* of the matching function, *i.e.* , the property that a correspondence established between two patches contributes to the image-level similarity only if the confidence is high enough. It is explicitly exploited in matching-based approaches only.

Second, the aggregation (or pooling) operator used in BoW, VLAD or in the Fisher vector, is not considered in pure matching approaches such as HE. We show that it is worth doing it even in matching-based approaches, and discuss its relationship with other methods (*e.g.* , [19,35]) introduced to handle the non-*iid* statistical behavior of local descriptors, also called the burstiness phenomenon [19].

This leads us to the conclusion that none of the existing schemes combines the best ingredients required to achieve the best possible image retrieval quality. As a result, we introduce a new method that exploits the best of both worlds to produce a strong image representation and its corresponding kernel between images. It combines an aggregation scheme with a selective kernel. This vector representation is advantageously compressed by vector binarization to drastically reduce the memory requirements, while also improving the search efficiency.

The compressed vectors offer an efficient way to detect groups of similar features in large image collections. We thus show that descriptor aggregation can further be applied off-line across multiple images. In contrast to previous approaches of feature augmentation [1] or cross matching images for query expansion [12,41,14], this method does not increase either storage requirements or query time.

Generic image search methods have been shown quite effective for the task of image geo-localization [47]. Location recognition is typically achieved by matching rigid objects, especially buildings. Our approach, by enforcing selective feature matching and by handling burstiness, is successfully utilized for this task. It is true that bursty matches dominate in urban sceneries and building photos.

Section 6 shows that our method significantly outperforms the state of the art in a comparable setup, *i.e.* when comparing the quality of the initial result set produced while searching a large collection. We further evaluate our method on the San Francisco landmark recognition dataset [8], again outperforming the state of the art.

This paper is the continuation of our previous work [45]. In the original work we presented the common framework for matches kernels and the selective match kernels for single images, now described in Sections 3 and 4, respectively. The new contributions are the extension to aggregation of similar features derived from multiple images, described in Section 5, and the application on place recognition.

2 Related work

Improved representations. A few works try to improve the matching performed by vectorized representations such as VLAD and Fisher vectors or by voting approaches such as Hamming Embedding. Arandjelovic and Zisserman [2] propose an improved vector normalization for VLAD and a way to adopt the codebook on the indexed dataset. At the same time, we indepently propose to apply the same normalization [45], but in the context of large vocabularies. Delhumeau *et al.* [13] revisit the VLAD representation by simple modifications, which are rather effective. Despite our common framework encompassing the VLAD representation, we depart from those approaches by focusing on larger codebooks and a sparse representation.

Attempting to improve the similarity estimation in HE, Jain *et al.* perform asymmetric distance computation [17]. Concurrently with our work [45], Tao *et al.* [43] also propose to use a selective matching function and to extend aggregated representations such as VLAD and Fisher vectors to large codebooks. The two different functions, our polynomial versus their exponential, are evaluated and shown to perform more or less the same. The main difference is that we apply the selectivity function after aggregation, a choice that results in the aggregated match kernel.

In both works [45,43] the match kernel is seen as a voting approach due to the non-linearity of the weighting function. In contrast, with the use of random Fourier features [5], a selective function is well approximated and the match kernel is estimated by the inner product of aggregated and vectorized representation.

Qin *et al.* [38] propose a probabilistic framework for feature matching and adapt the similarity measure to the query features. This concurrent work is close to ours, in the sense that they also study the increase in performance by using more space to index each local descriptor.

Query expansion. There are query expansion approaches [10, 9,46], which act as automatic relevance feedback. Relevant images are automatically identified and used to form an expanded query representation. In a different direction, there are methods which cross-match indexed images [12,41,52, 14] and employ this information at query time. Typically, index memory requirements are increased and so is query time. We overcome both those limitations. Similarly, there are approaches that combine information from several views of the same object or scene in a single compressed representation [24,4].

On a local feature level, the feature set of an indexed image is augmented by its spatially verified database counterparts [49,1]. The use of spatial matching makes the off-line cost of such approaches rather high. In our case, we implicitly offer a similar augmentation effect, but in a very efficient way, without using any spatial information.

Place recognition has lately received a lot of attention. It can be handled as a classification task, where groups of landmark photos are automatically identified and classifiers per landmark are trained [27]. More interestingly, generic image search approaches have been proven quite effective. Images are individually indexed and recognition is performed based on top ranked landmarks or locations. One of the first examples is the work of Hays and Efros [15].

Along the same lines, some methods employ supervision and use geo-tagged datasets to build improved codebooks [40,23], or detect common distracting patterns [26]. Our approach for place recognition follows the same principle as the work of Torii *et al.* [47]. It composes a generic retrieval method that is appropriate for matching urban scenery, by effectively handling bursty matches and by improving the matching accuracy of local descriptors.

3 A framework for match kernels

This section first describes the class of match kernels that we will analyze in this paper. This framework encompasses several popular techniques published in the literature. In the following, we denote the cardinality of a set $\mathcal{A}$ by $\#\mathcal{A}$.

Let us assume that an image is described by a set $\mathcal{X} = \{x_1, \ldots, x_n\}$ of n d-dimensional local descriptors. The descriptors are quantized by a k-means quantizer

$$\begin{aligned} q : \mathbb{R}^d &\to \mathcal{C} \subset \mathbb{R}^d \\ x &\mapsto q(x) \end{aligned} \tag{1}$$

where $\mathcal{C} = \{c_1, \ldots, c_k\}$ is a codebook comprising $k = \#\mathcal{C}$ vectors, which are referred to as visual words. We denote by $\mathcal{X}_c = \{x \in \mathcal{X} : q(x) = c\}$ the subset of descriptors in $\mathcal{X}$ that are assigned to a particular visual word c. In order to compare two image representations $\mathcal{X}$ and $\mathcal{Y}$, we consider a family of set similarity functions K of the general form

$$\mathrm{K}(\mathcal{X}, \mathcal{Y}) = \gamma(\mathcal{X})\,\gamma(\mathcal{Y}) \sum_{c \in \mathcal{C}} w_c\, \mathrm{M}\left(\mathcal{X}_c, \mathcal{Y}_c\right), \tag{2}$$

where function M is defined between two sets of descriptors $\mathcal{X}_c, \mathcal{Y}_c$ assigned to the same visual word. Depending on the definition of M, the set similarity function K is or is not a positive-definite kernel.

The scalar w_c is a constant that depends on visual word c, for instance it integrates the inverse document frequency (IDF) weighting term. The normalization factor $\gamma(.)$ is typically computed as

$$\gamma(\mathcal{X}) = \left(\sum_{c \in \mathcal{C}} w_c\, \mathrm{M}\left(\mathcal{X}_c, \mathcal{X}_c\right) \right)^{-1/2}, \tag{3}$$

such that the self-similarity of an image is $\mathrm{K}(\mathcal{X}, \mathcal{X}) = 1$. Several popular methods of the literature can be described by the framework of Equation (2).

Bag-of-words. The BOW representation [42,11] represents each local descriptor x solely by its visual word. As noticed in [5,20], bag-of-words with cosine similarity can be expressed in terms of Equation (2), by defining

$$\mathrm{M}(\mathcal{X}_c, \mathcal{Y}_c) = \#\mathcal{X}_c \times \#\mathcal{Y}_c = \sum_{x \in \mathcal{X}_c} \sum_{y \in \mathcal{Y}_c} 1, \tag{4}$$

Other comparison metrics are also possible. For instance, the histogram intersection would use $\min(\#X_c, \#Y_c)$ instead. In the case of max-pooling [6], $\mathrm{M}(\mathcal{X}_c, \mathcal{Y}_c)$ would be equal to 1 if both $\mathcal{X}_c, \mathcal{Y}_c$ are non-empty, and zero otherwise.

Hamming Embedding (HE) [18,20] is a matching model that extends BOW by representing each local descriptor x with both its quantized value $q(x)$ and a binary code b_x of B bits. It computes the scores between all pairs of descriptors assigned to the same visual word, as

$$\mathrm{M}\left(\mathcal{X}_c, \mathcal{Y}_c\right) = \sum_{x \in \mathcal{X}_c} \sum_{y \in \mathcal{Y}_c} w\left(h\left(b_x, b_y\right)\right), \tag{5}$$

where h is the Hamming distance and w is a weighting function that associates a weight to each of the $B + 1$ possible distance values. This function was first defined as binary [18], such that $w(h) = 1$ if $h \leq \tau$, and 0 otherwise. A smoother weighting scheme is a better choice [19,20], such as the (thresholded) Gaussian function [19]

$$w(h) = \begin{cases} e^{-h^2/\sigma^2}, & h \leq \tau \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

We assume that binary codes lie in the B-dimensional Hamming space $\{-1, +1\}^B$ and use the Hamming inner product

$$\langle a, b \rangle_h = \frac{a^\top b}{B} = \hat{a}^\top \hat{b} \in [-1, 1] \tag{7}$$

instead of the Hamming distance presented in the original HE paper [18]. Here $\hat{a}$ denotes the ℓ_2-normalized counterpart of vector a. Since $2h(a, b) = B(1 - \langle a, b \rangle_h)$, the two choices are equivalent.

VLAD [22] aggregates the descriptors associated with a given visual word to produce a $d \times k$ vector representation. This vector is constructed as the concatenation of d-dimensional vectors: $\boldsymbol{\nu}(\mathcal{X}) \propto [V(\mathcal{X}_{c_1}), \ldots, V(\mathcal{X}_{c_k})]$, where

$$V(\mathcal{X}_c) = \sum_{x \in \mathcal{X}_c} r(x), \tag{8}$$

and $r(x) = x - q(x)$ is the residual vector of x. Since the similarity of two VLADs is measured by the dot product, it is easy to show that VLAD corresponds to a match kernel of the form proposed in Equation (2):

$$\boldsymbol{\nu}(\mathcal{X})^\top \boldsymbol{\nu}(\mathcal{Y}) = \gamma(\mathcal{X})\,\gamma(\mathcal{Y}) \sum_{c \in \mathcal{C}} V(\mathcal{X}_c)^\top V(\mathcal{Y}_c), \tag{9}$$

where Equation (3) determines the normalization factors. Then it appears that

$$\mathrm{M}\left(\mathcal{X}_c, \mathcal{Y}_c\right) = V(\mathcal{X}_c)^\top V(\mathcal{Y}_c) \tag{10}$$

$$= \sum_{x \in \mathcal{X}_c} \sum_{y \in \mathcal{Y}_c} r(x)^\top r(y). \tag{11}$$

The power-law normalization proposed for Fisher vectors [35] is also integrated in this framework by modifying the definition of V, however it cannot be expanded as Equation (11). Its effect is similar to burstiness handling in [19].

Burstiness [19] refers to the phenomenon whereby a visual word appears more times in an image than what a statistically independent model would predict. It tends to corrupt the visual similarity measure. Once individual contributions are aggregated per cell as in the HE model of Equation (5), one solution is to down-weight highly populated cells.

For instance, one of the most effective burst weighting models of [19] assumes that the outer sum in Equation (5) refers to query descriptors $\mathcal{X}_c$ in the cell and down-weights the inner sum of the descriptors $\mathcal{Y}_c$ of a given database image by $(\#\mathcal{Y}_c(x))^{-1/2}$, where

$$\mathcal{Y}_c(x) = \{y \in \mathcal{Y}_c : w(h(b_x, b_y)) \neq 0\} \tag{12}$$

is the subset of descriptors in $\mathcal{Y}_c$ that match with x. The corresponding match kernel for burstiness normalization is

$$\mathrm{M}\left(\mathcal{X}_c, \mathcal{Y}_c\right) = \sum_{x \in \mathcal{X}_c} (\#\mathcal{Y}_c(x))^{-1/2} \sum_{y \in \mathcal{Y}_c} w(h(b_x, b_y)). \tag{13}$$

A more radical option is down-weighting by $(\#\mathcal{Y}_c(x))^{-1}$, effectively removing multiple matches within cells, similarly to max-pooling [6].

4 Investigating selectivity and aggregation

The three match kernels presented above share some similarities, in particular the fact that the set of descriptors is partitioned into cells and that only vectors lying in the same cell contribute to the overall similarity. VLAD and HE have key characteristics that we discuss in this section. This leads us to explore new possible kernels. We first develop a common model assuming that full descriptors are available in both images, *i.e.* uncompressed vectors, and then consider the case of binarized representations.

4.1 Towards a common model

The non-aggregated kernels individually match all the elements occurring in the same Voronoi cell. They are defined as the set of kernels M of the form

$$\mathrm{M_N}(\mathcal{X}_c, \mathcal{Y}_c) = \sum_{x \in \mathcal{X}_c} \sum_{y \in \mathcal{Y}_c} \sigma\left(\phi(x)^\top \phi(y)\right). \tag{14}$$

Model	$\mathrm{M}(\mathcal{X}_c, \mathcal{Y}_c)$	$\phi(x)$	$\sigma(u)$	$\psi(z)$	$\Phi(\mathcal{X}_c)$
BOW (4)	$\mathrm{M_N}$ or $\mathrm{M_A}$	1	u	z	$\#\mathcal{X}_c$
HE (5)	$\mathrm{M_N}$	$\hat{b}_x$	$w\left(\frac{B}{2}(1-u)\right)$	—	—
VLAD (11)	$\mathrm{M_N}$ or $\mathrm{M_A}$	$r(x)$	u	z	$V(\mathcal{X}_c)$
SMK (20)	$\mathrm{M_N}$	$\hat{r}(x)$	$\sigma_\alpha(u)$	—	—
ASMK (22)	$\mathrm{M_A}$	$r(x)$	$\sigma_\alpha(u)$	$\hat{z}$	$\hat{V}(\mathcal{X}_c)$
SMK* (23)	$\mathrm{M_N}$	$\hat{b}_x$	$\sigma_\alpha(u)$	—	—
ASMK* (24)	$\mathrm{M_A}$	$r(x)$	$\sigma_\alpha(u)$	$\hat{b}(z)$	$\hat{b}(V(\mathcal{X}_c))$

Table 1 Existing and new solutions for the match kernel M. They are classified as non-aggregated $\mathrm{M_N}$ (14) and aggregated kernels $\mathrm{M_A}$ (15), or possibly both. $\phi(x)$: scalar or vector representation of descriptor x. $\sigma(u)$: scalar selectivity of u, where u is assumed normalized in $[-1, 1]$. $\psi(z)$: representation of aggregated descriptor z per cell. $\Phi(\mathcal{X}_c)$ (17): equivalent representation of descriptor set $\mathcal{X}_c$ per cell. Given any vector x, we denote by $\hat{x} = x/\|x\|$ its ℓ_2-normalized counterpart.

This equation encompasses all the variants discussed so far, excluding the burstiness post-processing considered in Equation (12). Here ϕ is an arbitrary vector representation function, possibly non-linear or including normalization, and $\sigma : \mathbb{R} \to \mathbb{R}$ is a scalar selectivity function. Options for these functions are presented in Table 1 and discussed later in this section.

The aggregated kernels, in contrast, are written as

$$\mathrm{M_A}(\mathcal{X}_c, \mathcal{Y}_c) = \sigma\left\{\psi\left(\sum_{x \in \mathcal{X}_c} \phi(x)\right)^\top \psi\left(\sum_{y \in \mathcal{Y}_c} \phi(y)\right)\right\} \tag{15}$$

$$= \sigma\left(\Phi(\mathcal{X}_c)^\top \Phi(\mathcal{Y}_c)\right), \tag{16}$$

where ψ is another vector representation function, again possibly non-linear or including normalization. $\Phi(\mathcal{X}_c)$ is the aggregated vector representation of a set $\mathcal{X}_c$ of descriptors in a cell, such that $\Phi(\emptyset) = \mathbf{0}$ and

$$\Phi(\mathcal{X}_c) = \psi\left(\sum_{x \in \mathcal{X}_c} \phi(x)\right). \tag{17}$$

This formulation suggests other potential strategies. In contrast to Equation (14), there is at most a single match between aggregated representations $\Phi(\mathcal{X}_c)$ and $\Phi(\mathcal{Y}_c)$, and selectivity σ is applied after aggregation.

Of the variants discussed so far, BOW and VLAD both fit into Equation (15), with σ simply being identity. This is not the case for HE matching. Note that the aggregation, *i.e.* , computing $\Phi(\mathcal{X}_c)$, is an off-line operation.

4.2 Non-aggregated matching SMK

We introduce a *selective match kernel* (SMK) in this subsection. It is motivated by the observation that VLAD employs a linear weighting scheme in Equation (11) for the

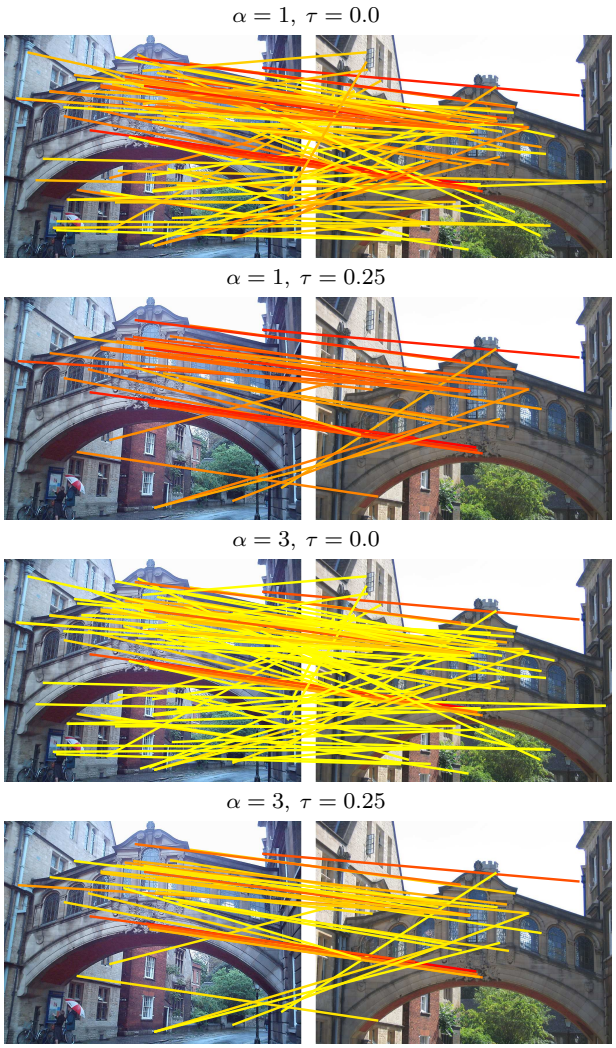


Fig. 1 Matching features with descriptors assigned to the same visual word and similarity above the threshold. Examples for different values of α and τ. Color denotes descriptor similarity defined by $\sigma_\alpha(\hat{r}(x)^\top \hat{r}(y))$, with yellow corresponding to 0 and red to the maximum similarity per image pair.

contribution of individual matching pairs (x, y) to M, while HE applies a non-linear weighting function σ to the similarity $\phi(x)^\top \phi(y)$ between a pair of descriptor x and y.

Choice of selectivity function σ. Without loss of generality, we consider a thresholded polynomial selectivity function $\sigma_\alpha : \mathbb{R} \to \mathbb{R}^+$ of the form

$$\sigma_\alpha(u) = \begin{cases} \text{sign}(u)|u|^\alpha & \text{if } u > \tau \\ 0 & \text{otherwise,} \end{cases} \tag{18}$$

and typically set $\alpha = 3$. In all our experiments we have used $\tau \geq 0$. It plays the same role as the weighting function w in Equation (5), applied to similarities instead of distances.

Figure 1 shows the effect of this function σ_α when matching features between two images, for different values of the exponent α and of the threshold τ. The descriptor similarity, measured by σ_α, is displayed in different colors. A larger α increases the selectivity and drastically down-weights false correspondences. This advantageously replaces hard thresholding as initially proposed in HE [18].

Choice of ϕ. We consider a non-approximate representation of the intermediate vector representation $\phi(x)$ in Equation (14), and adopt a choice similar to VLAD by using the ℓ_2-normalized residual $\hat{r}(x)$, defined as

$$\hat{r}(x) = \frac{x - q(x)}{\|x - q(x)\|}. \tag{19}$$

Our SMK kernel is obtained by setting $\sigma = \sigma_\alpha$ and $\phi = \hat{r}$ in Equation (14), as

$$\text{SMK}(\mathcal{X}_c, \mathcal{Y}_c) = \sum_{x \in \mathcal{X}_c} \sum_{y \in \mathcal{Y}_c} \sigma_\alpha(\hat{r}(x)^\top \hat{r}(y)), \tag{20}$$

It differs from HE in that it uses the normalized residual instead of binary vectors. It also differs from VLAD, considered as a matching function, by the selectivity function σ and because we normalize the residual vector. These differences are summarized in Table 1.

4.3 Aggregated selective match kernel ASMK

SMK weights the contributions of individual matches with a non-linear function. We now propose to apply a selective function *after* aggregating the different vectors per cell. Aggregating the vectors per cell has the advantage of producing a more compact representation.

Our ASMK kernel is constructed as follows. The residual vectors are summed as in VLAD, producing a single representative descriptor per cell. This sum is subsequently ℓ_2-normalized. The ℓ_2-normalization ensures that the similarity in input of σ always lies in the range $[-1, +1]$. It means that

$$\Phi(\mathcal{X}_c) = \hat{V}(\mathcal{X}_c) = V(\mathcal{X}_c)/\|V(\mathcal{X}_c)\| \tag{21}$$

describes all the descriptors assigned to the cell c. The selectivity function σ_α is applied after aggregation and normalization, therefore the matching kernel M_A becomes

$$\text{ASMK}(\mathcal{X}_c, \mathcal{Y}_c) = \sigma_\alpha\left(\hat{V}(\mathcal{X}_c)^\top \hat{V}(\mathcal{Y}_c)\right). \tag{22}$$

The database vectors $\hat{V}(\mathcal{X}_c)$ are computed off-line.

Figure 2 illustrates several examples of features that are aggregated with a small or a large codebook. In the latter case, they commonly correspond to repeated structure and textured regions. Such bursty features appear in most urban images, and their matches usually dominate the image level similarity. ASMK handles this by keeping only one representative instance of all bursty descriptors, which , due to normalization, is equal to the normalized mean residual.

Fig. 2 Examples of features mapped to the same visual word, finally being aggregated. Examples shown for a small codebook of 128 visual words (top), which is a typical size for VLAD, and a large one of size 65k (bottom), which is a typical size for ASMK. Each visual word is drawn with a different color. Top 25 visual words are drawn, based on the number of features mapped to them.

Normalization per visual word was recently proposed by a concurrent work [2] with comparatively small vocabularies. The choice of normalizing our vector representation resembles binary BOW [42] or max pooling [6] which both tackle burstiness by accounting at most one vote per visual word. Aggregating without normalizing still allows bursty features to dominate the total similarity score.

4.4 Binarization — SMK* and ASMK*

HE relies on the binary vector b_x instead of residual $r(x) = x - q(x)$. Although the choice of binarization was adopted for the sake of compactness, a question arises: What is the performance of the kernel if the full vector are employed instead? This is what has motivated us to develop the SMK and ASMK match kernels, which rely on full d-dimensional descriptors. However, these kernels are costly in terms of memory. That is why we also develop their binary versions (denoted with an additional *) in this section.

SMK* and ASMK*. The approximated version SMK* of SMK is similar to HE, the only difference is the inner product formulation and the choice of the selectivity function σ_α in Equation (18):

$$\text{SMK}^\star(\mathcal{X}_c, \mathcal{Y}_c) = \sum_{x \in \mathcal{X}_c} \sum_{y \in \mathcal{Y}_c} \sigma_\alpha\left(\hat{b}_x^\top \hat{b}_y\right). \tag{23}$$

It is an approximation of the full descriptor model of Equation (20), which uses the binary vector $\hat{b}$ instead of $\hat{r}$.

Similarly, the approximation ASMK* of the aggregated version ASMK is obtained by binarizing $V(\mathcal{X}_c)$ before applying the selectivity function:

$$\text{ASMK}^\star(\mathcal{X}_c, \mathcal{Y}_c) = \sigma_\alpha\left\{\hat{b}\left(\sum_{x \in \mathcal{X}_c} r(x)\right)^\top \hat{b}\left(\sum_{y \in \mathcal{Y}_c} r(y)\right)\right\}, \tag{24}$$

where b is an element-wise binarization function $b(x) = +1$ if $x \geq 0, -1$ otherwise. Note that the residual is here computed with respect to the median as in HE, and not the centroid. Moreover, in SMK* and ASMK* all descriptors are projected using the same projection matrix as in HE.

Remark: In LSH, the Hamming distance gives an estimate of the cosine similarity [7] between original vectors (through $\arccos$ function). The differences with HE are that (i) LSH is based on a set of random projections, whereas HE uses a randomly oriented orthogonal basis; (ii) HE binarizes the vectors according to their projected median values.

5 Aggregation across images

ASMK and ASMK* aggregate local descriptors per image and manage to handle bursty matches. As we show in our experiments they improve performance, while at the same time memory requirements are reduced. We further propose an approach to identify local features of different images that correspond to the same physical structure, *i.e.* the same object part. We aggregate their vector representation and index a single vector per group of similar features. In this fashion, we further compress the indexing structure, offer enhanced local representation derived from multiple similar images, and implicitly perform feature augmentation [49]. We adopt this method only with the binarized representation for scalability and efficiency reasons.

Given a set of images indexed with ASMK*, we cross-match all indexed descriptors and find pairs with similarity above or equal to threshold τ_l. In practice, we cross match only features assigned to the same visual word. We further examine whether each match originates from images that are globally similar. That is, we discard a match if it refers to a pair of images that have less than τ_g matches in common.

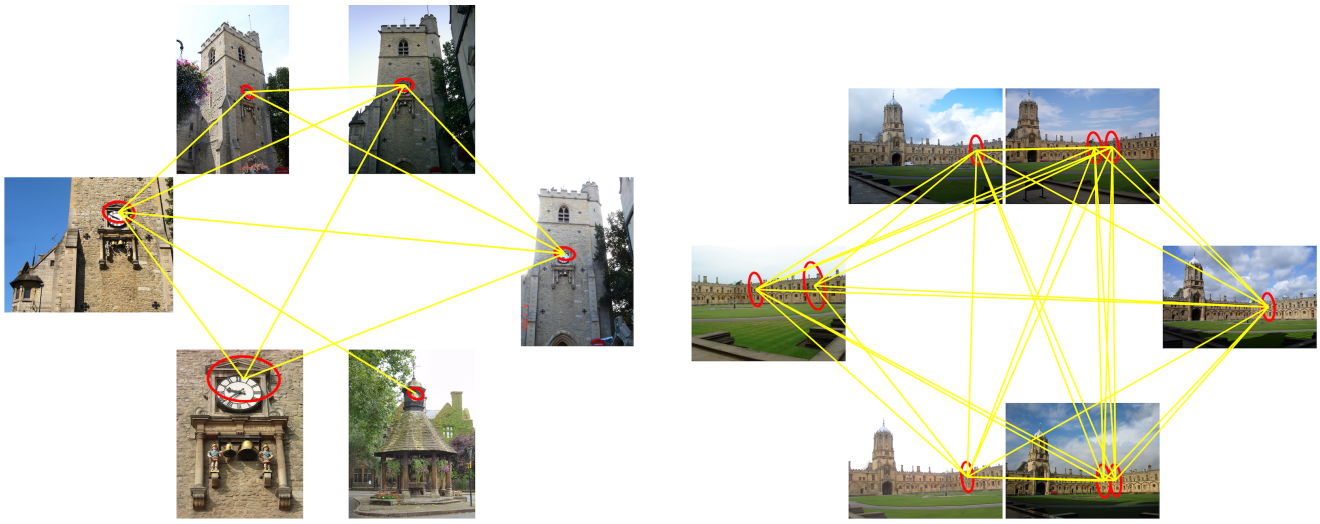


Fig. 3 Examples of connected components discovered on Oxford5k ($\tau_l = 0.375$ and $\tau_g = 5$). Connected local features form a component.

As a consequence, false positive matches are eliminated. We refer to thresholds τ_l and τ_g, as *local* and *global consistency thresholds*, respectively. They correspond to the descriptor level and image level matching.

The identified set of matches can be seen as an undirected graph, where nodes are indexed descriptors and edges are the matches. We then find the connected components of this graph, where each component corresponds to a set of similar local descriptors. We assign a single vector to each component, in analogy to the single vector per visual word (ASMK*). This vector is constructed by aggregating all binary vectors of the component and binarizing once more. More formally, the vector assigned to component $\mathcal{G}$ is

$$\hat{b}_{\mathcal{G}} = \hat{b}\left(\sum_{x\in\mathcal{G}} \hat{b}_x\right). \tag{25}$$

This process corresponds to majority voting per dimension.

During retrieval, when a query descriptor y matches a component $\mathcal{G}$, that is $\hat{b}_y^\top \hat{b}_{\mathcal{G}} \geq \tau$, the score of all images associated with this component is increased. This is not necessarily true when indexing features per image (ASMK*) independently. We refer to this method as *inter-image* ASMK* (i-ASMK*).

Observe that in contrast to ASMK* and (24), we aggregate vectors that are already binarized in (25). Our experiments show that this does not decrease performance compared to aggregating full descriptor vectors.

In Figure 3 we show examples of connected components found on images of Oxford5k. It appears that matching of binary signatures with the use of large similarity threshold provides a very fast way to identify true positive matches without any geometry in the loop.

6 Experiments

This section describes some implementation details and introduces the datasets and evaluation protocol used in our experiments. We present experiments for measuring the impact of the kernel parameters, and compare our methods against state-of-the-art methods. We finally evaluate our scalable variant (ASMK*) on a large scale location recognition.

6.1 Implementation and experimental setup

Datasets. We evaluate the proposed methods on 3 publicly available datasets for image retrieval, namely Holidays [20], Oxford Buildings [36] and Paris [37]. Evaluation measure is the mean Average Precision (mAP). Due to the randomness introduced to the binarized methods (SMK* and ASMK*) by the random projection matrix, the same as the one used in the original Hamming Embedding, we create 3 independent inverted files and measure the average performance.

All of the three aforementioned datasets include the query images in the indexed dataset. In that case, i-ASMK* has the "unrealistic" advantage of employing during the off-line process the query images, which are also used for evaluation. The same holds in previous approaches that follow a similar off-line matching across images [1, 12], without adopting a more proper protocol.

Therefore, we derive new datasets by excluding the query images from the dataset to be indexed. We refer to those as Oxford5k$_{\setminus \mathrm{q}}$, Paris6k$_{\setminus \mathrm{q}}$ and Holidays$_{\setminus \mathrm{q}}$. Note that these datasets contain less images than the original ones and the scores reported are not directly comparable to previously reported scores. We apply i-ASMK* on these datasets and also on the original ones to compare with previous approaches.

We additionally evaluate our approach on San Francisco landmarks dataset [8] for place recognition. The dataset consists of 1.06M perspective images derived from panoramas. In particular, we use the perspective central images (PCI) of this dataset. We report recall on the top ranked images.

Features. We have used the Hessian-Affine detector to extract local features. For Oxford and Paris datasets, we have used the Hessian-Affine detector of Perdoch *et al.* [32], which includes the gravity vector assumption and improves retrieval performance. Most of our experiments use the default detector threshold value. We also consider the use of lower threshold values to derive larger sets of features, and show the corresponding benefit in search quality, at the cost of a memory and computational overhead.

We use SIFT descriptors and component-wise square-rooting [1, 16]. This has proven to yield superior performance at no cost. In more details, we follow the approach [16] in which component-wise square rooting is applied and the final vector is ℓ_2-normalized. We also center the SIFT descriptors. Our SIFT descriptor post-processing is the same as the one of Tolias and Jégou [46].

For the San Francisco dataset we have adopted the standard choice of considering upright objects and extracting upright Hessian-Affine features. Due to low resolution images we have set the cornerness threshold equal to 100, compared to the default which is 500. This choice provides more features, known to improve performance [45, 46]. Root-SIFT [1, 16] is adopted for this dataset.

Vocabularies. We have used flat k-means to create our visual vocabularies. These are always trained on an independent dataset, different from the one indexed and used for evaluation each time. Using visual vocabularies trained on the evaluation dataset yields superior performance [36, 1] but is more prone to over-fitting. Vocabularies used for Oxford are trained on Paris, and vice versa, while the ones used for Holidays are trained on an independent set of images downloaded from Flickr. Unless stated otherwise, we use a vocabulary of 65k visual words.

An exception is the San Francisco dataset, where we follow the standard choice of creating the vocabulary from a random subset of the 1.06M images. We randomly sample 10M descriptors coming from 30k random images.

Inverted files. In contrast to VLAD, we apply our methods with relatively large vocabularies aiming at best performance for object retrieval, and use an inverted file structure to exploit the sparsity of the BOW based representation. With SMK and ASMK, each dimension of vectors $\phi(x)$ or $\Phi(\mathcal{X}_c)$ respectively, is uniformly quantized with 8 bits and stored in the inverted file. Correspondingly, a binary vector of 128 dimensions is stored along with SMK* and ASMK*.

Multiple assignment. We combine our methods with multiple assignment (MA) [20], which is applied on query side only. We replicate each descriptor vector and assign each instance to a different visual word. When it is stated that multiple assignment is used in our experiment, 5 nearest visual words are used. Single assignment will be referred to as SA.

Burstiness. The non-aggregated versions of the proposed methods allow multiple matches for a given visual word. Thus, we combine them with the intra-image burstiness normalization [19]. This is done to compare to our aggregated methods, which also deal with the burstiness phenomenon. We will refer to burstiness normalization as BURST.

Query expansion. We combine our methods with local visual query expansion [46] to further improve the performance. We employ the variant that is not using any geometrical information[1]. This method is referred to as Hamming Query Expansion (HQE). A brief description follows. The number of correspondences using a stricter similarity threshold ($\tau = 0.5$) are enumerated for the 100 top ranked images. The ones with at least 5 correspondences, when MA is used, are considered as relevant. We collect visual words of all relevant images, sort them based on the number of verified images in which they appear and select the top ranked ones. Descriptors assigned to those visual words are merged with the query features, and aggregation per visual word is applied once more. The new expanded query is of the same nature as the original one and can be issued to the same indexing structure.

Aggregation. For the aggregated methods descriptors of database images are aggregated off-line and then stored in the inverted file. On query time, query descriptors are aggregated in the same way. In the case of multiple assignment, aggregation is similarly applied once the aforementioned replication of descriptors is performed.

6.2 Impact of the parameters

Parameter α. Figure 4 shows the impact of the parameter α associated with our selectivity function. It controls the balance between strong and weaker matches. Setting $\alpha = 1$ corresponds to the linear weighting function used by VLAD. The weighting function significantly improves the performance in all cases. In the rest of our experiments, $\alpha = 3$ as a compromise for good performance across all datasets.

Threshold τ. We evaluate the performance on the Oxford dataset for different values of the threshold τ. Figure 5 shows that the performance is stable for small threshold values. In the rest of our experiments we will set the threshold value equal to 0, maintaining best performance but also reducing the number of matches obtained from the inverted file.

[1] This is in contrast to our previous work [45], where we have combined ASMK* with the geometry-based variant.

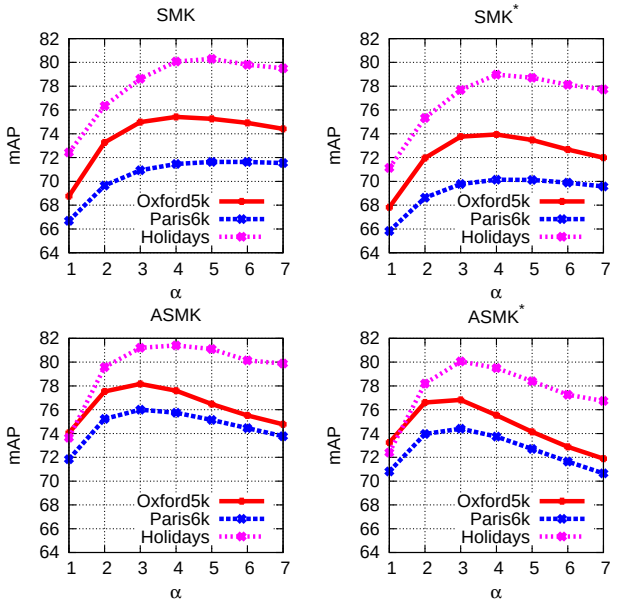


Fig. 4 Impact of parameter α for SMK and ASMK (left) and their binarized counterparts (right). In these experiments, $\tau = 0$.

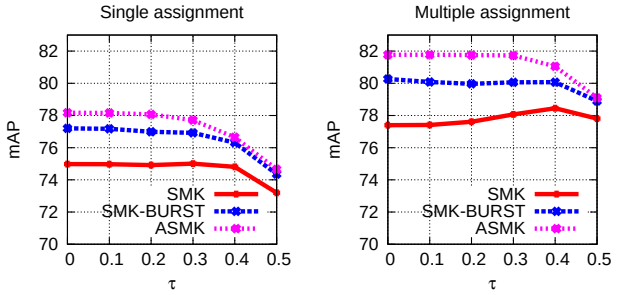


Fig. 5 Impact of threshold value τ on Oxford dataset for SMK, SMK with burstiness normalization and ASMK. Results for single (left) and multiple (right) assignment is shown.

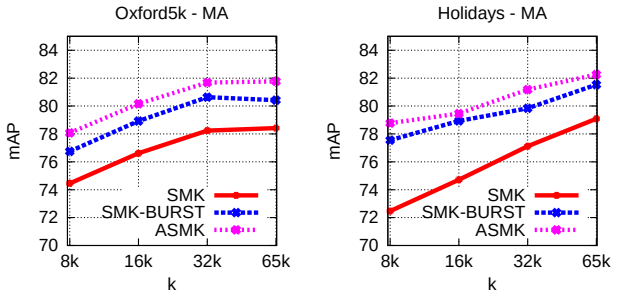


Fig. 6 Impact of vocabulary size k measured on Oxford5k and Holidays datasets. Multiple assignment is used.

Remark also that ASMK outperforms SMK combined with burstiness normalization [19]

Vocabulary size k. We evaluate our proposed methods for different vocabulary sizes and present performance in Figure 6. ASMK outperforms SMK combined with burstiness normalization. We have computed VLAD with the 8k vocabulary, which achieves 65.5 mAP on Oxford5k with a vector representation of 8192×128 dimensions. SMK and ASMK with single assignment and the 8k vocabulary achieve 74.2 and 78.1 respectively.

k	8k	16k	32k	65k
Oxford	69 %	78 %	85 %	89 %
Paris	68 %	76 %	82 %	86 %
Holidays	55 %	65 %	73 %	78 %

Table 2 Ratio of memory requirements after aggregation (ASMK or ASMK⋆) to the ones before aggregation (SMK or SMK⋆), for various vocabulary sizes.

We have measured the amount of descriptors being aggregated in each case by the *memory ratio* which is defined as the ratio of the total number of descriptors indexed after aggregation to the ones before aggregation. The memory savings are presented in Table 2. Our aggregated scheme not only improves performance, but also saves memory.

Larger feature sets. We have conducted experiments using lower detector threshold values than the default one, thus deriving a larger set of features per image. The performance is compared between the two features sets in Table 3, showing that using more features yields superior performance in all cases. The use of the selectivity function allows the use of more features which also includes more false matches, but these are properly down-weighted.

6.3 Inter-image aggregation (i-ASMK⋆)

Local and global consistency thresholds. We conduct experiments for different values of local (τ_l) and global (τ_g) consistency thresholds. We present performance in Figure 7 for Oxford5k$_{\setminus q}$, Paris6k$_{\setminus q}$ and Holidays$_{\setminus q}$. A significant boost is achieved except for Holidays$_{\setminus q}$, where the improvement is quite limited. This is expected as there are several groups with just two relevant images. After removing the query itself only one remains in the dataset and nothing can be cross-matched with our method for those groups.

A more relaxed local similarity threshold requires a stricter global one and vise versa. It is convenient to choose a strict local similarity threshold, as the complexity is reduced from the very first step of process (reduced number of collect matches). For the rest of our experiments we set $\tau_l = 0.375$ and $\tau_g = 10$. Note that the values of local threshold τ_l chosen in Figure 7 correspond to a Hamming distance of 48, 40, and 32. For instance, $\tau_l = 0.5$ corresponds to Hamming distance 32 since $(32 - 64)/64 = 0.5$.

We compare memory requirements of component vector indexing to ASMK⋆, where all extracted descriptors are indexed independently. Results are presented in Figure 8. Memory usage is slightly decreased compared to ASMK⋆, which also offers significant compression. We do not take

Dataset	Oxford5k				Paris6k				Holidays			
	Small		Large		Small		Large		Small		Large	
Method	SMK	ASMK	SMK	ASMK	SMK	ASMK	SMK	ASMK	SMK	ASMK	SMK	ASMK
SA	74.9	78.1	78.5	82.0	70.9	76.0	73.2	78.7	78.6	81.2	84.0	**88.0**
MA	77.4	81.7	79.3	**83.8**	71.8	78.2	74.2	**80.5**	79.0	82.2	82.9	86.5
Method	SMK⋆	ASMK⋆	SMK⋆	ASMK⋆	SMK⋆	ASMK⋆	SMK⋆	ASMK⋆	SMK⋆	ASMK⋆	SMK⋆	ASMK⋆
SA	73.7	76.4	77.5	80.3	69.8	74.4	72.0	77.2	77.7	80.0	83.1	86.5
MA	77.4	80.4	78.2	82.7	70.9	77.0	73.0	79.3	77.8	81.1	81.0	84.4
#features	12.5M	11.2M	21.9M	19.2M	15.0M	13.0M	25.1M	21.5M	4.4M	3.5M	16.7M	12.0M

Table 3 Performance evaluation for different feature set sizes, extracted by using different detector threshold values. Small = set with the default threshold, Large = set with lower threshold. Number of features indexed without (SMK-SMK⋆) and with (ASMK-ASMK⋆) aggregation are reported. Performance for single (SA) and multiple (MA) assignment. **These results are without spatial verification and without QE.**

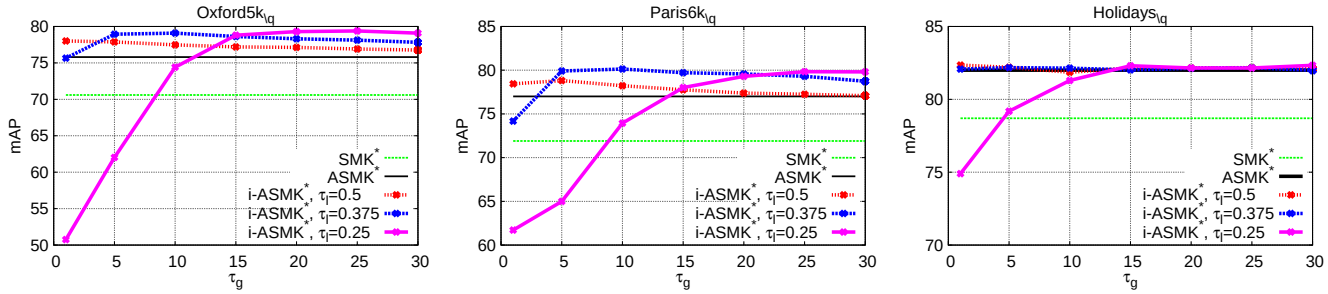


Fig. 7 Impact of parameters of i-ASMK⋆ measured on Oxford5k$_{\setminus q}$, Paris6k$_{\setminus q}$ and Holidays$_{\setminus q}$ datasets. Multiple assignment is used.

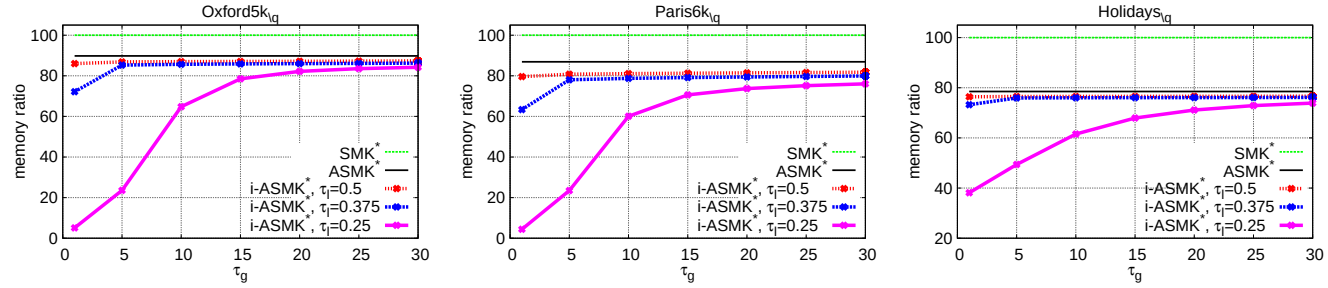


Fig. 8 Memory compression achieved by i-ASMK⋆ measured on Oxford5k$_{\setminus q}$, Paris6k$_{\setminus q}$ and Holidays$_{\setminus q}$ datasets. We report relative memory usage with respect to that of SMK⋆ (all initial descriptors indexed).

into account storage space for image ids. Note that i-ASMK⋆ has the same needs with ASMK⋆ in this respect. Since there are less binary vectors to be compared, i-ASMK⋆ does not increase query time.

Binary vector aggregation. We conduct an experiment to evaluate the loss by aggregating already binarized vectors with (25), compared to aggregating the full descriptors and then binarizing them. We present results in Table 5, where it is shown that any differences in performance are insignificant. This further supports our choice of working on binary vectors. However, restoring the full descriptors would not be efficient or preferred in any case.

Larger feature sets. In analogy to the experiment presented in Table 3, we compare i-ASMK⋆ performance measured on the standard and a larger feature set. Results are shown in Table 4, where a significant improvement using the large sets is achieved once more.

Component statistics. On Oxford105k$_{\setminus q}$ we initially discover 599M feature matches. After the global consistency check only 12.8M of them are left. In Figure 9 we show the distribution of similarity value of the detected matches, before and after applying the global consistency check. A large number of weak matches are identified in pairs of non-similar images and are finally removed.

Finally, we discover 5.6M connected components with an average size equal to 2.4. Out of the 21.2M descriptors indexed with ASMK⋆, 6.3% of them belong to some component. The rest are indexed individually. In order to give an insight on the detected components, we further report that 87.8% of the components are associated with 2 images, while 98.5% with at most 5 images.

Dataset	Oxford5k$_{\setminus q}$		Paris6k$_{\setminus q}$		Holidays$_{\setminus q}$	
	Small	Large	Small	Large	Small	Large
#features	10.6M	18.1M	11.7M	19.4M	2.3M	7.7M
SA	75.2	79.8	78.5	81.6	81.4	**87.6**
MA	79.1	**80.8**	80.1	**82.9**	82.2	85.6

Table 4 Performance evaluation of i-ASMK* for different feature set sizes (same as the ones of Table 3). Number of features indexed after aggregation across multiple images are reported. Performance for single (SA) and multiple (MA) assignment.

Dataset	MA	F	Oxford5k$_{\setminus q}$	Paris6k$_{\setminus q}$	Holidays$_{\setminus q}$
i-ASMK*		×	75.7	78.5	81.5
i-ASMK*			75.2	75.8	81.4
i-ASMK*	×	×	79.1	80.2	82.2
i-ASMK*	×		79.1	80.1	82.2

Table 5 Performance evaluation for i-ASMK* comparing aggregation of binary vectors to that of full descriptors. F = aggregate full descriptors. Note that in the rest of our experiments with i-ASMK*, we aggregate binary vectors with Equation (25).

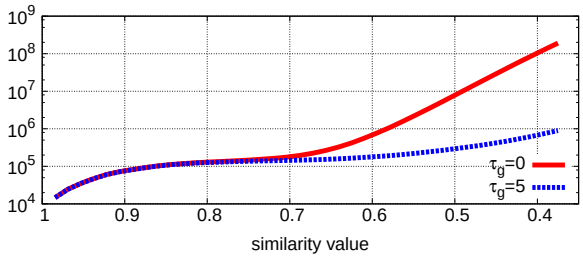


Fig. 9 Distribution of similarity value over the set of matches identified on Oxford105k$_{\setminus q}$ ($\tau_l = 0.375$). Distribution is shown before ($\tau_g = 0$), and after ($\tau_g = 5$) applying the global consistency check.

6.4 Comparison to the state of the art

Table 6 summarizes the performance of our methods for single and multiple assignment and compares to state of the art methods. Only our binarized methods are scalable for Oxford105k. ASMK achieves a better performance than the binarized ASMK* and outperforms all other methods.

We have evaluated Hamming Embedding with binary signatures of 128 bits using exactly the same descriptors as in our methods. This is denoted by HE(128bits) in Table 6 and is included for a more fair comparison, while all other scores followed by a reference to prior work are the scores reported by the authors. The difference of HE to SMK* is the use of a Gaussian selectivity function and the image normalization factor. The latter is equivalent to the one of BOW, not enforcing constant self similarity in the case of HE.

It appears that the binarized method offers a more efficient alternative without big loss in performance (1.3 on Oxford5k and 1.2 on Holidays). Apart from binary vectors, there exist more options to encode the aggregated residuals, such as methods based on product quantization [21,25]. We do not consider them here in order to seek a better trade-off between performance and efficiency, since the space for improvements appears to be tight, at least at this scale.

We also compare i-ASMK* to state of the art methods that involve query expansion in the loop. We combine i-ASMK* with HQE to further boost performance and outperform previous approaches in all datasets except Paris6k. Results are presented in Table 7. We evaluate both on the original dataset, in order to compare to other methods, and on the ones that exclude the query images, which is more realistic.

The combination with HQE performs poorly on Holidays dataset, which is due to fact that there are few similar images per query. This agrees with previous findings: query expansion based on geometry just slightly improves performance on Holidays in [30], while a similar drop is observed for approaches not using any geometry [38].

6.5 Timings

Query time for ASMK* on Oxford105k measured on a single core processor is 42 ms with single, and 177 ms with multiple assignment. The measurements include the aggregation and binarization operations for the query image, while feature extraction and quantization time are excluded. The reported query time corresponds to $\tau = 0.1$ (Hamming distance equal to 56), while our default threshold for experiments is $\tau = 0$. The choice of $\tau = 0$ has been adopted for maximum performance. However, performance has already saturated close to 0 and faster options are possible with similar performance. Performance drop, using $\tau = 0.1$, compared to the results shown in Table 6 (default value for τ) is 0.2 with SA and 0.03 with MA, which is insignificant.

We have measured the processing time for the individual steps of the off-line process to identify connected compo-

Dataset	MA	Oxf5k	Oxf105k	Par6k	Holidays
HE [20]		51.7	-	-	74.5
HE [20]	×	56.1	-	-	77.5
HE-BURST [17]		64.5	-	-	78.0
HE-BURST [17]	×	67.4	-	-	79.6
AHE-BURST [17]		66.6	-	-	79.4
AHE-BURST [17]	×	69.8	-	-	81.9
Fine vocab [30]	×	74.2	67.4	74.9	74.9
Rep. structures [47]	×	65.6	-	-	74.9
Locality [43]	×	77.0	-	-	78.7
HE(128bits)-BURST	×	78.2	70.4	73.8	80.4
ASMK*		76.4	69.2	74.4	80.0
ASMK*	×	80.4	**75.0**	77.0	81.0
ASMK		78.1	-	76.0	81.2
ASMK	×	**81.7**	-	**78.2**	**82.2**

Table 6 Performance comparison to state-of-the-art methods ($\alpha = 3$, $\tau = 0$, $k = 65$k). Note that both SMK and ASMK rely on full descriptors and do not scale to Oxford105k. Memory used by SMK* (reps., ASMK*) is equal (resp., lower) than in HE. The best ASMK* variant is faster than HE (less features after aggregation).

nents with i-ASMK⋆. On Oxford105k we need around 156 minutes to collect all pairwise matches and 506 seconds for filtering based on global consistency. Finally, forming the graph and finding connected components takes 53 seconds.

6.6 Place recognition

We apply the proposed ASMK⋆, which allows for scalability, and i-ASMK⋆ on San Francisco dataset. We measure recall at the top N retrieved images, most probably to depict the query landmark. Following the standard protocol [8], the recall of each query is 1 if at least one correct image is found among the N top ranked ones, and 0 otherwise; this measurement is averaged over all queries. We compare to the recent work of Torii *et al.* [47], the graph based query expansion method of Zhang *et al.* [52] and the work of Chen *et al.* [8]. The results for the latter are as reported by Torii *et al.* [47]. We further report performance for HE with 128 bits combined with burstiness normalization using the same descriptors as in our methods. Finally, we compare to the concurrent work of Arandjelovic and Zisserman [3]. It is based on HE and proposes to adjust the selectivity function according to descriptor space density. We compare to their variant mentioned as *DisLoc*.

Results are presented in Figure 10, where we set a new state of the art on this dataset. Notably, we achieve recall equal to 74.5 and 86.3 at the top 1 and 50 images, respectively. Our two methods perform similarly. There is a small drop for i-ASMK⋆, however it offers further memory compression compared to ASMK⋆. In Figure 11 we present performance evaluated using the fixed ground truth for San Francisco dataset, which was relased on April 2014.

Dataset	MA	Oxf5k	Oxf105k	Par6k	Holidays
Total recall II [9]		82.7	76.7	80.5	-
RNN [12]		81.4	76.7	80.3	-
Three things [1]		80.9	72.2	76.5	-
Fine voc+QE [30]	×	84.9	79.5	82.4	75.8
Q.adap+RNN [38]	×	85.0	81.6	**85.5**	80.1
i-ASMK⋆		81.9	77.8	79.7	80.5
i-ASMK⋆	×	84.5	80.6	81.1	**81.3**
i-ASMK⋆ +HQE	×	**86.9**	**85.3**	85.1	80.4
Datasets excluding the query images					
Dataset	MA	Oxf5k$_{\setminus q}$	Oxf105k$_{\setminus q}$	Par6k$_{\setminus q}$	Holidays$_{\setminus q}$
SMK⋆	×	70.6	56.7	69.7	78.7
ASMK⋆	×	75.8	68.9	76.2	82.0
i-ASMK⋆		75.4	68.5	78.5	81.4
i-ASMK⋆	×	79.2	73.4	80.2	**82.7**
i-ASMK⋆ +HQE	×	**81.7**	**78.6**	**84.0**	82.3

Table 7 Performance comparison to state-of-the-art methods on query expansion and augmentation. We report our performance on the original datasets and compare with state of the art methods. We further report performance on the modified datasets that exclude the query images. The latter, but not the former, forms a realistic dataset for methods that perform off-line cross matching [12,1,38]. i-ASMK⋆ is applied with $\tau_l = 0.375$ and $\tau_g = 10$.

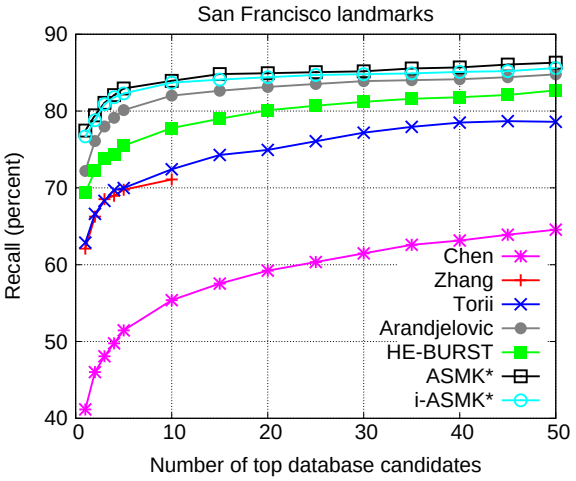


Fig. 10 Evaluation on the San Francisco dataset. We measure the fraction of correctly recognized queries versus the number of top ranked images considered.

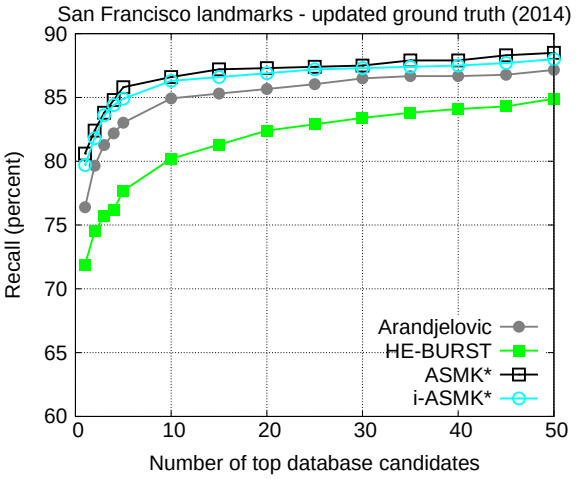


Fig. 11 Evaluation on the San Francisco dataset using the fixed ground truth dataset released on April 2014.

7 Conclusions

This paper draws a framework for well known matching kernels such as BOW, HE and VLAD. We build a common model in which we further incorporate our matching kernels sharing the best properties of HE and VLAD. We exploit the use of a selectivity function and show how aggregation per visual word can deal with burstiness.

We effectively apply the same aggregation principle on features of multiple images. This method offers significant increase in performance, while at the same time enjoys slight decrease of memory usage. Finally, our methods exhibit superior performance than state of the art on large scale image retrieval and place recognition.

References

1. Arandjelovic, R., Zisserman, A.: Three things everyone should know to improve object retrieval. In: CVPR (2012)
2. Arandjelović, R., Zisserman, A.: All about VLAD. In: CVPR (2013)
3. Arandjelović, R., Zisserman, A.: DisLocation: Scalable descriptor distinctiveness for location recognition. In: ACCV (2014)
4. Avrithis, Y., Kalantidis, Y., Tolias, G., Spyrou, E.: Retrieving landmark and non-landmark images from community photo collections. In: ACM Multimedia (2010)
5. Bo, L., Sminchisescu, C.: Efficient match kernel between sets of features for visual recognition. In: NIPS (2009)
6. Boureau, Y., Bach, F., Lecun, Y., Ponce, J.: Learning mid-level features for recognition. In: cvpr (2010)
7. Charikar, M.: Similarity estimation techniques from rounding algorithms. In: ACM Symposium on Theory of Computing (2002)
8. Chen, D.M., Baatz, G., Koser, K., Tsai, S.S., Vedantham, R., Pylvanainen, T., Roimela, K., Chen, X., Bach, J., Pollefeys, M., Girod, B., Grzeszczuk, R.: City-scale landmark identification on mobile devices. In: CVPR (2011)
9. Chum, O., Mikulik, A., Perdoch, M., Matas, J.: Total recall II: Query expansion revisited. In: CVPR (2011)
10. Chum, O., Philbin, J., Sivic, J., Isard, M., Zisserman, A.: Total recall: Automatic query expansion with a generative feature model for object retrieval. In: ICCV (2007)
11. Csurka, G., Dance, C., Fan, L., Willamowski, J., Bray, C.: Visual categorization with bags of keypoints. In: ECCV Workshop Statistical Learning in Computer Vision (2004)
12. Danfeng, Q., Gammeter, S., Bossard, L., Quack, T., Gool, L.V.: Hello neighbor: Accurate object retrieval with k-reciprocal nearest neighbors. In: CVPR (2011)
13. Delhumeau, J., Gosselin, P.H., Jégou, H., Pérez, P.: Revisiting the vlad image representation. In: ACM Multimedia (2013)
14. Delvinioti, A., Jégou, H., Amsaleg, L., Houle, M.E.: Image retrieval with reciprocal and shared nearest neighbors. In: VISAPP (2014)
15. Hays, J., Efros, A.A.: Im2gps: estimating geographic information from a single image. In: CVPR (2008)
16. Jain, M., Benmokhtar, R., Gros, P., Jégou, H.: Hamming embedding similarity-based image classification. In: ICMR (2012)
17. Jain, M., Jégou, H., Gros, P.: Asymmetric hamming embedding: Taking the best of our bits for large scale image search. In: ACM Multimedia (2011)
18. Jégou, H., Douze, M., Schmid, C.: Hamming embedding and weak geometric consistency for large scale image search. In: ECCV (2008)
19. Jégou, H., Douze, M., Schmid, C.: On the burstiness of visual elements. In: CVPR (2009)
20. Jégou, H., Douze, M., Schmid, C.: Improving bag-of-features for large scale image search. IJCV **87**(3), 316–336 (2010). URL http://lear.inrialpes.fr/pubs/2010/JDS10a
21. Jégou, H., Douze, M., Schmid, C.: Product quantization for nearest neighbor search. Trans. PAMI **33**(1), 117–128 (2011). URL http://lear.inrialpes.fr/pubs/2011/JDS11
22. Jégou, H., Douze, M., Schmid, C., Pérez, P.: Aggregating local descriptors into a compact image representation. In: CVPR (2010)
23. Ji, R., Duan, L., Chen, J., Yao, H., Yuan, J., Rui, Y., Gao, W.: Location discriminative vocabulary coding for mobile landmark search. IJCV pp. 1–25 (2012)
24. Johns, E., Yang, G.Z.: From images to scenes: Compressing an image cluster into a single scene model for place recognition. In: ICCV (2011)
25. Kalantidis, Y., Avrithis, Y.: Locally optimized product quantization for approximate nearest neighbor search. In: CVPR. Columbus, Ohio (2014)
26. Knopp, J., Sivic, J., Pajdla, T.: Avoiding confusing features in place recognition. In: ECCV (2010)
27. Li, Y., Crandall, D.J., Huttenlocher, D.P.: Landmark classification in large-scale image collections. In: ICCV (2009)
28. Lowe, D.: Distinctive image features from scale-invariant keypoints. IJCV **60**(2), 91–110 (2004)
29. Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. Trans. PAMI **27**(10), 1615–1630 (2005)
30. Mikulik, A., Perdoch, M., Chum, O., Matas, J.: Learning vocabularies over a fine quantization. IJCV **103**(1), 163–175 (2013)
31. Nistér, D., Stewénius, H.: Scalable recognition with a vocabulary tree. In: CVPR, pp. 2161–2168 (2006)
32. Perdoch, M., Chum, O., Matas, J.: Efficient representation of local geometry for large scale object retrieval. In: CVPR (2009)
33. Perronnin, F., Dance, C.R.: Fisher kernels on visual vocabularies for image categorization. In: CVPR (2007)
34. Perronnin, F., Liu, Y., Sanchez, J., Poirier, H.: Large-scale image retrieval with compressed Fisher vectors. In: CVPR (2010)
35. Perronnin, F., Sánchez, J., Mensink, T.: Improving the Fisher kernel for large-scale image classification. In: ECCV (2010)
36. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching. In: CVPR (2007)
37. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Lost in quantization: Improving particular object retrieval in large scale image databases. In: CVPR (2008)
38. Qin, D., Wengert, C., Van Gool, L.: Query adaptive similarity for large scale object retrieval. In: CVPR (2013)
39. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Information Processing & Management **24**(5), 513–523 (1988)
40. Schindler, G., Brown, M., Szeliski, R.: City-scale location recognition. In: CVPR (2007)
41. Shen, X., Lin, Z., Brandt, J., Avidan, S., Wu, Y.: Object retrieval and localization with spatially-constrained similarity measure and k-nn re-ranking. In: CVPR (2012)
42. Sivic, J., Zisserman, A.: Video Google: A text retrieval approach to object matching in videos. In: ICCV (2003)
43. Tao, R., Gavves, E., Snoek, C.G., Smeulders, A.W.: Locality in generic instance search from one example. In: CVPR (2014)
44. Tolias, G., Avrithis, Y.: Speeded-up, relaxed spatial matching. In: ICCV (2011)
45. Tolias, G., Avrithis, Y., Jégou, H.: To aggregate or not to aggregate: selective match kernels for image search. In: ICCV (2013)
46. Tolias, G., Jégou, H.: Visual query expansion with or without geometry: refining local descriptors by feature aggregation. Pattern Recognition (2014)
47. Torii, A., Sivic, J., Pajdla, T., Okutomi, M.: Visual place recognition with repetitive structures. In: CVPR (2013)
48. Torralba, A., Fergus, R., Weiss, Y.: Small codes and large databases for recognition. In: CVPR (2008)
49. Turcot, P., Lowe, D.G.: Better matching with fewer features: The selection of useful features in large database recognition problems. In: CVPR (2009)
50. Wang, J., Yang, J., K. Yu, F.L., Huang, T., Gong, Y.: Locality-constrained linear coding for image classification. In: CVPR (2010)
51. Wu, Z., Ke, Q., Isard, M., Sun, J.: Bundling features for large scale partial-duplicate web image search. In: CVPR, pp. 25–32 (2009)
52. Zhang, S., Yang, M., Cour, T., Yu, K., Metaxas, D.N.: Query specific fusion for image retrieval. In: ECCV (2012)

VI A comparison of dense region detectors for image search and fine-grained classification

Title: A comparison of dense region detectors for image search and fine-grained classification
Authors: A. Iscen, G. Tolias, P. Gosselin, H. Jégou
Published at: IEEE Transactions on Image Processing 2015

A comparison of dense region detectors for image search and fine-grained classification

Ahmet Iscen*, Giorgos Tolias, Philippe-Henri Gosselin and Hervé Jégou

***Abstract*—We consider a pipeline for image classification or search based on coding approaches like Bag of Words or Fisher vectors. In this context, the most common approach is to extract the image patches regularly in a dense manner on several scales. This paper proposes and evaluates alternative choices to extract patches densely. Beyond simple strategies derived from regular interest region detectors, we propose approaches based on super-pixels, edges, and a bank of Zernike filters used as detectors.**

The different approaches are evaluated on recent image retrieval and fine-grain classification benchmarks. Our results show that the regular dense detector is outperformed by other methods in most situations, leading us to improve the state of the art in comparable setups on standard retrieval and fined-grain benchmarks. As a byproduct of our study, we show that existing methods for blob and super-pixel extraction achieve high accuracy if the patches are extracted along the edges and not around the detected regions.

***Index Terms*—dense keypoints, image retrieval, fine-grained classification, Zernike polynomials**

I. Introduction

LOCAL image description is a popular research topic in computer vision, as it is involved in many applications such as image classification and particular object detection. Extracting local descriptors from an image consists of two steps. The *detection* step selects regions of interest, which are normalized into fixed-size patches. The *description* step produces a vector representation for each of the detected patches. The SIFT descriptor [29] and its RootSIFT extension [3] have been shown to perform very well for most applications. Many descriptors have been introduced in the last years to improve the description speed or descriptor compactness, such as SURF [6], CHOG [10] or BRIEF [9]. The matching accuracy is improved by learning the descriptor design [58][59][49]. In this paper, we are solely interested in the detection stage and therefore adopt the gold-standard SIFT and RootSIFT.

The early works in this line of research have focused on the detection of sparse interest points and regions, typically producing a few thousand descriptors per image. These approaches aim at extracting distinctive and repeatable image parts, such as blobs and corners [29][35][31][6], offering covariance properties: the same regions should be detected under some geometrical transformations. From a historical perspective, the choice of sparse representations was arguably motivated by the lack computational and memory resources. Although such methods perform well and are still widely used for image matching, they are not competitive in other application scenarios such as image classification.

INRIA, Campus Universitaire de Beaulieu 35042 Rennes Cedex FRANCE

Local feature detection has recently shifted towards denser techniques. Dense sampling is an easy way to provide a large number of patches and a better coverage of the objects of interest. Fei-Fei and Perona [15] were the first to show that dense patches leads to better classification accuracy. Nowak *et al*. [38] argue that the key parameter for classification is the number of extracted patches. Similar conclusions hold for other tasks, like fine-grained classification [19] or action recognition in videos [56][22]. Recent works [61][12][51][52] also evidence that methods for image and particular object retrieval, which traditionally rely on sparse regions typically extracted with the Hessian-Affine detector [34], are improved when using a larger set of descriptors.

Nevertheless, regular dense sampling has serious limitations. Uniform sampling of patches ignores the image structure and extracts many uniform and uninformative patches. Additionally, the position of the features is less or not repeatable. This prevents the image engine from employing a spatial verification method, such as RANSAC [17][44], which typically filters out many outliers by enforcing the spatial consistency of the detected regions.

In this paper, our goal is to develop and evaluate dense detection strategies for image retrieval and fine-grained image classification. Our motivation is similar to that of Tuytelaars when she introduced "dense interest points" [53]: we consider solutions in between localized sparse interest points and dense strategies, in order to produce a large number of localized regions. We depart from using traditional evaluation metrics for detector evaluation, which do not reflect the final goal. For instance, the repeatability score reflects the effectiveness in detecting inliers, but is not directly related to the determination of class membership. Instead, we evaluate the performance with the metrics employed for the target application scenario: mean average precision for image retrieval and accuracy for fine-grained classification. We stress that better localized dense patches (see Figure 1) are more important for these tasks than in traditional image classification: the objects are more repeatable and distinguishing between two classes often rely on tiny details that suffer from being loosely localized.

We make the following contributions in this context of dense detection for image retrieval and fine-grained classification:

1) We propose strategies derived from standard interest point detectors (Harris, Hessian and DoG) to extract patches densely. In particular, we modify the detection process by relaxing the standard local maxima criterion so that it focuses on edges and not only corners. This, jointly with the optimization of the scaling factor, is shown to be a key to achieve higher performance.

2) We depart from the typical choice of fitting an ellipse to describe a region of interest extracted by blob detectors (MSER or super-pixels). Instead, we sample patches at several scales along the region's borders. This increases the performance significantly when considering a large number of patches per image. We also show that sampling on the edges produced by a state-of-the-art edge detector [13] offers competitive performance.
3) Finally, we propose two novel response filters to select the patch locations. First, we propose to use a bank of Zernike polynomials [60] as detectors. These filters have been proposed to construct descriptors [48], but to our knowledge not as detectors. Our second strategy is descriptor-oriented: the response for each pixel is simply the norm of local descriptor associated with the patch centered at this location. These two new approaches appear to perform best in most cases of our experiments.

This paper is organized as follows. Section II discusses previous work related to the purpose of our study. Section III describes existing approaches which are part of our evaluation, while in section IV we present modified schemes of existing detectors and the use of pseudo-Zernike polynomials for dense feature detection. Finally in section V we present the outcome of this study on 2 datasets on image retrieval and 4 datasets on fine-grained classification.

II. Related work

A great deal of works has focused on the detection of local interest points [29][6][34][31]. They are rather designed to be appropriate for image matching applications. The standard evaluation metrics for such detectors [35], *e.g.* repeatability, are designed to reflect sufficiency for matching applications. However, this could be far from the final application tasks we focus in this paper, *i.e.* fine-grained classification.

Typically, local interest points are tuned to detect image structures, such as corners and blobs, highly distinctive and repeatable. One of the exceptions is the work of Mikolajczyk *et al.* [33] where they focus on edges to represent objects. Similarly, in this study we argue that such a choice is beneficial for image retrieval and fine-grained classification.

Tuytelaars's work on dense interest points [53] has a motivation close to ours. Initially, a dense grid of patches is considered, and then for each point, local refinement of its location and scale parameters is performed. In particular, the point of maximum *interestingness* is selected within a bounded area. This choice suffers from quantization artifacts, since a true local maximum might not exist in this search area. Apparently, a large number of patches are located on smooth and uninformative regions.

A previous survey closely related to this study is the work of Nowak *et al.* [38]. They are also not interested in the repeatability of patches, but directly measure classification accuracy. According to them, detectors designed to obtain high repeatability perform the same as randomly selected patches. They observe that performance is, up to some extent, an increasing function of the number of points per image. Inspired by their finding, we conduct a similar evaluation and try to study different ways to control the average number of patches. Similarly, Avrithis and Rapantzikos [4] do not restrict comparison to repeatability, but further compared image retrieval performance, while considering the average number of points as a crucial parameter.

One of the parameters we consider in our study is the size of the measurement region for local descriptor extraction. This has been given particular attention by the recent work of Simonyan *et al.* [49], who shows that introducing a scaling factor increases retrieval performance. Similarly, the improved Hessian-Affine detector by Perdoch *et al.* [41] uses a larger measurement region to improve the performance.

In regular sampling local descriptors are agnostically extracted on a dense grid, and as a consequence many uninformative descriptors are derived from smooth regions. A significant improvement is achieved by filtering out descriptors with low ℓ_2-norm [19]. We combine this approach with examined detectors and study its effect on different setups.

Convolutional neural networks are very effective for image classification [27][14] and detection [18]. However, for the tasks addressed in this paper, namely fine-grained classification, image and particular object, this is not (yet) the case. Fisher vectors based on a dense representation and employing medium-sized codebooks and spatial coding achieved much better results in the fine-grained challenge [19]. For image retrieval, recent work demonstrates that CNNs [5] achieve excellent performance for aggressive operating points for compact representations. However, their best reported performance is lower than that of the state-of-the-art Fisher vectors, and outperformed by a large margin by state-of-the-art methods like the selective match kernels [51].

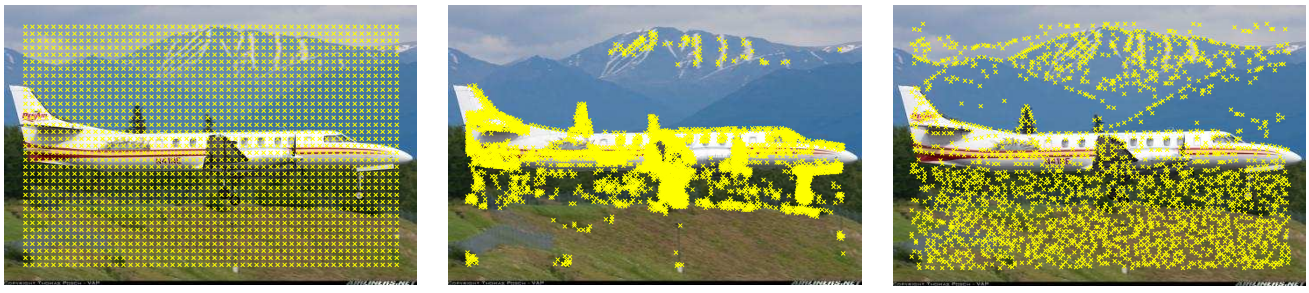


Fig. 1. Locations of patches detected by regular dense sampling (left), by Zernike detector (middle), and by dense ℓ_2-norm based detector (right). We visualize the patches only for the first scale.

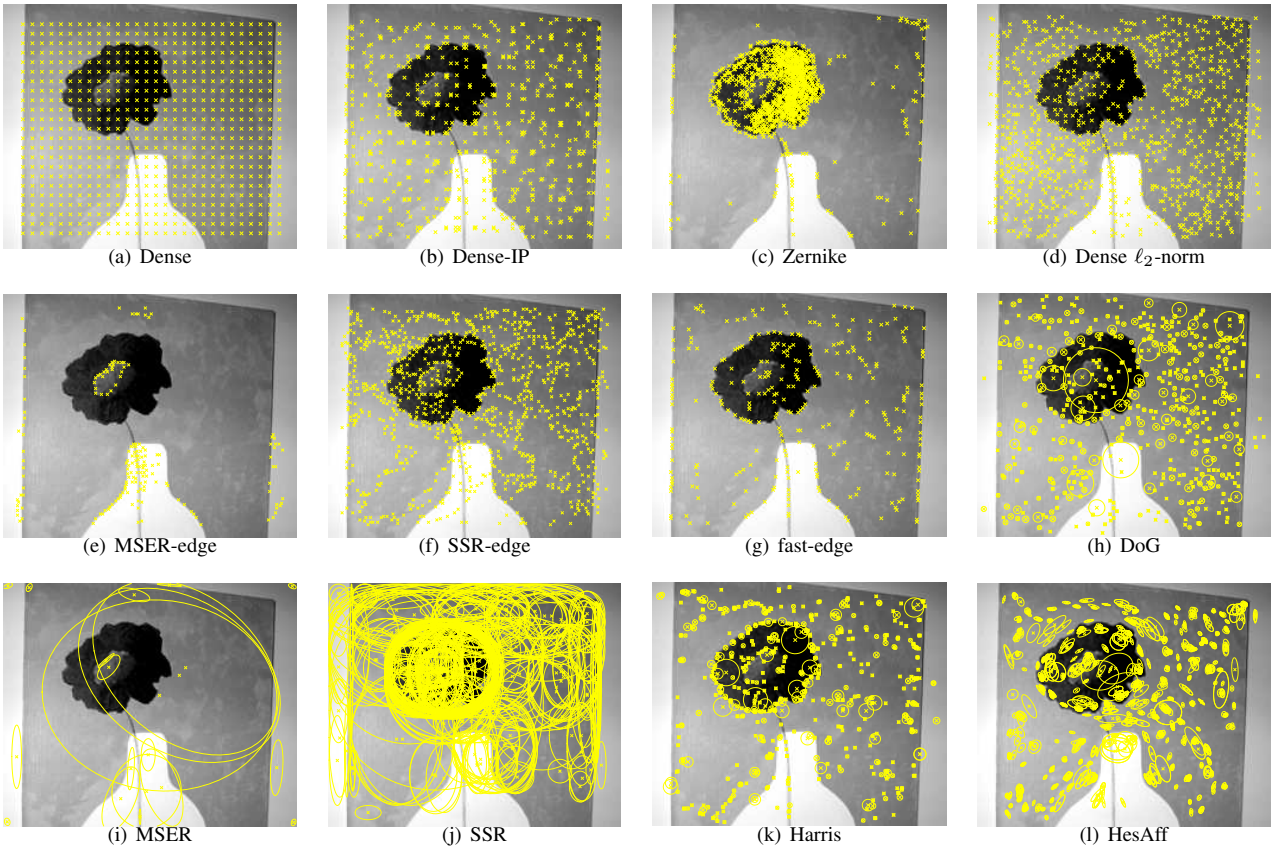


Fig. 2. Visualization of regions detected by each method. For methods with predefined scales, we visualize the center of points detected on the first scale (a)-(g), while for the ones that perform scale selection we draw the corresponding ellipses or circles (h)-(l).

III. Background

In this section we briefly describe existing methods for interest point detection or dense patch sampling, that are part of our study. In all those methods, a local descriptor is extracted from each region of interest and an image is represented by a set of such descriptors. We visualize detected features for all the detectors examined or proposed in this work on Figure 2.

A. Interest points

Harris-Laplace detector. Harris detector localizes corners, based on the fact that gradient values will change in multiple directions around a corner. It uses a scale adapted version of the second moment matrix, known as Harris matrix [20]:

$$M = \sigma_D^2 \cdot g(\sigma_I) * \begin{bmatrix} L_x^2(x,\sigma_D) & L_xL_y(x,\sigma_D) \\ L_xL_y(x,\sigma_D) & L_y^2(x,\sigma_D) \end{bmatrix}, \quad (1)$$

where σ_D is the differentiation scale, σ_I is the integration scale and L_z is the derivative computed in z direction. Differentiation scale σ_D is used to compute the local derivatives with Gaussian kernels, and a Gaussian window with a size σ_I is used to smooth and average the neighborhood around the point. The eigenvalues of this matrix represent the gradient changes in two directions. Consequently, if one eigenvalue is large while the other is small, there exists an edge, whereas if both eigenvalues are large, there exists a corner. The *interestingness* of a point is captured by the *cornerness* function, defined as $cornerness = \det(M) - \alpha \text{trace}^2(M)$, where α is usually set to 0.05.

Extending Harris detector to be scale invariant, Mikolajczyk *et al.* [34] use Laplacian-of-Gaussian (LoG) response and detect local extrema over multiple scales to perform scale selection [28]. This is the well known Harris-Laplace detector. Only points with cornerness value higher than threshold τ are retained and final point locations are chosen by a local maxima search procedure.

Hessian-Affine detector [35], similar to Harris detector, detects image locations that have large derivatives in both directions. Point locations are selected as local maxima of the Hessian matrix determinant, which now constitutes the interestingness measure. The Hessian matrix is defined as

$$H = \begin{bmatrix} L_{xx}(x,\sigma_D) & L_{xy}(x,\sigma_D) \\ L_{xy}(x,\sigma_D) & L_{yy}(x,\sigma_D) \end{bmatrix}, \quad (2)$$

where L_{zz} is the second order partial derivative. All points with interestingness below threshold τ are discarded. Although Hessian and Harris detectors are quite similar, the detected points may be slightly different. In particular, unlike Harris, Hessian detector tends to select locations with texture variations, in addition to corners.

The affine shape of the point is estimated by the eigenvalues of the second moment matrix M. An iterative procedure modifies the point's location, scale and shape until the estimated affine transform is able to map the detected region into one that has equal eigenvalues of its second moment matrix.

MSER *Maximally Stable Extremal Regions* were introduced by Matas *et al*. [31] as a feature detector that tends to localize blobs. An *extremal region* has all its intensity values greater (or less) than the outer region boundary pixels. Then, a sequence of nested extremal regions is considered. Along this sequence scale change between neighboring regions is estimated. Maximally stable are the local minima of this quantity. In this fashion, nested regions are also likely to appear. The detected regions can have very irregular shapes, therefore an ellipse is fitted to detected regions in order to extract local descriptors. A parameter Δ controls the locality of the scale change computation.

Difference of Gaussians (DoG) was originally used for local feature detection by Lowe [29]. DoG is an efficient way to approximate the Laplacian of Gaussian and to detect edges at various image scales. A Gaussian kernel is used to create multiple blurred versions of the image per octave. Simple subtraction of two consecutive blurred images produces the DoG response. Interest points are located in the scale-space as local-maxima in a 3D search area of size 3.

The original SIFT detection algorithm [29] employs DoG, but further applies additional steps to filter out points belonging to edges or low-contrast regions. However, in our experiments with DoG detector, we keep all the points for a denser representation.

B. Dense patch sampling

Regular grid dense sampling. In contrast to interest points, dense sampling methods give less importance to high repeatability and try to provide a dense coverage of the depicted objects. The most popular method is to sample points on a regular grid, every δ_{xy} pixels. Depending on the application δ_{xy} can be really small, such as 3, or quite larger, such as 16. In order to provide some scale tolerance, different scales are considered by following the same procedure at n_σ multiple scales of the image. All the patches from different scales are pooled together in the end. A typical value for n_σ is 5, with 2 scales per octave. We adopt this choice in our experiments.

Dense interest points were introduced by Tuytelaars [53] as a hybrid solution to trade-off between sparse interest point detection and dense sampling. Instead of selecting the center pixel of grid cell, as in regular dense sampling, they conduct local search inside each cell over spatial and scale space. The point with maximum response is kept per cell. Selected points are not necessarily local maxima. Since a single interest point is selected from each cell, patches are very likely to be localized on smooth regions. Moreover, quite a few patches are localized on the cell borders. In the original work, a large local search area of 16×16 pixels $\times$ 8 scale levels is used, while in our study we experiment with smaller search areas in order to supply more points.

IV. Proposed Methods

Our purpose is to provide a dense set of patches without targeting high repeatability. We rather focus on accuracy in retrieval and fine-grained classification. The proposed methods try to provide a nice coverage of the depicted objects, while focusing on edges in addition to corners and blobs.

In this section we present the proposed approaches for dense patch representation. We relax the Harris-Laplace detector to detect mostly edges. We propose to use convolution by Zernike polynomials [60] as a response function to select point locations. Moreover, we modify the well-known region detectors, such as MSER, to sample points on edge maps, or the edges of the detected regions. Finally, we propose a detector that exploits the ℓ_2-norm of already extracted local descriptors to select the patches. In the following, whenever we mention that local maxima search is performed, a 3×3 neighborhood is used.

A. Relaxing Harris-Laplace detector

Harris-Laplace detector is designed to detect corners. We propose to replace Harris-Laplace cornerness function with Frobenius norm in order to select other points in addition to corners. We also propose to adjust the local extrema selection criterion to sample denser points for classification context.

The first modification is to adjust the standard cornerness function to sample denser points. We estimate the energy of the Harris matrix by its *Frobenius norm*. It is defined as the square root of the sum of the absolute squares of its elements. This offers the ability to produce high response for structures with high energy other than corners.

As our second modification, we propose to modify the local maxima criterion. Regularly, keypoints are selected as local maxima of some interestingness measure. Instead of using the 8-neighborhood of a pixel as local maxima criterion as the standard procedure, we propose to use multiple 2-neighborhoods in different directions, as shown in Figure 4. We select the points that are local maxima in any of those neighborhoods. In this fashion, we retrieve more points along the edges of objects or object parts.

In order to investigate the contribution of different modifications, we end up with four different methods. These are, the regular Harris-Laplace (Harris), Frobenius norm based detector (Frobenius), Harris-Laplace with relaxed local maxima selection (relaxed-Harris) and Frobenius norm based detector with relaxed local maxima selection (relaxed-Frobenius).

In the example of Figure 3, we observe that points are located on edges as well as corners using the Frobenius norm. The relaxed local maxima criterion also provides more points outlining the shapes on the image.

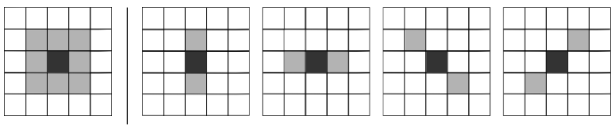
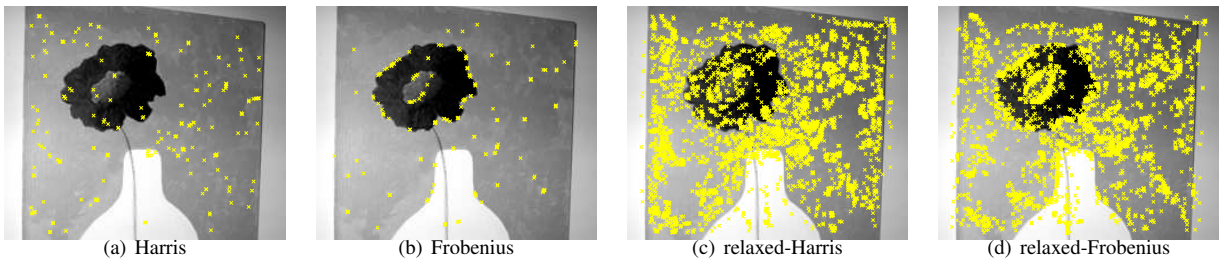


Fig. 4. The search neighborhood typically used to detect local maxima is shown on the left. We relax it by taking the union of extrema obtained from multiple neighborhoods shown on the right.



(a) Harris (b) Frobenius (c) relaxed-Harris (d) relaxed-Frobenius

Fig. 3. The detected keypoints for our Harris Laplace modifications. We visualize the centers of detected points.

B. Zernike

Zernike polynomials were originally introduced by Zernike [60] and are traditionally used in optics and also as shape descriptors [25][48]. They form a set of orthogonal basis functions defined on the unit disc. We focus on the use of pseudo-Zernike polynomials [7] which have proven to be more robust to image noise [50]. They are defined as

$$v_n^l = R_{nl}(r)e^{il\theta}, \tag{3}$$

with

$$R_{nl}(r) = \sum_{s=0}^{n-|l|} (-1)^s \frac{(2n+1-s)!}{s!(n-|l|-s)!(n+|l|+1-s)!} r^{n-s}, \tag{4}$$

where n is the order and l is the repetition. In Figure 5 we present pseudo-Zernike polynomials of order 1 up to order 4. Starting from low frequency patterns they end up being more complex patterns and more localized.

We normalize pseudo-Zernike functions to a 2D rectangular patch (patch width equal to 11 in all our experiments) and convolve the input image using them. Each pseudo-Zernike function is used as a filter and this response constitutes the interestingness measure for detection of point locations. Local maxima and local minima are detected for each filter independently. We now define the maximum number of patches to be extracted per image as N_z. This is used to define the available capacity per filter, in order to retain only the strongest detections. Capacity is uniformly shared among different filters, and also among maxima and minima. We apply the same procedure in $n_\sigma = 5$ scales, similarly to regular sampling. The ratio of capacities of two consecutive scales is 2. That is equal to the ratio of the down-sampled image areas. Filter responses are ranked per filter, and point locations are selected until each filter capacity is filled or until there are no more local extrema left.

We employ N_f filters, which detect complementary structures as being orthogonal and produce high responses for different structures. In contrast to previous feature detectors, we do not focus particularly on detecting corners, blobs or edges and claim that all such structures are useful for image representation. For given capacity, the more filters we use the stronger features we select per filter. In Figure 6, we show the responses produced by various filters. Observe, from the responses and from the filters of Figure 5, that lower order polynomials will detect vertical edges, horizontal edges and

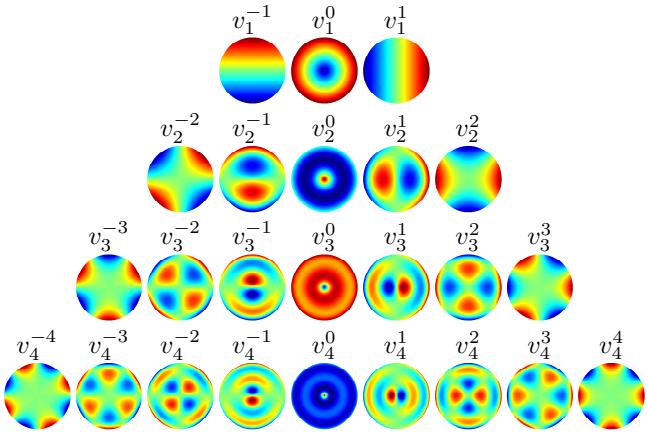


Fig. 5. Visualization of pseudo-Zernike polynomials for $n = 1 \ldots 4$.

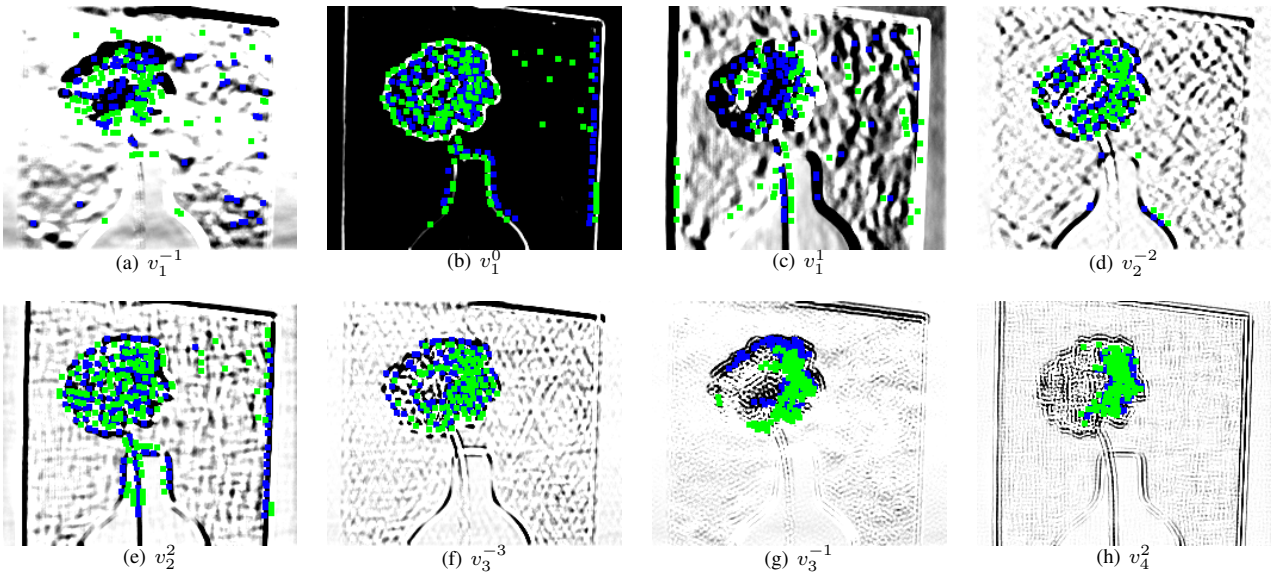
blobs (v_1^{-1},v_1^1 and v_1^0 respectively.), while filters of higher orders will detect more complex image structures.

C. Edge maps

We stress that it is advantageous to describe an image by patches centered on its edges. We do not focus on solving the edge detection problem, where breakthrough research have been done, such as the recent fast edge detection method based on structured forests [13]. We use the existing techniques to sample dense points from images.

We initially consider MSER arbitrarily shaped regions to create an edge map. This 2D map is equal 1 on the region borders and 0 everywhere else. We compute the gradient magnitude on edge pixels and use it as interestingness measure. A local maxima selection is performed to select keypoint locations. Note that only edge pixels are selected in this manner. We center n_σ patches at each detected location, one at each of the multiple scales used. We typically use the same number of scales employed in regular dense sampling, as reported in Section III. We will refer to this detection approach as MSER-edge. Comparing MSER to MSER-edge is a way to compare performance based on region description by its interior or by its boundaries. A similar prior attempt is to describe an MSER region by a set of Local Affine Frames extracted with multiple affine-covariant procedures [39].

We now consider exactly the same procedure, but applied on the edge map derived from the selective search algorithm of Uijlings's *et al.* [54]. Their starting point is Felzenszwalb's segmentation algorithm [16] applied on multiple color spaces.



(a) v_1^{-1} (b) v_1^0 (c) v_1^1 (d) v_2^{-2}

(e) v_2^2 (f) v_3^{-3} (g) v_3^{-1} (h) v_4^2

Fig. 6. The convolution response for some of the pseudo-Zernike polynomials. The response map is visualized only for the first scale. We use green (blue) patches to visualize the local maxima (minima). *Best viewed in color.*

Similar segments are merged in a hierarchical manner, and the resulting segments form candidate object regions. We tune it to produce smaller regions, via parameter k, potentially capturing object parts. A super-pixel [1] like segmentation is produced in this fashion.

Except for centering patches on selective search region (SSR) borders, we also fit an upright ellipse to each one. We refer to the latter as SSR, and to the former as SSR-edge.

Finally, we consider the fast edge detection algorithm of Dollár and Zitnick [13] and apply the same process. The only difference is that the interestingness measure now comes directly from the edge detection process, that is the edge strength. We further eliminate all pixels with strength less than τ, in order to control the number of patches per image. We refer to this approach as *fast-edge* in our experiments.

D. Dense ℓ_2-norm local maxima

Gosselin *et al.* [19] show that the classification performance increases when SIFT vectors with low energy are filtered out. This reasoning is supported visually, as these features correspond to homogeneous regions with very little visual information. However, finding a threshold value is not a feasible operation as it requires cross validation. The optimal threshold value may vary depending on the image and the size of sampled patches. The effect of this filtering strategy on regular dense is shown in Figure 7, where the descriptors whose squared ℓ_2-norm is lower than the threshold τ are removed.

It appears that SIFT vectors and in particular their ℓ_2-norm provide an interestingness information subsequent at feature detection. In contrast to traditional detectors that localize interest points by low-level statistics, such as gradient changes, we propose to use the ℓ_2-norm of each SIFT as an interestingness measure. Initially, we compute SIFT on a very dense uniform grid of step $\delta_{xy} = 1$ and at 5 different scales. The ℓ_2-norm of SIFT descriptors constitutes a response map. That is the interestingness value per pixel. Following the standard detector procedure, we select the local maxima of this response map. Similar to interest point detectors, we introduce a threshold parameter τ, to only retain the stronger points.

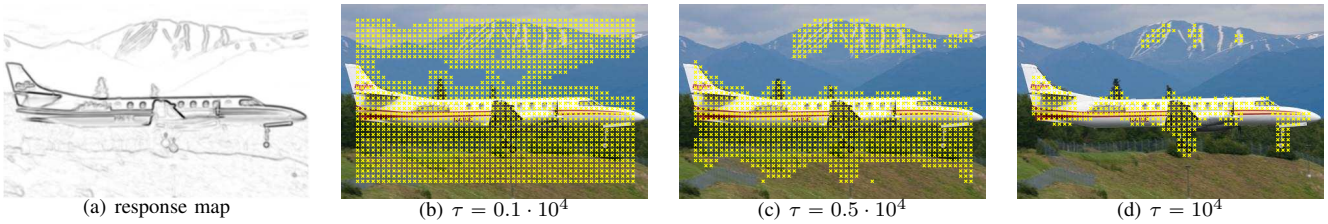
As a consequence, patches from homogeneous regions are discarded and the ones with highest energy of SIFT descriptors and more structured regions are retained. As shown in Figure 2 the retained points are mostly localized along edges and corners. It is the only proposed method that is based on the local descriptor to compute the interestingness measure.

Detector	Section
Dense	Section III-B
Dense Interest Points (Dense-IP) [53]	Section III-B
Dense ℓ_2-norm	Section IV-D
Harris-Laplace [34], +relaxed	Sections III-A, IV-A
Frobenius, +relaxed	Section IV-A
Hessian Affine (HesAff) [35]	Section III-A
Difference of Gaussians (DoG) [29]	Section III-A
Zernike	Section IV-B
MSER [31], +edge	Sections III-A,IV-C
Selective Search [54] (SSR), +edge	Section IV-C
Edge Detector (Fast-edge) [13]	Section IV-C

TABLE I
LIST OF DETECTORS USED IN EXPERIMENTS.

V. EXPERIMENTS

We compare different detection strategies in two different applications, image retrieval and fine-grained image classification. In our experiments, we analyze aspects of feature detection and extraction separately and report our findings. List of detectors used in our experiments are shown in Table I. Note that all the detectors we use are rotation variant (up-right).



(a) response map (b) $\tau = 0.1 \cdot 10^4$ (c) $\tau = 0.5 \cdot 10^4$ (d) $\tau = 10^4$

Fig. 7. The response map of ℓ_2-norm of SIFT descriptors is shown in (a). Dark pixels correspond to higher responses. Filtering with different threshold values are shown in (b)-(d). We use the patch size of 41×41 pixels.

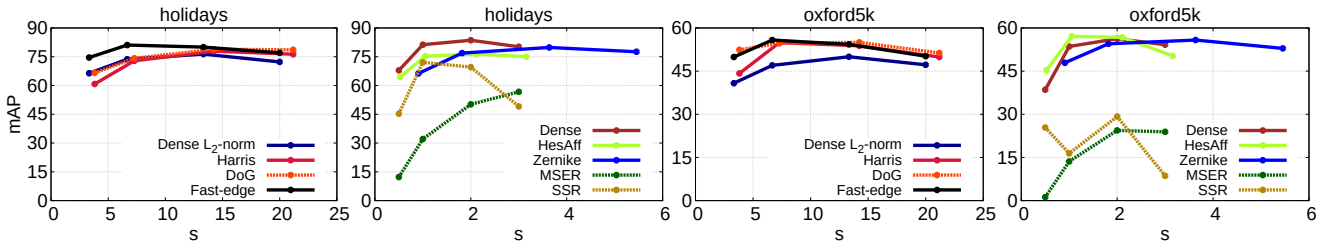


Fig. 8. Impact of scaling factor on retrieval performance.

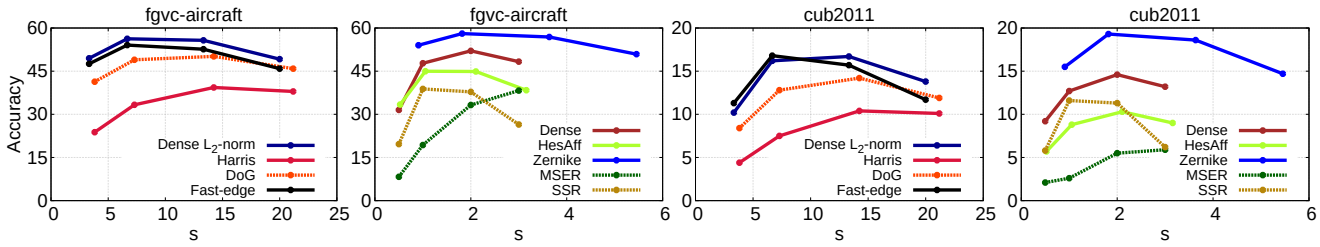


Fig. 9. Impact of scaling factor on fine-grained classification performance.

A. *Experimental Setup*

Image Retrieval datasets. INRIA Holidays dataset [23] consists of 1,491 personal holiday photographs. There are 500 image groups, with each group having a single query image. We use the up-right version of this dataset, where each image is rotated according to the natural orientation of objects depicted. Flickr60k dataset [23] is used as a training set for this dataset.

Oxford5K Buildings dataset [45] has 5,063 images of Oxford landmarks collected from Flickr. There are 55 query images. Training is performed using Paris 6k dataset [46].

Fine-grained classification datasets. We use the following evaluation benchmarks:

- FGVC-Aircraft dataset [30] has 10,200 aircraft images. There are three different classification tasks, classification by 102 different variants, 70 different families, and 41 different manufacturers. We perform experiments on classification of variants, and use the provided train, validation and test sets.
- Caltech-UCSD Birds 2011 dataset [55] consists of images of bird species. The dataset has 200 categories, and 11,788 images. We randomly select 30% of the training images as validation set [55].
- Oxford-IIIT Pet dataset [40] has 37 categories and 7,349 images of cats and dogs. We randomly select 50% of the training images as validation set [40].
- The 102 Category Flower dataset [37] consists of 102 flower categories, with each category having between 40 and 258 images. We use the provided train, validation and test sets.

We did not make any use of the provided objects regions or segments in any of the aforementioned datasets. Although using segmentation information is shown to improve results [57], we do not consider such an approach. Evaluating the performance on full images offers a direct and fair way to compare different detectors.

Image retrieval pipeline. All images are initially down-sampled to 150k pixels. We extract 128-dimensional SIFT descriptors from the detected set of keypoints. We apply RootSIFT [3], [21], center, rotate along the PCA-axis and ℓ_2 normalize the local descriptors. Codebook of 256 visual words is trained with k-means on an independent dataset. We encode the set of local descriptors with the VLAD [24] representation. Finally, power-law [43] and ℓ_2 normalization is applied to the VLAD vectors. During evaluation, nearest neighbors are found for each query vector, and the performance is measured with mean Average Precision (mAP).

Fine-grained classification pipeline. The down-sampling of images and post-processing of local descriptors is the same as in the retrieval task, except to the fact that we reduce to 80 dimensions using PCA, instead of only rotating and keeping all dimensions. A GMM of 256 components is trained, and the

Detector	conversion	p=11	p=21	p=41	p=61
Dense Dense-IP [53]	$s = (p-1)/20$	0.5	1	2	3
HesAff [35] MSER [31] SSR [54]	$s = (p-1)/20$	0.5	1	2	3
Harris-Laplace [34] DoG [29]	$s = p/2.88$	3.82	7.29	14.24	21.18
Dense ℓ_2-norm MSER-edge SSR-edge Fast-edge [13]	$s = (p-1)/3$	3.33	6.67	13.33	20
Zernike	$s = (p-1)/11$	1	1.91	3.73	5.55

TABLE II
RELATION OF SCALING FACTOR TO PATCH SIZE. WE LINEARLY RELATE SCALING FACTOR TO PATCHSIZE.

Detector	param	explanation	values
Dense Dense-IP [53]	δ_{xy}	step size	4,8,12,16
Dense ℓ_2-norm Harris-Laplace [34] HesAff [35] DoG [29]	τ	threshold	0,50,100,200 0,50,100,200,300,400 0,50,100,200,300,400 0.1,0.2,0.3
Zernike	N	number of features	1k,5k,10k,16k,20k
MSER [31] MSER-edge	Δ	stable region parameter	3,5,10,15
SSR [54] SSR-edge	k	initial segment size	25,50,100
Fast-edge [13]	τ	edge strength threshold	0,0.1,0.2,0.3

TABLE III
PARAMETERS CONTROLLING NUMBER OF POINTS FOR EACH DETECTOR AND THEIR CORRESPONDING VALUES.

Fisher vector [42] is adopted to represent an image. Power-law [43] normalization is applied to the Fisher vectors, with its optimal parameter found through cross-validation. A linear classifier is trained using stochastic gradient descent [2], and the accuracy is reported.

B. Impact of the scaling factor

Traditionally, a detected region of interest is mapped to a rectangular patch of width equal to p pixels and a local descriptor is extracted from it. It is beneficial to enlarge this measurement region by a scaling factor [49], denoted by s. The ratio between the descriptor measurement region (used to extract local descriptors) and the detection measurement region (used to compute the interestingness measure to detect keypoints) is exactly the scaling factor s. When $s = 1$, these two are identical.

We investigate the impact of the scaling factor for a variety of detection processes. We linearly relate the scaling factor to the patch size, such that larger regions will be normalized to patches of larger resolution. This exact relation is presented in Table II for all the examined detectors.

Affine regions are isotropically enlarged, and when no enlargement is involved, the patch size is equal to 21. For Zernike, no scaling factor corresponds to descriptor measurement region equal to 11, since this is the filter size we use (see Section IV). For regular dense sampling, where there is no detector measurement region, we have arbitrarily defined the scaling factor to be equal to 1 when the descriptor measurement region size is 21 pixels.

We perform experiments to investigate the impact of the scaling factor and present results in Figures 8 and 9. It is observed that such kind of region enlargement is beneficial for all detectors. Performance increases by increasing the descriptor measurement regions, while for large increase it saturates or even drops. Our conclusions agree with the conclusions of Simonyan *et al*. [49], but we evaluate the scaling factor for a larger number of detectors and both classification and retrieval tasks. We adopt the scaling factor that corresponds to a patch size of 41 for all detectors, as it seems to be a good choice overall. The same stands for detectors not included in the corresponding figures. Note that we do not compare different detectors with each other in this experiment. We investigate the impact of scaling factor for each detector individually. Each detector produces different number of features per image in this experiment. We consider this a crucial parameter for performance and investigate it in Section V-D.

C. Impact of focusing on edges

In Section IV we proposed detectors that have different localization properties than the existing ones. In particular, we try to focus the local representation on image edges.

In Figure 10, we present the performance measured for Harris-Laplace detector and the relaxed modifications that we proposed. Relaxed-Harris produces more keypoints and consistently improves performance. The relaxation based on Frobenius norm also seems to improve. An exception is the Oxford5K Buildings dataset, where due to particular object matching, original Harris keypoints are localized to have higher repeatability and perform better.

We argue that the typical choice of fitting an ellipse and trying to describe a region by its interior is not necessarily the optimal option for the tasks we consider. Figure 11 shows the results of comparing such a traditional approach to our proposal of extracting local descriptors from patches centered on the region borders. We perform this comparison for MSER regions and selective search regions (SSR). Interestingly, the standard choice does not appear to be the optimal. A larger number of features is obtained when focused on edges, which generally results in better performance. Finally, the region detector based on the fast edge detection algorithm [13] (fast-edge) performs rather well, especially for intermediate number of descriptors per image.

Additionally, we conduct localization experiments for the proposed Zernike detector by directly controlling the number of keypoints via parameter N_z. We also capture more complementary information by increasing the number of filters N_f. We evaluate performance using all polynomials up to order 2, 3 and 4. These correspond to 8, 15 and 24 filters respectively. We present the results in Figure 12. Less filters seem to perform better on retrieval datasets. On the contrary, more filters are needed for fine-grained classification, possibly to capture larger intra class variations that exist. We set N_f equal to 8 for retrieval and 15 for fine-grained classification.

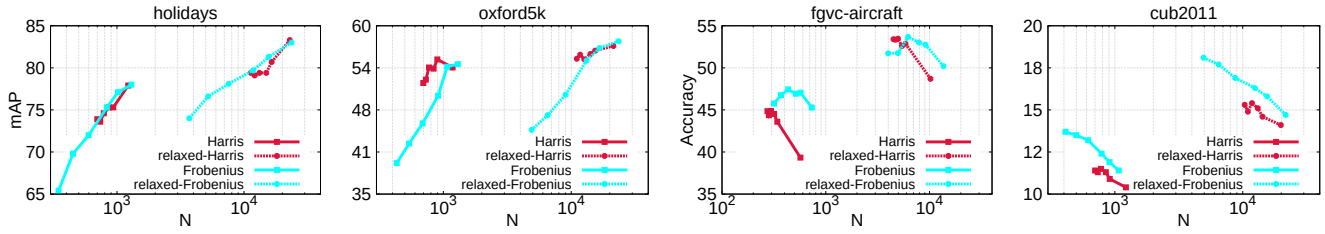


Fig. 10. Performance comparison between Harris-Laplace and its relaxed modifications that we propose. We show performance versus average number of descriptors per image N.

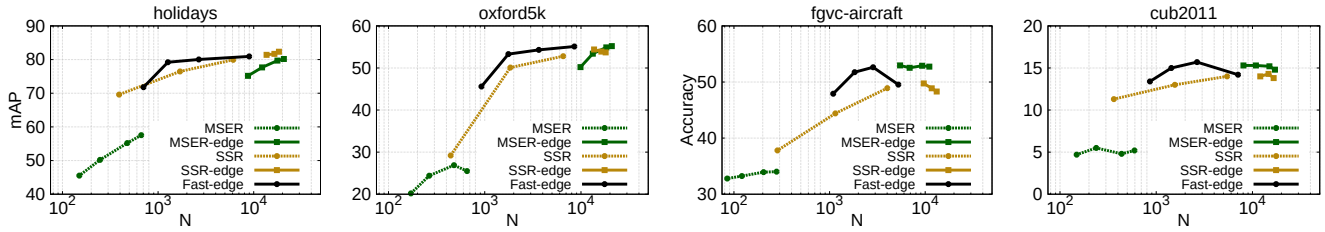


Fig. 11. Performance comparison between approaches that describe a detected region by fitting a single ellipse and approaches that sample patches along the region edges. Fast-edge is another detector that samples patches on edges. We show performance versus average number of descriptors per image N.

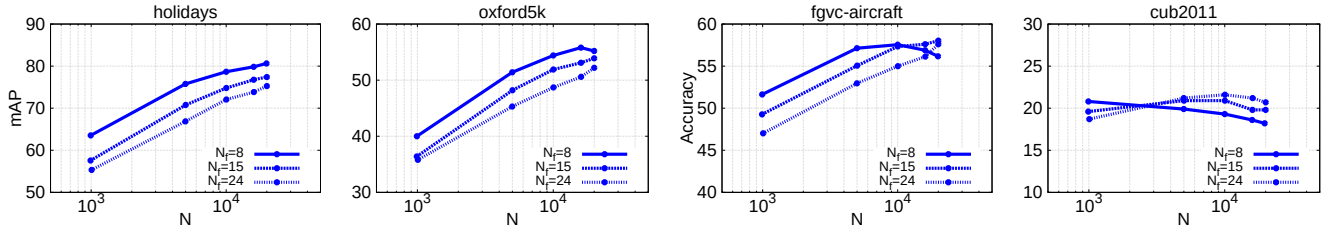


Fig. 12. Performance versus average number of descriptors for Zernike detector with different number of filters.

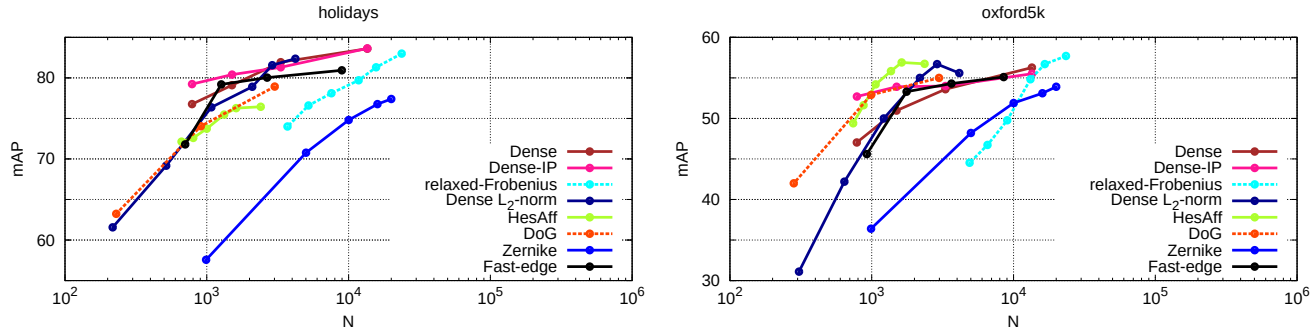


Fig. 13. Performance comparison on image retrieval versus number of descriptors per image N. The parameter values shown in Table III are used.

D. Performance comparison versus number of descriptors

A single parameter for each detector controls the number of keypoints per image. These are shown in Table III. We evaluate performance for multiple values. This allows a fair comparison of all detectors with respect to the number of descriptors used to describe an image. The descriptor set cardinality is directly related to the complexity of encoding stage.

Results of performance comparison between different detectors versus the average number of features per image N are shown in Figures 13 and 14, for retrieval and classification respectively. Only the best performing variants of the previous experiments are included in this comparison.

On retrieval it appears that the more descriptors the better, while for fine-grained classification this is not always the case. However, such a behavior seems to be consistent among all detectors for a particular dataset. Thus, it seems to be related with the nature of images and objects depicted in them.

The relaxed Frobenius detector achieves the best performance on Oxford5k, however Hessian-Affine also performs very well with less number of features, due to its better

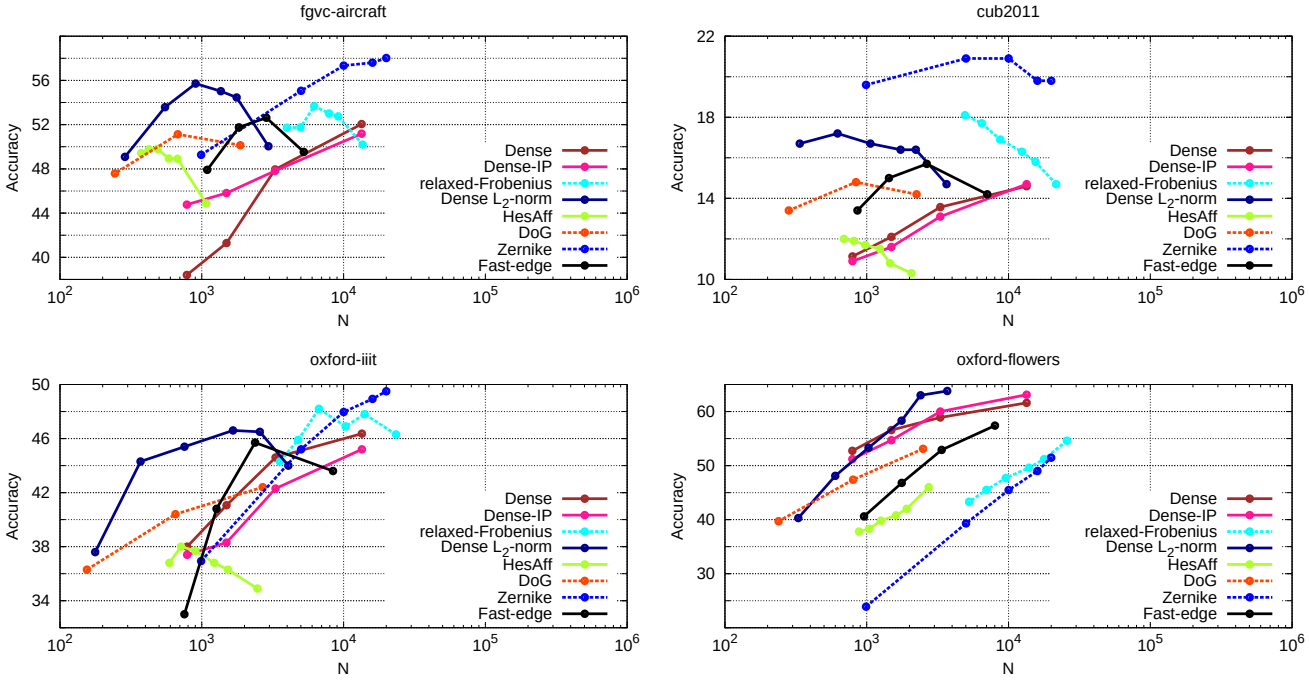


Fig. 14. Performance comparison on fine-grained classification versus number of descriptors per image N. The parameter values shown in Table III are used.

Fig. 15. Detection example showing the failure of Zernike detector on images of the Oxford-Flowers dataset.

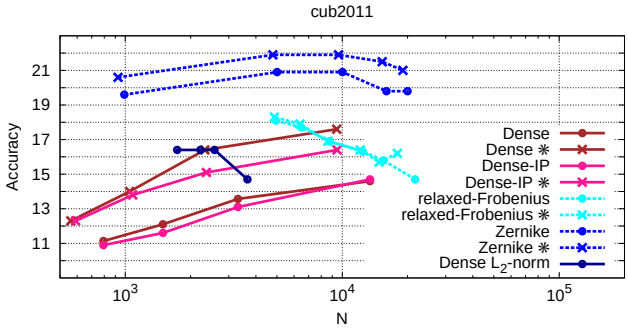


Fig. 16. Impact of ℓ_2-norm based filtering when applied on top of a variety of detectors. We show performance versus average number of descriptors per image N. $*$ corresponds to detection with ℓ_2-norm filtering.

localization properties. However, we observe that the standard dense and relaxed-Frobenius detectors give the best performance using larger number of features on Holidays. This fact is related to the nature of the dataset; it contains more scenes and not particular objects. On fine-grained classification, the Zernike detector outperforms all other detectors by far, with an exception on Oxford-Flowers dataset. The latter is attributed to many points localized on background texture, as shown in the example of Figure 15. Interestingly, over both tasks our dense ℓ_2-norm local maxima detector achieves best performance for an intermediate amount of points.

Finally, note that detectors which extract descriptors from patches of 5 constant scales perform better than the detectors that detect local maxima in the scale space. It appears that the former attains enough tolerance to scale changes, although it is not scale invariant.

E. *Effect of ℓ_2-norm based filtering*

Gosselin *et al.* [19] show that the accuracy on fine-grained classification based on regular dense sampling significantly increases when the SIFT vectors with low ℓ_2-norm are filtered out. We apply the same strategy and show results on Figure 16. We compute squared ℓ_2-norm to avoid the calculation of square root, and choose $T = 5000$. As expected, this strategy which aims at removing descriptors extracted from too uniform regions improves performance for regular dense sampling, but the improvement is not significant for detectors that are better localized.

F. *State of the art in fine-grained classification*

We compare accuracy of Zernike detector, which is best performing among our contributions, with the state of the

Method	$\#C$	FGVC-Air.	CUB2011	Ox.-IIIT	Ox.-Flowers
Zernike	256	56.7	19.3	49.6	51.1
Zernike	4096	66.2	29.6	57.1	51.2
Zernike+SCC	4096	69.9	31.9	59.5	56.1
GMP [36]	256	-	17.0	49.2	73.3
GMP+XColor [36]	256	-	33.3	56.8	84.6

TABLE IV
PERFORMANCE COMPARISON WITH THE STATE OF ART METHODS ON FINE-GRAINED IMAGE CLASSIFICATION. SCC DENOTES SPATIAL COORDINATE CODING AND $\#C$ IS THE CODEBOOK SIZE. ALL METHODS USE SIFT DESCRIPTORS, EXCEPT FOR THE LAST ONE WHICH ADDITIONALLY MAKES USE OF COLOR DESCRIPTORS.

art results on fine-grained classification. Fisher vectors are employed once more to encode an image based on Zernike patches. This time, we also apply horizontal mirroring to images, in order to obtain another Fisher vector representation per image. As recently shown, encoding spatial information can boost performance [19]. We adopt the choice of spatial coordinate coding (SCC) [32][26] that allows the use of much larger codebooks, without increase of dimensionality, in contrast to spatial pyramid matching. We choose SCC instead of spatial pyramid matching following the work of Gosselin [19], which shows that SCC performs better than spatial pyramid matching for fine-grained image classification. It allows the use of much larger codebooks. In particular, we append regularized 2D coordinates of patches to RootSIFT descriptors before quantizing them. Regularization weights are learned via cross validation. We have set $N_{\mathrm{z}} = 10\mathrm{k}$ for the Zernike detector.

Several works focus on a single domain of fine-grained classification, *e.g.* birds, and optimize their methods only for that particular domain. Since this is not the case with our study, we compare with methods that are similar. We present our results in Table IV, and compare them with the work of Murray and Perronnin [36]. Using the same codebook, with no additional features, we outperform their results in 2 out of 3 datasets. They also employ color descriptors [11] which further improves the accuracy. This can complement also for Zernike patches. We also increase the codebook size and combine with SCC that drastically boosts performance.

To our knowledge, there are not any published results for the FGVC-Aircraft dataset yet, except for the initial paper that achieved accuracy equal to 48.69% [30]. For the other datasets, it is possible to improve the results using the provided annotations of bounding boxes or object parts. We do not consider such an option. However, the best reported result on CUB2011 is 85.40% [8], which trains convolutional neural networks (CNN) using such annotations. On the other hand, we are superior to methods which use segmentation information on Oxford-IIIT dataset, such as Wang *et al.* [57] that report 59.29%. In the case of Oxford-Flowers dataset, the highest reported score is 86.6% [47], where the authors also use CNNs without the provided segmentation information.

VI. CONCLUSIONS

We investigated dense keypoint detection solutions that lie in between sparse interest points and dense sampling on a uniform grid. We propose to modify existing interest point detectors by relaxing the cornerness criterion and the local maxima selection. We introduce a new detection method using Zernike filters, which provides dense, yet localized image patches. We also show that sampling patches on the borders of a region of interest performs better than the standard choice of fitting an ellipse and describing it by a single descriptor. Finally, we propose to detect dense patches by using the ℓ_2-norm of the descriptor instead of low-level pixel information. To our knowledge, this is the first detection strategy which focuses on descriptors, and the results seem very promising.

Interestingly, solutions employing patches of multiple fixed scales perform better than patches detected as local maxima in the scale space. Albeit not scale invariant, apparently this option provides enough scale tolerance for the tasks of image retrieval and fine-grained classification.

Compared with the existing studies, Zernike patches encoded with a standard technique, such as Fisher vectors, appear to outperform state of the art approaches for some of the fine-grained classification datasets. An exception is the Oxford-Flowers dataset that Zernike seem to perform poorly.

ACKNOWLEDGMENT

This work was supported by ERC grant VIAMASS no. 336054 and ANR project FIRE-ID.

REFERENCES

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *Trans. PAMI*, vol. 34, no. 11, pp. 2274–2282, 2012.
[2] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid, "Good practice in large-scale learning for image classification," *Trans. PAMI*, Jun. 2013.
[3] R. Arandjelovic and A. Zisserman, "Three things everyone should know to improve object retrieval," in *CVPR*, Jun. 2012.
[4] Y. Avrithis and K. Rapantzikos, "The medial feature detector: Stable regions from image boundaries," in *ICCV*, Nov. 2011.
[5] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky, "Neural codes for image retrieval," in *ECCV*, 2014.
[6] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "SURF: Speeded up robust features," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, May 2008.
[7] A. Bhatia and E. Wolf, "On the circle polynomials of Zernike and related orthogonal sets," in *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 50, no. 01. Cambridge Univ Press, 1954, pp. 40–48.
[8] S. Branson, G. V. Horn, S. Belongie, and P. Perona, "Bird species categorization using pose normalized deep convolutional nets," Arxiv, Tech. Rep., 2014.
[9] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in *ECCV*, Oct. 2010.
[10] V. Chandrasekhar, G. Takacs, D. Chen, S. Tsai, R. Grzeszczuk, and B. Girod, "CHoG: compressed histogram of gradients," in *CVPR*, Jun. 2009.
[11] S. Clinchant, G. Csurka, F. Perronnin, and J.-M. Renders, "XRCEs participation to imageval," in *ImageEval workshop at CVIR*, 2007.
[12] J. Delhumeau, P.-H. Gosselin, H. Jégou, and P. Pérez, "Revisiting the VLAD image representation," in *ACM Multimedia*, Oct. 2013.
[13] P. Dollár and C. L. Zitnick, "Structured forests for fast edge detection," in *ICCV*. IEEE, Dec. 2013, pp. 1841–1848.
[14] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition," in *ICML*, 2014.
[15] L. Fei-Fei and P. Perona, "A bayesian hierarchical model for learning natural scene categories," in *CVPR*, 2005.
[16] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *IJCV*, vol. 59, no. 2, pp. 167–181, 2004.
[17] M. A. Fischler and R. C. Bolles, "Random sample consensus," *Communications of ACM*, vol. 6, no. 24, pp. 381–395, 1981.

[18] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *CVPR*, 2014.

[19] P. Gosselin, N. Murray, H. Jégou, and F. Perronnin, "Revisiting the Fisher vector for fine-grained classification," *Pattern Recognition Letters*, 2014.

[20] C. Harris and M. Stephens, "A combined corner and edge detector." in *Alvey vision conference*, 1988.

[21] M. Jain, R. Benmokhtar, P. Gros, and H. Jégou, "Hamming embedding similarity-based image classification," in *ICMR*, Jun. 2012.

[22] M. Jain, H. Jégou, and P. Bouthemy, "Better motion for better action recognition," in *CVPR*, Jun. 2013.

[23] H. Jégou, M. Douze, and C. Schmid, "Hamming embedding and weak geometric consistency for large scale image search," in *ECCV*, Oct. 2008.

[24] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *CVPR*, Jun. 2010.

[25] A. Khotanzad and Y. H. Hong, "Invariant image recognition by Zernike moments," *Trans. PAMI*, vol. 12, no. 5, pp. 489–497, 1990.

[26] P. Koniusz, F. Yan, and K. Mikolajczyk, "Comparison of mid-level feature coding approaches and pooling strategies in visual concept detection," *Computer Vision and Image Understanding*, vol. 117, no. 5, pp. 479–492, 2013.

[27] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.

[28] T. Lindeberg, "Feature detection with automatic scale selection," *IJCV*, vol. 30, no. 2, pp. 77–116, 1998.

[29] D. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[30] S. Maji, J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi, "Fine-grained visual classification of aircraft," Arxiv, Tech. Rep., 2013.

[31] J. Matas, O. Chum, U. Martin, and T. Pajdla, "Robust wide baseline stereo from maximally stable extremal regions," in *BMVC*, Sep. 2002, pp. 384–393.

[32] S. McCann and D. G. Lowe, "Spatially local coding for object recognition," in *Computer Vision–ACCV 2012*. Springer, 2013, pp. 204–217.

[33] K. Mikolajczyk, A. Zisserman, and C. Schmid, "Shape recognition with edge-based features," in *BMVC*, 2003.

[34] K. Mikolajczyk and C. Schmid, "Scale and affine invariant interest point detectors," *IJCV*, vol. 60, no. 1, pp. 63–86, Oct. 2004.

[35] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. V. Gool, "A comparison of affine region detectors," *IJCV*, vol. 65, no. 1/2, pp. 43–72, Nov. 2005.

[36] N. Murray and F. Perronnin, "Generalized max pooling," in *CVPR*, 2014.

[37] M.-E. Nilsback and A. Zisserman, "Automated flower classification over a large number of classes," in *Indian Conference on Computer Vision, Graphics and Image Processing*, Dec. 2008.

[38] E. Nowak, F. Jurie, and B. Triggs, "Sampling strategies for bag-of-features image classification," in *ECCV*, 2006.

[39] Š. Obdržálek and J. Matas, "Object recognition using local affine frames on maximally stable extremal regions," in *Toward Category-Level Object Recognition*. Springer, 2006, pp. 83–104.

[40] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. V. Jawahar, "Cats and dogs," in *CVPR*, 2012.

[41] M. Perdoch, O. Chum, and J. Matas, "Efficient representation of local geometry for large scale object retrieval," in *CVPR*, Jun. 2009.

[42] F. Perronnin and C. R. Dance, "Fisher kernels on visual vocabularies for image categorization," in *CVPR*, Jun. 2007.

[43] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the Fisher kernel for large-scale image classification," in *ECCV*, Sep. 2010.

[44] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," in *CVPR*, Jun. 2007.

[45] ——, "Oxford5k dataset," http://www.robots.ox.ac.uk/~vgg/data, 2007.

[46] ——, "Lost in quantization: Improving particular object retrieval in large scale image databases," in *CVPR*, 2008.

[47] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: an astounding baseline for recognition," Arxiv, Tech. Rep., 2014.

[48] J. Revaud, G. Lavoué, and A. Baskurt, "Improving Zernike moments comparison for optimal similarity and rotation angle retrieval," *Trans. PAMI*, vol. 31, no. 4, pp. 627–636, 2009.

[49] K. Simonyan, A. Vedaldi, and A. Zisserman, "Learning local feature descriptors using convex optimisation," *Trans. PAMI*, 2014.

[50] C.-H. Teh and R. T. Chin, "On image analysis by the methods of moments," *Trans. PAMI*, vol. 10, no. 4, pp. 496–513, 1988.

[51] G. Tolias, Y. Avrithis, and H. Jégou, "To aggregate or not to aggregate: Selective match kernels for image search," in *ICCV*, Dec. 2013.

[52] G. Tolias and H. Jégou, "Visual query expansion with or without geometry: refining local descriptors by feature aggregation," *Pattern Recognition*, Apr. 2014.

[53] T. Tuytelaars, "Dense interest points," in *CVPR*, 2010.

[54] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *IJCV*, vol. 104, no. 2, pp. 154–171, 2013.

[55] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The Caltech-UCSD Birds-200-2011 Dataset," California Institute of Technology, Tech. Rep. CNS-TR-2011-001, 2011.

[56] H. Wang, A. Kläser, C. Schmid, and L. Cheng-Lin, "Action recognition by dense trajectories," in *CVPR*, Jun. 2011.

[57] Z. Wang, J. Feng, and S. Yan, "Collaborative linear coding for robust image classification," *IJCV*, pp. 1–12, 2014.

[58] S. Winder and M. Brown, "Learning local image descriptors," in *CVPR*, Jun. 2007.

[59] S. Winder, G. Hua, and M. Brown, "Picking the best Daisy," in *CVPR*, Jun. 2009.

[60] v. F. Zernike, "Beugungstheorie des schneidenver-fahrens und seiner verbesserten form, der phasenkontrastmethode," *Physica*, vol. 1, no. 7, pp. 689–704, 1934.

[61] W. Zhao, H. Jégou, and G. Gravier, "Oriented pooling for dense and non-dense rotation-invariant features," in *BMVC*, Sep. 2013.

Ahmet Iscen received his B.S. degree from The State University of New York at Binghamton, and M.S from Bilkent University. He is a PhD student at Inria Rennes and University of Rennes I, working on large-scale image retrieval.

Giorgos Tolias holds a Ph.D. in Electrical and Computer Engineering from the National Technical University of Athens. He is a post-doctoral researcher at Inria Rennes, working mainly in the area of Computer Vision with focus on image search.

Philippe-Henri Gosselin received the PhD degree in image and signal processing in 2005 (Cergy, France). After 2 years of post-doctoral positions at the LIP6 Lab. (Paris, France) and at the ETIS Lab. (Cergy, France), he joined the MIDI Team in the ETIS Lab as an assistant professor, and then promoted to full professor in 2012. His research focuses on machine learning for online multimedia retrieval. He developed several statistical tools for dealing with the special characteristics of content-based multimedia retrieval. This includes studies on kernel functions on histograms, bags and graphs of features, but also weakly supervised semantic learning methods. He is involved in several international research projects, with applications to image, video and 3D objects databases.

Hervé Jégou holds a Ph.D in Computer Science from the University of Rennes, defended in 2005 on joint source channel coding. He joined INRIA as a permanent researcher in 2006, in the LEAR team, and moved to INRIA Rennes in 2009. His research interests include large scale indexing, essentially for large image and video collections.

VII Learning and aggregating deep local descriptors for instance-level recognition

Title: Learning and aggregating deep local descriptors for instance-level recognition
Authors: G. Tolias, T. Jenicek, O. Chum
Published at: ECCV 2014

Learning and aggregating deep local descriptors for instance-level recognition

Giorgos Tolias, Tomas Jenicek, and Ondřej Chum

Visual Recognition Group, Faculty of Electrical Engineering
Czech Technical University in Prague

Abstract. We propose an efficient method to learn deep local descriptors for instance-level recognition. The training only requires examples of positive and negative image pairs and is performed as metric learning of sum-pooled global image descriptors. At inference, the local descriptors are provided by the activations of internal components of the network. We demonstrate why such an approach learns local descriptors that work well for image similarity estimation with classical efficient match kernel methods. The experimental validation studies the trade-off between performance and memory requirements of the state-of-the-art image search approach based on match kernels. Compared to existing local descriptors, the proposed ones perform better in two instance-level recognition tasks and keep memory requirements lower. We experimentally show that global descriptors are not effective enough at large scale and that local descriptors are essential. We achieve state-of-the-art performance, in some cases even with a backbone network as small as ResNet18.

Keywords: deep local descriptors, deep local features, efficient match kernel, ASMK, image retrieval, instance-level recognition

1 Introduction

Instance-level recognition tasks are dealing with a very large number of classes and relatively small intra-class variability. Typically, even instance-level classification tasks are cast as instance-level search in combination with nearest neighbor classifiers. The first instance-level search approach to achieve good performance, *i.e.* Video Google [42], is based on local features and the Bag-of-Words (BoW) representation. Representing images as collections of vector-quantized descriptors of local features allows for efficient spatial verification [32], which turns out to be a key ingredient in search for small objects. Follow-up approaches improve the BoW paradigm either with finer quantization and better matching schemes [20,44,2] or with compact global descriptors generated through aggregation [22,1]. Good performance is achieved even without spatial verification.

The advent of deep networks made it easy to generate and train global image descriptors. A variety of approaches exist [14,35,3,28,50,16] that differ in the training data, in the loss function, or in the global pooling operation. However, the performance of global descriptors deteriorates for very large collections of

Fig. 1. Learned local features and descriptors matched with ASMK. Features assigned to the same visual word (65k words codebook) are shown in the same color; only top 20 common visual words (out of 94) are included. Accurate localization is not required since we do not use spatial verification to perform instance-level recognition.

images. Noh *et al.* [29] are the first to exploit the flexibility of global descriptor training in order to obtain local features and descriptors, called DELF, for the task of instance-level recognition. DELF descriptors are later shown [34] to achieve top performance when combined with the state-of-the-art image search approach, *i.e.* the Aggregated Selective Match Kernel (ASMK) [44]. Compared to compact global descriptors, this comes with higher memory requirements and search time cost. In contrast to other learned local feature detectors [13] that use keypoint-level supervision and non-maxima suppression, DELF features are not precisely localized and suffer from redundancy since deep network activations are typically spatially correlated.

In this work[1], we propose a local feature detector and descriptor based on a deep network. It is trained through metric learning of a global descriptor with image-level annotation. We design the architecture and the loss function so that the local features and their descriptors are suitable for matching with ASMK, see Figure 1. ASMK is known to deliver good performance even without spatial verification, *i.e.* precise feature localization is not crucial, and it deals well with repeated or bursty features. Therefore, the common drawbacks of existing deep local features for instance-level recognition are overcome. Unlike classical local features that attempt to offer precise localization to extract reliable descriptors, multiple nearby locations can give rise to a similar descriptor in our training; multiple similar responses are not suppressed, but averaged.

The main contribution of this work is the proposed combination of deep feature detector and descriptor with ASMK matching, which outperforms existing global and local descriptors on two instance-level recognition tasks, *i.e.* classification and search, in the domain of landmarks. Our ablation study shows that the proposed components reduce the memory required by ASMK. The learned local descriptors outperform by far deep global descriptors as well as other deep local descriptors combined with ASMK. Finally, we provide insight into why the image-level optimization is relevant for local-descriptors and ASMK matching.

[1] `https://github.com/gtolias/how`

2 Related work

We review the related work in learning global or local descriptors for instance-level matching task and local descriptors for registration tasks.

Global descriptors. A common approach to obtain global image descriptors with deep fully-convolutional neural networks is to perform global pooling on 3D feature maps. This approach is applied to activations generated by pre-trained networks [4,45,23] or end-to-end learned networks [35,14,15]. One of the first examples is SPoC descriptor by Babenko and Lempitsky [4] that is generated by simple global sum-pooling. Weighted sum-pooling is performed in CroW by Kalantidis *et al.* [23], where the weights are given by the magnitude of the activation vectors at each spatial location of the feature map. Such a 2D map of weights, seen as an attention map, is related to our approach as discussed in Section 4.2. Inspired by classical embeddings, Arandjelović *et al.* [3] extend the VLAD [22] descriptor to NetVLAD. Its contextually re-weighted counterpart, proposed by Kim and Frahm [24], introduces a learned attention map which is generated by a small network.

Local features and descriptors for instance-level recognition. Numerous classical approaches that are based on hand-crafted local features [27,25] and descriptors [26,8] exist in the literature of instance-level search [42,32,44,51,30]. Inspired by classical feature detection, Simeoni *et al.* [41] perform MSER [25] detection on activation maps. The features detected at one feature channel are used as tentative correspondences, hence no descriptors are required; the approach is applicable to any network and does not require learning. Learning of attentive deep local features (DELF) is introduced in the work of Noh *et al.* [29]. A global descriptor is derived from a network that learns to attend on feature map positions. The global descriptor is optimized with category-level labels and classification loss. At test time, locations with the strongest attention scores are selected while the descriptors are the activation vectors at the selected locations. This approach is highly relevant to ours. We therefore provide a number of different ablation experiments to reveal the key differences. A recent variant shows that it is possible to jointly learn DELF-like descriptors and global descriptors with a single model [11]. A similar achievement appears in the work of Yang *et al.* [49] with a scope that goes beyond instance-level recognition and covers image registration too.

Local features and descriptors for registration. A richer line of work exists in learning local feature detection and description for image registration where denser point correspondences are required. As in the previous tasks, some methods do not require any learning and are applicable on any pre-trained network. This is the case of the work Benbihi *et al.* [9] where activation magnitudes are back-propagated to the input image and local-maxima are detected. Learning is performed with or without labeling at the local level in a number of different approaches [38,13,12,7,10,47]. A large number of features is typically required for good performance. This line of research differentiates from our work; our focus is on large-scale instance-level recognition where memory requirements matter.

3 Background

In this section, the binarized versions of Selective Match Kernel (SMK) and its extension, the Aggregated Selective Match Kernel (ASMK) [44][2], are reviewed as the necessary background. This paper exploits the ASMK indexing and retrieval.

In SMK, an image is represented by a set $\mathcal{X} = \{\mathbf{x} \in \mathbb{R}^d\}$ of $n = |\mathcal{X}|$ d-dimensional local descriptors. The descriptors are quantized by k-means quantizer $q : \mathbb{R}^d \to \mathcal{C} \subset \mathbb{R}^d$, where $\mathcal{C} = \{\mathbf{c}_1, \dots, \mathbf{c}_k\}$ is a codebook comprising $|\mathcal{C}|$ vectors (visual words). Descriptor $\mathbf{x}$ is assigned to its nearest visual word $q(\mathbf{x})$. We denote by $\mathcal{X}_c = \{x \in \mathcal{X} : q(x) = \mathbf{c}\}$ the subset of descriptors in $\mathcal{X}$ that are assigned to visual word $\mathbf{c}$, and by $\mathcal{C}_\mathcal{X}$ the set of all visual words that appear in $\mathcal{X}$. Descriptor $\mathbf{x}$ is mapped to a binary vector through function $b : \mathbb{R}^d \to \{-1, 1\}^d$ given by $b(\mathbf{x}) = \text{sign}(r(\mathbf{x}))$, where $r(\mathbf{x}) = \mathbf{x} - q(\mathbf{x})$ is the residual vector w.r.t. the nearest visual word and sign is the element-wise sign function.

The SMK similarity of two images, represented by $\mathcal{X}$ and $\mathcal{Y}$ respectively, is estimated by cross-matching all pairs of local descriptors with match kernel

$$S_{\text{SMK}}(\mathcal{X}, \mathcal{Y}) = \gamma(\mathcal{X})\,\gamma(\mathcal{Y}) \sum_{\mathbf{x}\in\mathcal{X}} \sum_{\mathbf{y}\in\mathcal{Y}} [q(\mathbf{x}) = q(\mathbf{y})] k(b(\mathbf{x}), b(\mathbf{y})), \tag{1}$$

where $[\cdot]$ is the Iverson bracket, $\gamma(\mathcal{X})$ is a scalar normalization that ensures unit self-similarity[3], *i.e.* $S_{\text{SMK}}(\mathcal{X}, \mathcal{X}) = 1$. Function $k : \{-1, 1\}^d \times \{-1, 1\}^d \to [0, 1]$ is given by

$$k(b(\mathbf{x}), b(\mathbf{y})) = \begin{cases} \left(\frac{b(\mathbf{x})^\top b(\mathbf{y})}{d}\right)^\alpha, & \frac{b(\mathbf{x})^\top b(\mathbf{y})}{d} \geq \tau \\ 0, & \text{otherwise,} \end{cases} \tag{2}$$

where $\tau \in [0, 1]$ is a threshold parameter. Only descriptor pairs that are assigned to the same visual word contribute to the image similarity in (1). In practice, not all pairs need to be enumerated and image similarity is equivalently given by

$$S_{\text{SMK}}(\mathcal{X}, \mathcal{Y}) = \gamma(\mathcal{X})\,\gamma(\mathcal{Y}) \sum_{\mathbf{c}\in\mathcal{C}_\mathcal{X}\cap\mathcal{C}_\mathcal{Y}} \sum_{\mathbf{x}\in\mathcal{X}_\mathbf{c}} \sum_{\mathbf{y}\in\mathcal{Y}_\mathbf{c}} k(b(\mathbf{x}), b(\mathbf{y})), \tag{3}$$

where cross-matching is only performed within common visual words.

ASMK first *aggregates* the local descriptors assigned to the same visual word into a single binary vector. This is performed by $B(\mathcal{X}_c) = \text{sign}\left(\sum_{x\in\mathcal{X}_c} r(\mathbf{x})\right)$, with $B(\mathcal{X}_c) \in \{-1, 1\}^d$. Image similarity in ASMK is given by

$$S_{\text{ASMK}}(\mathcal{X}, \mathcal{Y}) = \gamma(\mathcal{X})\,\gamma(\mathcal{Y}) \sum_{\mathbf{c}\in\mathcal{C}_\mathcal{X}\cap\mathcal{C}_\mathcal{Y}} k(B(\mathcal{X}_c), B(\mathcal{Y}_c)). \tag{4}$$

[2] The binarized versions are originally [44] referred to as SMK* and ASMK*. Only binarized versions are considered in this work and the asterisk is omitted.

[3] To simplify, we use the same notation, *i.e.* $\gamma(\cdot)$, for the normalization of different similarity measures in the rest of the text. In each case, it ensures unit self-similarity of the corresponding similarity measure.

This is computationally and memory-wise more efficient than SMK. In practice, it is known to perform better due to handling the burstiness phenomenon [18]. Efficient search is performed by using an inverted-file indexing structure.

Simplifications. Compared to the original approach [44], we drop IDF weighting, pre-binarization random projections, and median-value thresholding, as these are found unnecessary.

4 Method

Learning local descriptors with ASMK in an end-to-end manner is challenging and impractical due to the use of large visual codebooks and the hard-assignment of descriptors to visual words. In this section, we first describe a simple framework to generate global descriptors that can be optimized with image-level labels and provide an insight into why this is relevant to optimizing the local representation too. Then, we extend the framework by additional components and discuss their relation to prior work.

4.1 Derivation of the architecture

In the following, let us assume a deep Fully Convolutional Network (FCN), denoted by function $f : \mathbb{R}^{w\times h\times 3} \to \mathbb{R}^{W\times H\times D}$, that maps an input image I to a 3D tensor of activations $f(I)$. The FCN is used as an extractor of dense deep local descriptors. The 3D activation tensor can be equivalently seen as a set $\mathcal{U} = \{\mathbf{u} \in \mathbb{R}^D\}$ of $W \times H$ D-dimensional local descriptors[4]. Each local descriptor is associated to a keypoint, also called local feature, that is equivalent to the receptive field, or a fraction of it, of the corresponding activations.

Let us consider global image descriptors constructed by global sum-pooling, known as SPoC [4]. Pairwise image similarity is estimated by the inner product of the corresponding ℓ_2-normalized SPoC descriptors. This can be equivalently seen as an efficient match kernel. Let $\mathcal{U} = f(I)$ and $\mathcal{V} = f(J)$ be sets of dense feature descriptors in images I and J, respectively. Image similarity is given by

$$S_{\text{SPoC}}(\mathcal{U},\mathcal{V}) = \gamma(\mathcal{U})\,\gamma(\mathcal{V}) \sum_{\mathbf{u}\in\mathcal{U}} \sum_{\mathbf{v}\in\mathcal{V}} \mathbf{u}^\top \mathbf{v} \tag{5}$$

$$= \left(\gamma(\mathcal{U}) \sum_{\mathbf{u}\in\mathcal{U}} \mathbf{u}\right)^\top \left(\gamma(\mathcal{V}) \sum_{\mathbf{v}\in\mathcal{V}} \mathbf{v}\right) = Z_{\text{SPoC}}(\mathcal{U})^\top Z_{\text{SPoC}}(\mathcal{V}), \tag{6}$$

where $\gamma(\mathcal{U}) = 1/||\sum_{\mathbf{u}\in\mathcal{U}} \mathbf{u}||$. The optimization of the network parameters is cast as metric learning.

The interpretation of global descriptor matching in (6) as local descriptor cross-matching in (5) provides some useful insight. Local descriptor similarity

[4] Both $f(I)$ and $\mathcal{U}$ correspond to the same representation seen as a 3D tensor and a set of descriptors, respectively. We write $\mathcal{U} = f(I)$ implying the tensor is transformed into a set of vectors. $\mathcal{U}$ is, in fact, a multi-set, but it is referred to as set in the paper.

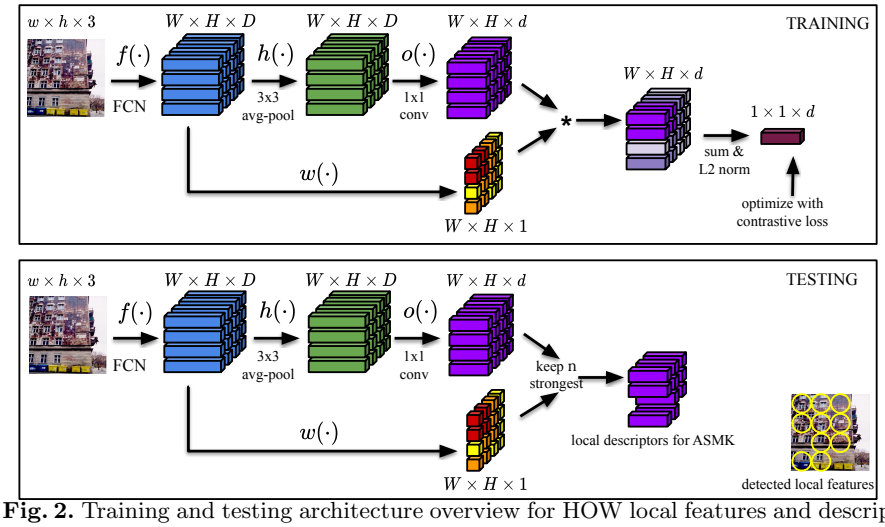


Fig. 2. Training and testing architecture overview for HOW local features and descriptors. A global descriptor is generated for each image during training and optimized with contrastive loss and image-level labels. During testing the strongest local descriptors (features), according to the attention map, are kept to represent the image. Then, these are used with ASMK for image search.

is estimated by $\mathbf{u}^\top \mathbf{v} = ||\mathbf{u}|| \cdot ||\mathbf{v}|| \cdot \cos(\mathbf{u}, \mathbf{v})$. Optimizing SPoC with image-level labels and contrastive loss implicitly optimizes the following four cases. First, local descriptors of background features, or image regions, are pushed to have small magnitude, *i.e.* small $||\mathbf{u}||$, so that their contribution in cross-matching is minimal. Similarly, local descriptors of foreground features are pushed to have large magnitude. Additionally, local descriptors of truly-corresponding features, *i.e.* image locations depicting the same object or object part, are pushed closer in the representation space, so that their inner product is maximized. Finally, local descriptors of non-corresponding features are pushed apart, so that their inner product is minimized. Therefore, optimizing global descriptors is a good surrogate to optimize local descriptors that are used to measure similarity via efficient match kernels, such as in ASMK. Importantly, this is possible with image-level labels and no local correspondences are required for training. In the following, we introduce additional components to the presented model, that are designed to amplify these properties.

Feature strength and attention. The feature strength, or importance, is estimated as the ℓ_2 norm of the feature descriptor $\mathbf{u}$ by the attention function $w(\mathbf{u}) = ||\mathbf{u}||$. During training, the feature strength is used to weigh the contribution of each feature descriptor in cross-matching. This way, the impact of the weak features is limited during the training, which is motivated by only the strongest features being selected in the test time. The attention function is fixed, without any parameters to be learned.

Local smoothing. Large activation values tend to appear in multiple channels of the activation tensor on foreground features [41]. Moreover, these activa-

tions tend not to be spatially well aligned. We propose to spatially smooth the activations by average pooling in an $M \times M$ neighborhood. The result is denoted by $\bar{\mathcal{U}} = h(f(I))$ or $\bar{\mathcal{U}} = h(\mathcal{U})$. Our experiments show that it is beneficial for the aggregation operation performed in ASMK. It is a fixed function, without any further parameters to be learned, and parameter M is a design choice.

Mean subtraction. Commonly used FCNs (all that are used in this work) generate non-negative activation tensors since a Rectified Linear Unit (ReLU) constitutes the last layer. Therefore, the inner product for all local descriptor pairs in cross-matching is non-negative and contributes to the image similarity. Mean descriptor subtraction is known to capture negative evidence [19] and allows to better disambiguate non-matching descriptors.

Descriptor whitening. Local descriptor dimensions are de-correlated by a linear whitening transformation; PCA-whitening improves the discriminability of both local [6] and global descriptors [36]. For efficiency, dimensionality reduction is performed jointly with the whitening. Formally, we group mean subtraction, whitening, and dimensionality reduction into function $o : \mathbb{R}^D \to \mathbb{R}^d$ given by $o(\mathbf{u}) = P(\mathbf{u} - \mathbf{m})$, where $P \in \mathbb{R}^{d \times D}$, $d \leq D$. Function $o(\cdot)$ is implemented by 1×1 convolution with bias. In practice, we initialize P and $\mathbf{m}$ according to the result of PCA whitening on a set of local descriptors from the training set and keep them fixed during the training.

Learning. Let $\bar{\mathcal{V}} = h(\mathcal{V})$ and $\bar{\mathbf{v}} \in \bar{\mathcal{V}}$ denote the activation vector after local average pooling $h(\cdot)$ at the same spatial location as $\mathbf{v} \in \mathcal{V}$. Similarly for $\bar{\mathcal{U}}$ and $\bar{\mathbf{u}}$. The image similarity that is being optimized during learning is expressed as

$$S_{how}(\mathcal{U}, \mathcal{V}) = \gamma(\bar{\mathcal{U}})\,\gamma(\bar{\mathcal{V}}) \sum_{\mathbf{u} \in \mathcal{U}} \sum_{\mathbf{v} \in \mathcal{V}} w(\mathbf{u}) \cdot w(\mathbf{v}) \cdot o(\bar{\mathbf{u}})^\top o(\bar{\mathbf{v}}) \tag{7}$$

$$= \left(\gamma(\bar{\mathcal{U}}) \sum_{\mathbf{u} \in \mathcal{U}} w(\mathbf{u}) o(\bar{\mathbf{u}})\right)^\top \left(\gamma(\bar{\mathcal{V}}) \sum_{\mathbf{v} \in \mathcal{V}} w(\mathbf{v}) o(\bar{\mathbf{v}})\right) = Z_{how}(\mathcal{U})^\top Z_{how}(\mathcal{V}). \tag{8}$$

A metric learning approach is used to train the network. In particular, contrastive loss is minimized: $||Z_{how}(\mathcal{U}) - Z_{how}(\mathcal{V}))||^2$ if $\mathcal{U}$ and $\mathcal{V}$ originate from matching (positive) image pairs, or $([\mu - ||Z_{how}(\mathcal{U}) - Z_{how}(\mathcal{V})||]_+)^2$ otherwise, where $[\cdot]_+$ denotes the positive part.

Test-time architecture. The architecture in test time is slightly modified; no global descriptor is generated. The n strongest descriptors $o(\bar{\mathbf{u}})$ for $\mathbf{u} \in \mathcal{U}$ are kept according to the importance given by $w(\mathbf{u})$. These are used as the local descriptor set $\mathcal{X}$ in ASMK. Multi-scale extraction is performed by resizing the input image at multiple resolutions. Local features from all resolutions are merged and ranked jointly according to strength. Selection of the strongest features is performed in the merged set. Multi-scale extraction is performed only during testing, and not during training. The training and testing architectures are summarized in Figure 2, while examples of detected features are shown in Figure 3.

Fig. 3. Example of multi-scale local feature detection. Left: Strongest 1,000 local features with color-coded strength; red is the strongest. Middle: Only the strongest feature per visual word is shown. Right: Attention maps for input images resized by scaling factors 0.25, 0.5, 1, and 2.

4.2 Relation to prior work

The proposed method has connections to different approaches in the literature which are discussed in this section. The work closest to ours, in terms of training of the local detector and descriptor, is DELF [29]. Following our notation, DELF generates local feature descriptors $Z_{\hat{h}\hat{o}\hat{w}}(\mathcal{U})$, where $\hat{h}(.)$ is identity, *i.e.* no local smoothing, $\hat{o}(.)$ is identity, *i.e.* no mean subtraction, whitening, and dimensionality reduction, and $\hat{w}(.)$ is a learned attention function. In particular, $\hat{w} : \mathbb{R}^D \to \mathbb{R}_+$ is a 2 layer convolutional network with 1×1 convolutions whose parameters are learned during the training. Dimensionality reduction of the descriptor space in DELF is performed as post-processing and is not a part of the optimization.[5] In terms of optimization, DELF performs the training in a classification manner with cross entropy loss. We show experimentally, that when combined with ASMK, the proposed descriptors are superior to DELF.

Fixed attention. Function $w(\cdot)$ is previously used to weigh activations and generate global descriptor, in particular by CroW [23]. It is also used in concurrent work for deep local features by Yang *et al.* [49]. The same attention function is used by Iscen *et al.* [17] for feature detection on top of dense SIFT descriptors without any learning. In our case, during learning, background or domain irrelevant descriptors are pushed to have low ℓ_2 norm in order to contribute less. The corresponding features are consequently not detected during test time.

Learned attention. Further example of learned attention, apart from DELF, is the contextual re-weighting that is performed to construct global descriptors in the work of Kim and Frahm [24]. The attention function is similar to DELF but with larger spatial kernel; a contextual neighborhood is used. A comparison between learned and fixed attention is included in the experimental ablations in Section 5.

Whitening. A common approach to whitening is to apply it as the last step in the pipeline, as post-processing. A similar approach to ours – applying the PCA whitening during training and learning it end-to-end – is followed by Gordo *et al.* [14] in the context of processing and aggregating regional descriptors to construct global descriptors. Comparison between "in-network" and post-processing whitening-reduction is included in Section 5.

[5] The main difference is that we do not follow the two stage training performed in the original work [29]; DELF is trained in a single stage for our ablations.

5 Experiments

We first review the datasets used for training, validation, and testing. Then, we discuss the implementation details for training and for testing with ASMK. Finally, we present our results on two instance-level tasks, namely recognition and search in the domain of landmarks and buildings.

5.1 Datasets

Training. The training dataset *SfM120k* [35] is used. It is the outcome of Stucture-from-Motion (SfM) [40] with 551 3D models for training. Matching pairs (anchor-positive) are formed by images with visual overlap (same 3D model). Non-matching pairs (anchor-negative) come from different 3D models.

Validation. We use the remaining 162 3D models of SfM120k to construct a challenging validation set reflecting the target task; this is different than the validation in [35]. We randomly choose 5 images per 3D model as queries. Then, for each query, images of the same 3D model with enough (more than 3), but not too many (at most 10), common 3D points with the query are marked as positive images to be retrieved. This avoids dominating the evaluation measure by a large number of easy examples. The remaining images of the same 3D model are excluded from evaluation for the specific query [32]. Skipping queries with an empty list of positive images results in 719 queries and 12,441 database images in total. Evaluation on the validation set is performed by instance-level search and performance is measured by mean Average Precision (mAP).

Evaluation on instance-level search. We use $\mathcal{R}$Oxford [32] and $\mathcal{R}$Paris [33] to evaluate search performance in the revisited benchmark [34]. They consist of 70 queries each, and 4993 and 6322 database images, respectively, and 1 million distractors called $\mathcal{R}$1M. We measure mAP on the Medium and Hard setups.

Evaluation on instance-level classification. We use instance-level classification as another task on which to evaluate the performance of the learned local descriptors. We perform search with ASMK and use k-nearest-neighbors classifiers for class predictions. The *Google Landmarks Dataset – version 2* (GLD_2) [48] is used. It consists of 4,132,914 train/database images with known class labels, and 117,577 test/query images which either correspond to the database landmarks, to other landmarks, or to non-landmark images. The query images are split into testing (private) and validation (public) sets with 76,627 and 40,950 images, respectively. Performance is measured by micro Average Precision (μAP) [31], also known as Global Average Precision (GAP), on the testing split. Note that we do not perform any learning on this dataset. We additionally create a mini version of GLD_2 to use for ablation experiments. It includes 1,000 query images, that are sampled from the testing split, and 10,0000 database images with labels, where the images come from 50 landmarks in total. We denote it by Tiny-GLD_2.

Classification is performed by accumulating the N top-ranked images per class. Prediction is given by the top-ranked class and the corresponding confidence is equal to the accumulated similarity. We use three variants, *i.e.* $N = 1$

(CLS1), $N = 10$ (CLS2), and $N = 10$ with accumulation of square-rooted similarity multiplied by a class weight (CLS3). The class weight is equal to the logarithm of the number of classes divided by the class frequency to down-weigh frequent classes.

5.2 Implementation details

Network architecture. We perform experiments with a backbone FCN ResNet18 and ResNet50, initialized by pre-training on ImageNet [39]. Descriptor dimensionality D is equal to 512 and 2048, respectively. We additionally experiment by removing the last block, *i.e.* "conv5_x", where D becomes 256 and 1024, respectively. We set $d = 128$ and $M = 3$ to perform 3×3 average pooling for local smoothing.

Training. We use a batch size of 5 tuples, where a tuple consists of 7 images – an anchor, a positive, and 5 negatives. Training images are restricted to a maximum resolution of 1,024 pixels. For each epoch, we randomly choose 2,000 anchor-positive pairs. The pool of candidate negatives contains 20,000 randomly chosen images, and hard-negative mining is performed before every epoch. We adopt the above choices from the work of Radenovic *et al.* [35], whose public implementation[6] we use to implement our method. To initialize P and $\mathbf{m}$, we use local descriptors from 5,000 training images extracted at a single scale. We use Adam optimizer with weight decay equal to 10^{-4}. The learning rate and margin μ are tuned, per variant, according to the performance on the validation set, by trying values $10^{-6}, 5 \cdot 10^{-6}, 10^{-5}, 5 \cdot 10^{-5}$ for learning rate and 0.5 to 0.9 with step 0.05 for margin μ. Margin μ is set equal to 0.8 for the proposed method, and learning rate equal to $5 \cdot 10^{-6}$ and 10^{-5} for ResNet18 and ResNet50 without the last block, respectively, according to the tuning process. Training is performed for 20 epochs and 1 epoch takes about 22 minutes for ResNet50 without the last block on a single NVIDIA Tesla V100 GPU with 32GB of DRAM. Training with cross entropy loss for ablations is performed with a batch size equal to 64 for 10 epochs. We repeat the training of each model and report mean and standard deviation over 5 runs. In large scale experiments, we evaluate a single model, the one with median performance on the validation set.

Validation. Validation performance is measured with ASMK-based search on the validation set. We measure validation performance every 5 epochs during training and the best performing model is kept.

Testing. We use ASMK to perform testing and to evaluate the performance of the learned local descriptors. The default ASMK configuation is as follows. We set threshold $\tau = 0$, $d = 128$, and use a codebook of $\kappa = 65536$ visual words, which is learned on local descriptors from 20,000 training images extracted at a single scale. Images are resized to have maximum resolution of 1024 pixels and multi-scale extraction is performed by re-scaling with factors 0.25, 0.353, 0.5, 0.707, 1.0, 1.414, 2.0. Assignment to multiple, in particular 5, visual words is performed for the query images. The strongest $n = 1000$ local descriptors are

[6] `https://github.com/filipradenovic/cnnimageretrieval-pytorch`

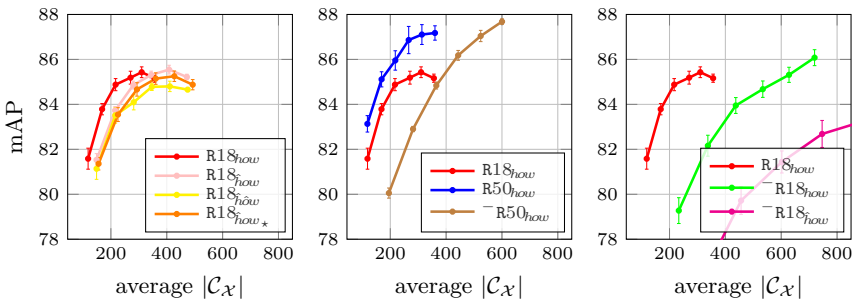


Fig. 4. Ablation study reporting performance versus average number of vectors per image in ASMK on the validation set for varying n (400,600,800,1000,1200,1400). Components used by DELF are denoted with $\hat{}$, while ours without. $\star$: random initialization of $\mathbf{m}$ and P (learned during training). Mean and standard deviation over 5 runs.

kept. The default configuration is used unless otherwise stated. The inverted file is compressed by delta coding.

5.3 Ablation experiments

We denote ResNet18 and ResNet50 by $\mathtt{R18}$ and $\mathtt{R50}$, and their versions with the last block skipped by $^{-}\mathtt{R18}$ and $^{-}\mathtt{R50}$, respectively. The local smoothing, whitening with reduction, and the fixed attention are denoted by subscripts h, o and w, respectively. The proposed method is denoted by $\mathtt{R18}_{how}$ when the backbone network is ResNet18. Following the same convention, the original DELF architecture is denoted by $^{-}\mathtt{R50}_{\hat{h}\hat{o}\hat{w}}$, where the dimensionality reduction is not part of the network but is performed by PCA whitening as post-processing.

Figure 4 shows the performance on the validation set versus the average number of binary vectors indexed by ASMK for the database images. We perform an ablation by excluding the proposed components and by using different backbone networks. Local smoothing results in larger amount of aggregation in ASMK (red vs pink) and reduces the memory requirements, which are linear in $|\mathcal{C}_\mathcal{X}|$. It additionally improves the performance when the last ResNet block is removed and feature maps have two times larger resolution (green vs magenta). Initializing and fixing $h(\cdot)$ with the result of PCA whitening is a beneficial choice too (orange vs pink). ResNet50 gives a good performance boost compared to ResNet18 (blue vs red), while removing the last block is able to reach higher performance at the cost of larger memory requirements (brown vs blue). More ablation experiments are shown in Table 1. Fixed attention is better than learned attention (3 vs 4). Metric learning with the contrastive loss delivers significantly better performance than cross entropy loss, in a classification manner, as done by Noh *et al.* [29] (4 vs 5). This is a confirmation of results that appear in the literature of instance-level recognition [14,46].

In Figure 5, we present the evolution of the model during training. We evaluate the performance of the optimized global descriptor for nearest neighbor

Method	Loss	Validation mAP	Validation $\lvert\mathcal{C}_\mathcal{X}\rvert$	$\mathcal{R}$Oxford mAP	$\mathcal{R}$Oxford $\lvert\mathcal{C}_\mathcal{X}\rvert$	Tiny-GLD$_2$ μAP	Tiny-GLD$_2$ $\lvert\mathcal{C}_\mathcal{X}\rvert$
1: R18$_{how}$	CO	85.2±0.3	263.8± 0.1	74.8±0.2	283.6± 0.2	81.3±1.0	252.2± 0.5
2: R18$_{\hat{h}ow}$	CO	85.3±0.2	344.1± 0.7	75.4±0.3	365.9± 0.5	80.6±0.3	332.8± 1.0
3: R18$_{\hat{h}\hat{o}w}$	CO	84.8±0.2	343.5± 2.7	73.1±0.3	365.7± 2.8	78.6±1.0	336.2± 3.5
4: R18$_{\hat{h}\hat{o}\hat{w}}$	CO	83.7±0.9	354.4± 2.0	70.0±1.7	380.7± 2.9	74.2±3.6	358.6± 5.5
5: R18$_{\hat{h}\hat{o}\hat{w}}$	CE	75.5±1.3	391.0± 8.2	63.7±1.6	442.3± 9.7	64.0±1.8	427.5±15.6
6: R18$_{\hat{h}\hat{o}w}$	CE	77.1±0.9	375.0± 8.5	67.0±1.0	429.0±13.8	67.3±2.1	417.8±11.7
7: R18$_{\hat{h}o\hat{w}}$	CE	78.4±0.8	354.6±10.5	67.8±1.3	402.8±12.2	66.7±1.5	367.2±14.8
8: R18$_{h\hat{o}\hat{w}}$	CE	77.0±0.9	279.6± 5.6	65.4±0.5	320.6± 6.8	68.6±1.8	300.9±11.4
9: R18$_{how}$	CE	80.4±0.5	308.3± 4.0	69.9±1.4	345.4± 5.2	71.1±1.8	308.4± 4.1

Table 1. Ablation study for performance and average number of descriptors per image in ASMK (mean and standard deviation over 5 runs). 1: our method, 5: DELF variant. CO: contrastive loss, CE: cross entropy. On Tiny-GLD$_2$, classifier CLS3 is used.

search with multi-scale global descriptors, *i.e.* aggregation of global descriptors extracted at multiple image resolutions (same set of 7 resolutions as for the local descriptors). We additionally evaluate performance of the corresponding local descriptors with ASMK. The descriptor that is directly optimized in the loss performs worse than the internal local descriptors.

In the following we use two backbone networks – R18, which is fast with low memory footprint, and $^{-}$R50, which achieves better results at the cost of slower extraction and more memory.

5.4 Large-scale instance-level search

The performance comparison on $\mathcal{R}$Oxford and $\mathcal{R}$Paris is presented in Table 2. We do our best to evaluate the best available variants or models of the state-of-the-art approaches. The proposed local descriptors and DELF descriptors are evaluated with identical implementation and configuration of ASMK. The

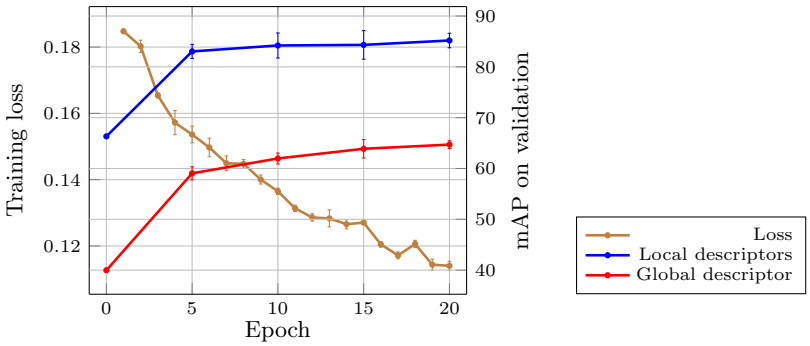


Fig. 5. Training loss and validation performance during the training. Mean and standard deviation (×5 for better visualization) is reported over 5 runs. Both performance curves correspond to the same model, only its inference differs.

Method	FCN	Mem (GB)	Mean		$\mathcal{R}$O		$\mathcal{R}$O+$\mathcal{R}$1M		$\mathcal{R}$Par		$\mathcal{R}$P+$\mathcal{R}$1M	
			all	$\mathcal{R}$1M	med	hard	med	hard	med	hard	med	hard
Global descriptors & Euclidean distance search												
R-MAC [14]	R101	7.6	45.8	33.6	60.9	32.4	39.3	12.5	78.9	59.4	54.8	28.0
GeM [35]	R101	7.6	47.4	35.5	64.7	38.5	45.2	19.9	77.2	56.3	52.3	24.7
GeM [35]†	R101	7.6	47.3	35.2	65.4	40.1	45.1	22.7	76.7	55.2	50.8	22.4
GeM_w†	R101	7.6	49.0	37.2	67.8	41.7	47.7	23.3	77.6	56.3	52.9	25.0
GeM_{AP}[37]	R101	7.6	49.9	37.1	67.5	42.8	47.5	23.2	80.1	60.5	52.5	25.1
GeM_{AP}[37]†	R101	7.6	49.7	36.7	67.1	42.3	47.8	22.5	80.3	60.9	51.9	24.6
GeM_{AP}[37] PQ1	R101	1.9	49.6	36.7	67.1	42.2	47.7	22.5	80.3	60.8	51.9	24.6
GeM_{AP}[37] PQ8	R101	0.2	48.1	35.5	65.1	40.4	46.1	21.6	78.7	58.7	50.7	23.5
GeM_{AP}[37]† ▷1024	R101	3.8	49.0	35.8	66.6	41.6	46.7	21.7	80.0	60.3	51.0	23.8
GeM_{AP}[37]† ▷512	R101	1.9	47.5	33.7	65.9	40.5	43.9	19.7	79.5	59.4	48.9	22.3
GeM_{AP}+ GeM_w†	R101	15.3	53.4	41.4	70.5	45.7	52.6	27.1	**81.9**	**63.4**	57.0	29.1
Local descriptors & ASMK												
DELF [29]†	⁻R50	9.2	53.0	43.1	69.0	44.0	54.1	31.1	79.5	58.9	59.3	28.1
DELF [29] ‡	⁻R50	9.2	52.5	42.7	69.2	44.3	54.3	31.3	78.7	57.4	58.2	26.9
DELF [29][34]	⁻R50	9.2	51.5	42.2	67.8	43.1	53.8	31.2	76.9	55.4	57.3	26.4
R18_{how}, $n = 1000$	R18	4.6	54.8	43.5	75.1	51.7	55.7	32.0	79.4	58.3	57.4	28.9
$^{-}\text{R50}_{how}$, $n = 1000$	⁻R50	7.9	58.0	47.4	78.3	55.8	63.6	36.8	80.1	60.1	58.4	30.7
$^{-}\text{R50}_{how}$, $n = 1200$	⁻R50	9.2	58.8	48.4	78.8	56.7	64.5	37.7	80.6	61.0	59.6	31.7
$^{-}\text{R50}_{how}$, $n = 1400$	⁻R50	10.6	59.3	49.0	79.1	56.8	64.9	38.2	81.0	61.5	60.4	32.6
$^{-}\text{R50}_{how}$, $n = 2000$	⁻R50	14.3	**60.1**	**50.1**	**79.4**	**56.9**	**65.8**	**38.9**	81.6	62.4	**61.8**	**33.7**

Table 2. Performance comparison with global and local descriptors for instance-level search on $\mathcal{R}$Oxford ($\mathcal{R}$O) and $\mathcal{R}$Paris ($\mathcal{R}$P). Memory is reported for $\mathcal{R}$1M. Methods marked by † are evaluated by us using the public models for descriptor extraction. The method marked by ‡ is evaluated by us using the public descriptors [34]. GEM_w is a public model that includes a "whitening" (FC) layer. Dimensionality reduction is denoted by ▷ and descriptor concatenation by +. PQ8 and PQ1 denote PQ quantization using 8D and 1D sub-spaces, respectively.

proposed descriptors outperform all approaches by a large margin at large scale; global descriptors perform well enough at small scale, but at large scale, local representation is essential. Even with a backbone network as small as ResNet18, we achieve the second best performance (ranked after our method with ResNet50) on all cases of $\mathcal{R}$Oxford and at the hard setup of $\mathcal{R}$Paris +$\mathcal{R}$1M. Compared to DELF, our descriptors, named *HOW*, perform better for less memory.

Teichmann *et al.* [43] achieve average performance (mean all in Table 2) equal to 56.0 and 57.3 without and with spatial verification, respectively. They use additional supervision, *i.e.* manually created bounding boxes for 94,000 images, which we do not. The concurrent work of Cao *et al.* [11] achieves average performance equal to 58.3 but requires 485 GB of RAM, and slightly lower performance with 22.6 GB of RAM for binarized descriptors.

In an effort to further compress the memory requirements of competing global descriptors, we evaluate the best variant with dimensionality reduction and with Product Quantization (PQ) [21]. We further improve their performance by concatenating the two best performing ones. Among all these variants, the proposed approach appears to be a good solution in the performance-memory trade-off.

Storing less vectors in ASMK affects memory but also speed. A query of average statistics computes the hamming distance for about $1.2 \cdot 10^6$ (average

Method	FCN	Training set	Memory (GB)	CLS1	CLS2	CLS3
GEM [35]	R101	SfM-120k	31.5	1.9	18.0	24.1
GEM_w	R101	SfM-120k	31.5	3.7	23.4	28.7
GEM-AP [37]	R101	SfM-120k	31.5	2.8	14.8	20.7
DELF [29]	$^{-}$R50	Landmarks [5]	24.1	2.1	11.9	21.9
R18_{how}	R18	SfM-120k	17.5	8.5	20.0	27.0
$^{-}\text{R50}_{how}$	$^{-}$R50	SfM-120k	29.3	18.5	33.1	36.5

Table 3. Performance comparison on instance-level recognition (GLD_2). μAP is reported for classification with 3 different k-nn classifiers. Existing methods are evaluated by us using the public models. Global descriptors are combined with simple nearest neighbor search, and local descriptors are combined with ASMK-based retrieval.

number of vectors stored per inverted list multiplied by average $|\mathcal{C}_\mathcal{X}|$) 128D binary vector pairs in the case of $^{-}$R18 with our method at large scale. The same number for $^{-}$R50 is $3.2 \cdot 10^6$. Search on $\mathcal{R}$Oxford $+\mathcal{R}$1M takes on average 0.75 seconds on a single threaded Python non-optimized CPU implementation.

5.5 Large-scale instance-level classification

Performance comparison on GLD_2 is presented in Table 3. We extract DELF keeping at most top 1,000 local descriptors and reduce dimensionality to 128. The proposed local descriptors and DELF descriptors are evaluated with identical configuration of ASMK. Our method outperforms all other methods with memory requirements that are even lower than raw 2048D global descriptors. DELF, R18_{how}, and $^{-}\text{R50}_{how}$, end up with 347, 252, and 423 vectors to store per image on average, respectively. Due to the strength threshold, DELF extracted 504 local descriptors per image on average which is significantly less than for images of $\mathcal{R}$1M. Multiple visual word assignment is not performed in this experiment, for any of the methods, to reduce the computational cost of search.

6 Conclusions

An architecture for extracting deep local features is designed to be combined with ASMK matching. The proposed method consistently outperforms other methods on a number of standard benchmarks, even if a less powerful backbone network is used. Through an extensive ablation study, we show that the SoA performance is achieved by the synergy of the proposed local feature detector with ASMK. We show that methods based on local features outperform global descriptors in large scale problems, and also that the proposed method outperforms other local feature detectors combined with ASMK. We demonstrate why the proposed architecture, despite being trained with image-level supervision only, is effective in learning image similarity based on local features.

Acknowledgements. The authors would like to thank Yannis Kalantidis for valuable discussions. This work was supported by MSMT LL1901 ERC-CZ grant. Tomas Jenicek was supported by CTU student grant SGS20/171/OHK3/3T/13.

References

1. Arandjelović, R., Zisserman, A.: All about VLAD. In: CVPR (2013)
2. Arandjelović, R., Zisserman, A.: DisLocation: Scalable descriptor distinctiveness for location recognition. In: ACCV (2014)
3. Arandjelović, R., Gronat, P., Torii, A., Pajdla, T., Sivic, J.: NetVLAD: CNN architecture for weakly supervised place recognition. In: CVPR (2016)
4. Babenko, A., Lempitsky, V.: Aggregating deep convolutional features for image retrieval. In: ICCV (2015)
5. Babenko, A., Slesarev, A., Chigorin, A., Lempitsky, V.: Neural codes for image retrieval. In: ECCV (2014)
6. Balntas, V., Lenc, K., Vedaldi, A., Mikolajczyk, K.: Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors. In: CVPR (2017)
7. Barroso Laguna, A., Riba, E., Ponsa, D., Mikolajczyk, K.: Key. net: Keypoint detection by handcrafted and learned cnn filters. In: ICCV (2019)
8. Bay, H., Ess, A., Tuytelaars, T., Gool, L.V.: SURF: Speeded up robust features. Computer Vision and Image Understanding **110**(3), 346–359 (May 2008)
9. Benbihi, A., Geist, M., Pradalier, C.: Elf: Embedded localisation of features in pre-trained cnn. In: CVPR (2019)
10. Bhowmik, A., Gumhold, S., Rother, C., Brachmann, E.: Reinforced feature points: Optimizing feature detection and description for a high-level task. In: CVPR (2020)
11. Cao, B., Araujo, A., Sim, J.: Unifying deep local and global features for efficient image search. In: arxiv (2020)
12. DeTone, D., Malisiewicz, T., Rabinovich, A.: Superpoint: Self-supervised interest point detection and description. In: CVPRW (2018)
13. Dusmanu, M., Rocco, I., Pajdla, T., Pollefeys, M., Sivic, J., Torii, A., Sattler, T.: D2-net: A trainable cnn for joint detection and description of local features. In: CVPR (2019)
14. Gordo, A., Almazan, J., Revaud, J., Larlus, D.: End-to-end learning of deep visual representations for image retrieval. IJCV (2017)
15. Gu, Y., Li, C., Jiang, Y.G.: Towards optimal cnn descriptors for large-scale image retrieval. In: ACM Multimedia (2019)
16. Husain, S., Bober, M.: Improving large-scale image retrieval through robust aggregation of local descriptors. PAMI **39**(9), 1783–1796 (Jan 2016)
17. Iscen, A., Tolias, G., Gosselin, P.H., Jégou, H.: A comparison of dense region detectors for image search and fine-grained classification. IEEE Transactions on Image Processing **24**(8), 2369–2381 (2015)
18. Jégou, H., Douze, M., Schmid, C.: On the burstiness of visual elements. In: CVPR (Jun 2009)
19. Jégou, H., Chum, O.: Negative evidences and co-occurences in image retrieval: The benefit of PCA and whitening. In: ECCV (Oct 2012)
20. Jégou, H., Douze, M., Schmid, C.: Improving bag-of-features for large scale image search. IJCV **87**(3), 316–336 (Feb 2010)
21. Jégou, H., Douze, M., Schmid, C.: Product quantization for nearest neighbor search. PAMI **33**(1), 117–128 (Jan 2011)
22. Jégou, H., Perronnin, F., Douze, M., Sánchez, J., Pérez, P., Schmid, C.: Aggregating local descriptors into compact codes. In: PAMI (Sep 2012)
23. Kalantidis, Y., Mellina, C., Osindero, S.: Cross-dimensional weighting for aggregated deep convolutional features. In: ECCVW (2016)

24. Kim, H.J., Dunn, E., Frahm, J.M.: Learned contextual feature reweighting for image geo-localization. In: CVPR (2017)
25. Matas, J., Chum, O., Urban, M., Pajdla, T.: Robust wide-baseline stereo from maximally stable extremal regions. Image and Vision Computing **22**(10), 761–767 (2004)
26. Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. PAMI **27**(10), 1615–1630 (2005)
27. Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., Gool, L.V.: A comparison of affine region detectors. IJCV **65**(1/2), 43–72 (2005)
28. Mohedano, E., McGuinness, K., O'Connor, N.E., Salvador, A., Marques, F., Giroi Nieto, X.: Bags of local convolutional features for scalable instance search. In: ICMR (2016)
29. Noh, H., Araujo, A., Sim, J., Weyand, T., Han, B.: Large-scale image retrieval with attentive deep local features. In: ICCV (2017)
30. Perronnin, F., Liu, Y., Sanchez, J., Poirier, H.: Large-scale image retrieval with compressed Fisher vectors. In: CVPR (2010)
31. Perronnin, F., Liu, Y., Renders, J.M.: A family of contextual measures of similarity between distributions with application to image retrieval. In: CVPR. pp. 2358–2365 (2009)
32. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching. In: CVPR (2007)
33. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Lost in quantization: Improving particular object retrieval in large scale image databases. In: CVPR (Jun 2008)
34. Radenović, F., Iscen, A., Tolias, G., Avrithis, Y., Chum, O.: Revisiting oxford and paris: Large-scale image retrieval benchmarking. In: CVPR (2018)
35. Radenović, F., Tolias, G., Chum, O.: Fine-tuning CNN image retrieval with no human annotation. PAMI **41** (Jul 2019)
36. Razavian, A.S., Sullivan, J., Carlsson, S., Maki, A.: Visual instance retrieval with deep convolutional networks. ITE Trans. on Media Technology and Applications (2016)
37. Revaud, J., Almazán, J., de Rezende, R.S., de Souza, C.R.: Learning with average precision: Training image retrieval with a listwise loss. In: ICCV (2019)
38. Revaud, J., Weinzaepfel, P., De Souza, C., Pion, N., Csurka, G., Cabon, Y., Humenberger, M.: R2d2: Repeatable and reliable detector and descriptor. In: NeurIPS (2019)
39. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. IJCV (2015)
40. Schönberger, J.L., Radenović, F., Chum, O., Frahm, J.M.: From single image query to detailed 3D reconstruction. In: CVPR (2015)
41. Siméoni, O., Avrithis, Y., Chum, O.: Local features and visual words emerge in activations. In: CVPR (2019)
42. Sivic, J., Zisserman, A.: Video Google: A text retrieval approach to object matching in videos. In: ICCV (2003)
43. Teichmann, M., Araujo, A., Zhu, M., Sim, J.: Detect-to-retrieve: Efficient regional aggregation for image search. In: CVPR (2019)
44. Tolias, G., Avrithis, Y., Jégou, H.: Image search with selective match kernels: aggregation across single and multiple images. IJCV (2015)

45. Tolias, G., Sicre, R., Jégou, H.: Particular object retrieval with integral max-pooling of CNN activations. In: ICLR (2016)
46. Vo, N., Jacobs, N., Hays, J.: Revisiting im2gps in the deep learning era. In: CVPR (2017)
47. Wang, Q., Zhou, X., Hariharan, B., Snavely, N.: Learning feature descriptors using camera pose supervision. In: arXiv (2020)
48. Weyand, T., Araujo, A., Cao, B., Sim, J.: Google landmarks dataset v2-a large-scale benchmark for instance-level recognition and retrieval. In: CVPR (2020)
49. Yang, T., Nguyen, D., Heijnen, H., Balntas, V.: Ur2kid: Unifying retrieval, keypoint detection, and keypoint description without local correspondence supervision. In: arxiv (2020)
50. Yue-Hei Ng, J., Yang, F., Davis, L.S.: Exploiting local features from deep networks for image retrieval. In: CVPR (2015)
51. Zhu, C.Z., Jégou, H., ichi Satoh, S.: Query-adaptive asymmetrical dissimilarities for visual object retrieval. In: ICCV (2013)

VIII Kernel Local Descriptors with Implicit Rotation Matching

Title: Kernel Local Descriptors with Implicit Rotation Matching
Authors: A. Bursuc, G. Tolias, H. Jégou
Published at: ICMR 2015

Kernel Local Descriptors with Implicit Rotation Matching

Andrei Bursuc, Giorgos Tolias, and Hervé Jégou
Inria
{firstname.lastname}@inria.fr

ABSTRACT

In this work we design a kernelized local feature descriptor and propose a matching scheme for aligning patches quickly and automatically. We analyze the SIFT descriptor from a kernel view and identify and reproduce some of its underlying benefits. We overcome the quantization artifacts of SIFT by encoding pixel attributes in a continuous manner via explicit feature maps. Experiments performed on the patch dataset of Brown *et al.* [3] show the superiority of our descriptor over methods based on supervised learning.

Categories and Subject Descriptors

I.4.10 [**Image Processing and Computer Vision**]: Image Representation

General Terms

Algorithms and Experimentation

Keywords

local descriptor; kernel descriptor; rotation invariance

1. INTRODUCTION

In this paper we deal with the design of a local descriptor, which is one of the fundamental problems in computer vision. A variety of tasks, such as image classification, retrieval, detection, and registration, rely on descriptors derived from local patches. The local patches can be originated from dense sampling [5] or from a sparse feature detector [7] which is co-variant to certain image transformations.

A variety of local descriptors builds upon image gradients. A standard choice is to build a histogram of gradient orientations, as in SIFT [7]. Another example concerns methods that rely on pixel intensities, *e.g.* by pairwise comparisons [4]. Learning [3] based on pairs of similar and non-similar patches is employed to select the spatial pooling regions and for dimensionality reduction [10].

In this work, we focus on a kernelized view of patch descriptors. In particular, we are motivated by the kernel descriptor of Bo *et al.* [2]. However, in our method we encode pixels in polar coordinates and rely on explicit feature maps [12], which provide a better approximation by Fourier series. We present a kernelized framework for local descriptors, out of which, known descriptors such as SIFT and our proposed descriptor are derived. This viewpoint of SIFT highlights some of its advantages and drawbacks: namely the encoding of the pixel position and the hard assignment which inevitably inserts some artifacts. We rather offer continuous encoding of pixel position and gradient orientations.

There are local descriptors that are rotation invariant by construction [8]. An alternative approach is to rely on the dominant orientation [7] of the patch in order to provide rotation invariance. The patches are rotated with respect to this dominant angle and transformed to up-right. Therefore, such descriptors are sensitive to this angle. In addition, in several tasks it is necessary to detect several multiple angles per patch, which further increases the computational cost of the procedures that follow. We adapt the latter choice of up-right patches and develop a fast patch alignment method with respect to rotation. Patch similarity is computed for multiple rotations at a slight increase of the computational cost. It allows us to handle the sensitivity to dominant orientation estimation and to dispense with the need for multiple dominant angles. This procedure is similar to the trigonometric polynomial of Tolias *et al.* [11], but at a patch level. It further resembles the way that Henriques *et al.* [6] speed-up learning with multiple shifted versions of negative samples, instead of performing costly sliding window based mining.

Our contribution includes a novel kernel local descriptor that encodes pixel position and gradient orientation in a continuous manner. It is also accompanied by an efficient way to compute patch similarity for multiple rotations, which constitutes our second contribution.

2. LOCAL DESCRIPTORS: KERNEL VIEW

Match kernels have gained an increasing interest after it was shown that it is possible to approximate non-linear kernels with linear ones by using an appropriate feature map [9, 12]. Similarity of sets can be efficiently computed with match kernels just by embedding the features in a suitable feature space in advance. Match kernels provide new grounds for designing better similarity functions.

Here, we define kernel descriptors over sets of pixels belonging to the same patch. An image patch can be considered as a set of pixels $\mathcal{X} = \{\mathbf{x}\}$. Without loss of generality we assume that we are dealing with grayscale patches. The relative position of a pixel $\mathbf{x}$ with respect to the patch center

ICMR'15, June 23–26, 2015, Shanghai, China.

http://dx.doi.org/10.1145/2671188.2749379.

is expressed in polar coordinates and denoted by $(\varphi_{\mathbf{x}}, \rho_{\mathbf{x}})$. We employ gradient information, thus each pixel is described by the gradient magnitude $m_{\mathbf{x}}$ and the gradient orientation $\theta_{\mathbf{x}}$. The latter is expressed relatively to the angle $\varphi_{\mathbf{x}}$ (Figure 3). The pixel position and its gradient information are referred as *pixel attributes* in the following. Two patches $\mathcal{X}$ and $\mathcal{Y}$ can be now compared via a match kernel of the form

$$\mathcal{K}(\mathcal{X}, \mathcal{Y}) = \gamma(\mathcal{X})\gamma(\mathcal{Y}) \sum_{\mathbf{x}\in\mathcal{X}} \sum_{\mathbf{y}\in\mathcal{Y}} k(\mathbf{x}, \mathbf{y}), \quad (1)$$

where k is the local kernel and γ is the normalization factor ensuring that self-similarity $\mathcal{K}(\mathcal{X}, \mathcal{X}) = 1$. The local kernel computes the similarity between two pixels $\mathbf{x}$ and $\mathbf{y}$, while the global kernel $\mathcal{K}$ accumulates similarities of all pairs of pixels. An interesting option is to obtain such a kernel by mapping pixels attributes to a higher-dimensional space with a feature map $\psi : \mathbf{x} \to \psi(\mathbf{x})$, such that the inner product evaluates the local kernel $k(\mathbf{x}, \mathbf{y}) = \langle \psi(\mathbf{x}) | \psi(\mathbf{y}) \rangle$. The match kernel can be now expressed as:

$$\mathcal{K}(\mathcal{X}, \mathcal{Y}) = \gamma(\mathcal{X})\gamma(\mathcal{Y}) \sum_{\mathbf{x}\in\mathcal{X}} \sum_{\mathbf{y}\in\mathcal{Y}} \langle \psi(\mathbf{x}) | \psi(\mathbf{y}) \rangle = \langle \mathbf{X} | \mathbf{Y} \rangle, \quad (2)$$

where $\mathbf{X} = \gamma(\mathcal{X}) \sum_{\mathbf{x}\in\mathcal{X}} \psi(\mathbf{x})$ and $\mathbf{Y} = \gamma(\mathcal{Y}) \sum_{\mathbf{y}\in\mathcal{Y}} \psi(\mathbf{y})$ are the local descriptors of patches $\mathcal{X}$ and $\mathcal{Y}$, respectively. Note that all the pairwise similarities are never explicitly enumerated; the linearity of the inner-product allows us to aggregate in advance the pixel feature vectors per patch.

Several popular methods for patch and image description can be actually described by this framework, among which SIFT. The SIFT descriptor is arguably one of the most popular and effective local feature descriptors in various computer vision applications. Some of the underlying advantages of SIFT can be emphasized better from a kernel perspective.

In the case of SIFT, consider that each pixel $\mathbf{x}$ is mapped to $\psi(\mathbf{x}) \in \mathbb{R}^{128}$, which is a sparse feature map due to the quantization of gradient orientation and spatial location. The aggregation of all pixel feature maps $\psi(\mathbf{x})$ results to the SIFT descriptor. The similarity of two SIFT descriptors can be then computed via inner product. The quantization to the spatial grid and to the orientation bins enforces to take into account only pixels with similar gradient orientations and spatial positions. The hard assignment in the quantization process inevitably inserts some artifacts and leads to a loss in the selectivity of the similarity function.

In the following, we design a kernelized local descriptor that imitates some of the advantages of the SIFT descriptor. At the same time, we alleviate some of drawbacks related to the quantization artifacts by encoding pixel position and gradient orientation in a continuous manner.

3. METHOD

We want to exploit jointly the photometric and position information of all patch elements (pixels). We target a kernel function for patch elements that reflects their resemblance in terms of gradients and their proximity in terms of their spatial position. To this effect, the local kernel $k(\mathbf{x}, \mathbf{y})$ can be decomposed into $k_\theta(\theta_{\mathbf{x}}, \theta_{\mathbf{y}})\, k_\varphi(\varphi_{\mathbf{x}}, \varphi_{\mathbf{y}})\, k_\rho(\rho_{\mathbf{x}}, \rho_{\mathbf{y}})$. It captures the similarity of the gradients with k_θ, and the spatial proximity on each coordinate separately with k_φ and k_ρ. Our match kernel now becomes

$$\mathcal{K}(\mathcal{X}, \mathcal{Y}) \propto \sum_{\mathbf{x}\in\mathcal{X}} \sum_{\mathbf{y}\in\mathcal{Y}} \tilde{m}_{\mathbf{x}}\, \tilde{m}_{\mathbf{y}}\, k_\theta(\theta_{\mathbf{x}}, \theta_{\mathbf{y}})\, k_\varphi(\varphi_{\mathbf{x}}, \varphi_{\mathbf{y}})\, k_\rho(\rho_{\mathbf{x}}, \rho_{\mathbf{y}}),$$

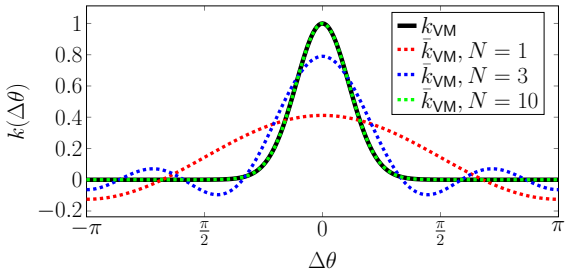


Figure 1: Target weighting function (Von Mises) and the corresponding approximations with 1, 3 and 10 frequencies.

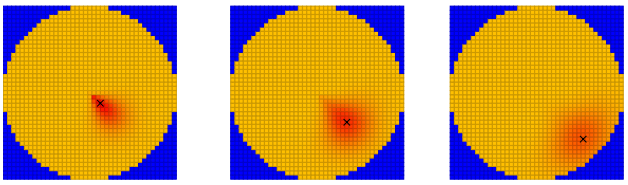


Figure 2: Visualization of the 2D weighting function for 3 sample pixels (shown by a cross). Colors reflect the spatial similarity to other pixels. Red (yellow) color corresponds to maximum (minimum) similarity. Blue color corresponds to the area that is neglected by our descriptor.

where the magnitude $\tilde{m}_{\mathbf{x}} = G(\rho_x, \sigma) * \sqrt{m_{\mathbf{x}}}$ is weighted by a Gaussian window in order to give higher importance to the patch center. In this fashion, our match kernel turns into scalar value comparison, such as angles and radii. Typically, non-linear functions are employed for such a comparison.

3.1 Feature maps for pixel attributes

We employ a normalized version of the Von Mises distribution [11] (normal distribution for angles) in order to compare two angles:

$$k_{\mathsf{VM}}(\theta_1, \theta_2) = k_{\mathsf{VM}}(\Delta\theta) = \frac{\exp(\kappa \cos(\Delta\theta)) - \exp(-\kappa)}{2 \sinh(\kappa)}. \quad (3)$$

This is a stationary kernel that depends only on the difference of the two angles $\Delta\theta = \theta_1 - \theta_2$. The selectivity of the function is controlled by parameter κ.

We define a mapping $\phi : [-\pi, \pi] \to \mathbb{R}^M$ of an angle θ to a vector $\phi(\theta)$ such that the inner product of two such vectors approximates the target function, that is $\phi(\theta_1)^\top \phi(\theta_2) = \bar{k}_{\mathsf{VM}}(\theta_1, \theta_2) \approx k_{\mathsf{VM}}(\theta_1, \theta_2)$. For this purpose we make use of explicit feature maps [12] and follow the methodology of Tolias *et al.* [11]. The desired mapping is

$$\phi(\theta) = (\sqrt{\gamma_0}, \sqrt{\gamma_1}\cos(\theta), \sqrt{\gamma_1}\sin(\theta), \ldots, \sqrt{\gamma_N}\cos(N\theta), \sqrt{\gamma_N}\sin(N\theta))^\top, \quad (4)$$

where γ_i is the i-th Fourier coefficient of Von Mises (3). The approximation by N Fourier coefficients (coresponding to N frequencies) produces a vector of $M = 2N + 1$ dimensions. The number of frequencies influences the accuracy of the approximation. Figure 1 illustrates the target function and its approximation for different values of N.

We choose the Von Mises to implement all 3 local kernels k_θ, k_φ and k_ρ. The mapping of Equation (4) is trivially used on local kernels k_θ and k_φ, since they deal with angles. We further map the radius of pixel $\mathbf{x}$ to an angle by $\tilde{\rho}_{\mathbf{x}} = \rho_{\mathbf{x}}\pi$, with $\rho_{\mathbf{x}} \in [0, 1]$. In this way, we are now able to use the same

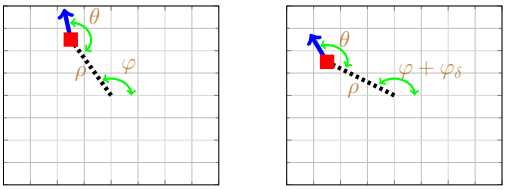


Figure 3: A sample patch and a corresponding pixel denoted by red rectangle. The gradient vector is shown by a blue arrow. The initial patch (left), is rotated by angle φ_δ (right). Both radius ρ and angle θ remain unchanged, while only angle φ changes.

mapping of scalar to vectors for the radius also. In Figure 2 we visualize the combination of the two local kernels k_ρ and k_φ (by their product) that evaluate the spatial proximity.

3.2 Wrapping up the descriptor

Each patch element $\mathbf{x}$ is now associated to three vectors describing its attributes, namely $\phi(\theta_\mathbf{x})$, $\phi(\varphi_\mathbf{x})$ and $\phi(\tilde{\rho}_\mathbf{x})$. In order to obtain a single descriptor we propose to describe $\mathbf{x}$ by the Kronecker product of these vectors, defined by $\psi(\mathbf{x}) = \tilde{m}_\mathbf{x}\phi(\theta_\mathbf{x}) \otimes \phi(\varphi_\mathbf{x}) \otimes \phi(\tilde{\rho}_\mathbf{x})$. By aggregating such vectors for all patch elements, the patch descriptor is now formed by $\mathbf{X} \propto \sum_{\mathbf{x}\in\mathcal{X}} \psi(\mathbf{x})$.

By using the Kronecker product properties we can show that comparing two such local descriptors via inner product is equivalent to the approximation of our match kernel:

$$\begin{aligned}\langle \mathbf{X}|\mathbf{Y}\rangle &\propto \sum_{\mathbf{x}\in\mathcal{X}} \psi(\mathbf{x})^\top \sum_{\mathbf{y}\in\mathcal{Y}} \psi(\mathbf{y}) = \sum_{\mathbf{x}\in\mathcal{X}}\sum_{\mathbf{y}\in\mathcal{Y}} \psi(\mathbf{x})^\top \psi(\mathbf{y}) \\ &\approx \sum_{\mathbf{x}\in\mathcal{X}}\sum_{\mathbf{y}\in\mathcal{Y}} \tilde{m}_\mathbf{x}\, \tilde{m}_\mathbf{y}\, k_\theta(\theta_\mathbf{x},\theta_\mathbf{y})\, k_\varphi(\varphi_\mathbf{x},\varphi_\mathbf{y})\, k_\rho(\tilde{\rho}_\mathbf{x},\tilde{\rho}_\mathbf{y}) \\ &= \mathcal{K}(\mathcal{X},\mathcal{Y}). \end{aligned} \quad (5)$$

The desired property of our mapping is the linearity of the inner product used to compare two vectors, which approximates a non-linear function comparing two angles. The pixel representation can be then aggregated in advance for each patch. The dimensionality of the local descriptor is equal to $(2N_\theta+1)(2N_\varphi+1)(2N_\rho+1)$, where different number of frequencies can be used for each pixel attribute (N_θ, N_φ and N_ρ) depending on the use-case. The descriptor is subsequently square-rooted and L_2-normalized.

3.3 Fast rotation alignment

At this stage, our match kernel and, equivalently, the kernel descriptor assume that all patches have the same global orientation. Patches are typically orientated to up-right position according to their dominant orientation. We adopt the same choice. However, this type of alignment can be quite noisy. We propose a method for identifying the rotation that maximizes the patch similarity and aligns them in an optimal way. We achieve this without explicitly computing the descriptors for all possible rotations.

Imagine that a patch $\mathcal{X}$ is rotated by an angle φ_δ into patch $\mathcal{X}_\delta$, and denote the descriptor of the rotated patch by $\mathbf{X}_\delta$. The only pixel attribute that changes is the angle φ, shifted by φ_δ, as illustrated in the toy example of Figure 3. Under this point of view and with respect to variable φ, it can be seen that local descriptor $\mathbf{X}$ is decomposed into the following sub-vectors $[\mathbf{X}_0{}^\top, \mathbf{X}_{1,c}{}^\top, \mathbf{X}_{1,s}{}^\top, \ldots, \mathbf{X}_{N,c}{}^\top, \mathbf{X}_{N,s}{}^\top]^\top$.

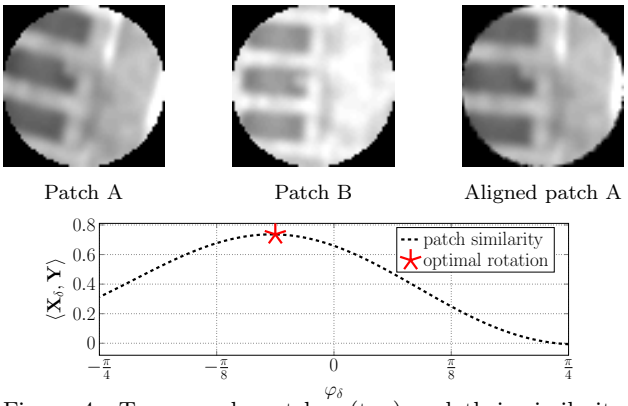


Figure 4: Two sample patches (top) and their similarity (bottom) for multiple rotations of patch A. At the (top) right, patch A is rotated by the optimal orientation.

$\mathbf{X}_0$ is constant for any patch rotation. The sub-vectors $\mathbf{X}_{i,c}$ and $\mathbf{X}_{i,s}$ are related to components of the form $cos(i\varphi_\mathbf{x})$ and $sin(i\varphi_\mathbf{x})$, respectively, for frequency i.

Interestingly, by using simple trigonometric identities, it turns out [11] that the similarity of two patches, when one of them undergoes rotation, forms a trigonometric polynomial:

$$\begin{aligned}\langle \mathbf{X}_\delta|\mathbf{Y}\rangle =& \langle \mathbf{X}_0|\mathbf{Y}_0\rangle + \sum_{n=1}^{N_\varphi} \cos(n\varphi_\delta)\ (\langle \mathbf{X}_{n,c}|\mathbf{Y}_{n,c}\rangle + \langle \mathbf{X}_{n,s}|\mathbf{Y}_{n,s}\rangle) \\ &+ \sum_{n=1}^{N_\varphi} \sin(n\varphi_\delta)\ (-\langle \mathbf{X}_{n,c}|\mathbf{Y}_{n,s}\rangle + \langle \mathbf{X}_{n,s}|\mathbf{Y}_{n,c}\rangle)\,. \end{aligned} \quad (6)$$

The complexity of computation of the polynomial coefficients in Equation (6) is less than twice the cost to compute the standard similarity between two kernel descriptors, while the cost to evaluate similarity for multiple angles φ_δ is negligible. In Figure 4 we present an example of the similarity of two patches under multiple rotations. The rotation of maximum similarity is used to align the patches.

4. EXPERIMENTS

We compare our descriptor against the state-of-the-art RootSIFT [1], and its counterpart rotated by PCA, noted as RootSIFT-PCA in the following. We further compare with the learned descriptors of Simonyan *et al.* [10] and the ones of Brown *et al.* [3] that rely on pairs of known related and non-related patches. We do not consider in our evaluation the descriptor of Bo *et al.* [2] as it is optimized to be used in image level aggregated manner for image classification, whereas we test patch similarities. We use a patch dataset [3] comprising three subsets, Notre Dame, Liberty and Yosemite, corresponding to different landmarks on which 3D reconstruction was performed. Each subset consists of 450k image patches of size 64×64 which are derived from keypoints detected with Difference-of-Gaussians. They are normalized with respect to scale and rotation. The ground-truth denotes the groups of similar patches, and two patches are considered similar if they are projections of the same 3D point.

4.1 Implementation details

We refer to our kernel descriptor as $KD_{N_\theta,N_\varphi,N_\rho}$ and evaluate for different number of frequencies. Parameter κ is always fixed and equal to 8, except for the case of $N_\rho = 1$

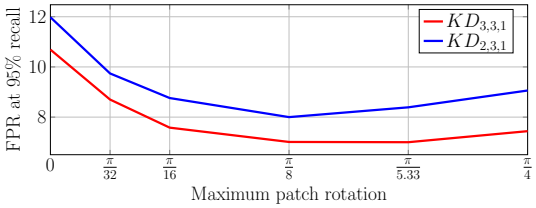


Figure 5: Impact of the rotation alignment on the performance. We evaluate patch similarity for 0, 4, 8, 16, 24 and 32 fixed rotations at each direction (clockwise and counter-clockwise). Results reported on Notre Dame dataset.

when κ is 2. In order to avoid artifacts for the computation of the rotated patches, we keep only the pixels inside the circle inscribed in the patch, as shown in Fig. 4.

Post-processing. The patch descriptor is power-law normalized with the power-law exponent α set to 0.5. For the PCA rotated variant, we obtain better results for $\alpha = 1$. For this case, we apply powerlaw normalization with $\alpha = 0.5$ after the projection. We proceed similarly for RootSIFT-PCA.

Orientation alignment. In order to align two patches we test up to 64 fixed rotations on each direction with a step of $\pi/128$. We find a good trade-off for rotations in the interval $[-\pi/8, \pi/8]$. Since the patches are already up-right, our algorithm reduces the quantization errors in the computation of the dominant orientations.

4.2 Hypothesis test

We evaluate our kernel descriptors following the standard protocol, generate the ROC curves and report false positive rate (FPR) at 95% recall. In Figure 5 we illustrate how performance improves by evaluating similarities for multiple patch rotations. After some extent performance decreases, since patches are already up-right by the rough dominant orientation and we are only introducing noisy matches.

We now consider each of the six possible combinations of training and test sets and we test 100k pairs for each run. Table 1 compares the error rates of our kernel descriptor against other local descriptors. Our descriptor is better than RootSIFT and RootSIFT-PCA and its performance is reaching that of learned descriptors [10]. While our kernel descriptor is higher dimensional, it doesn't require any training. We further evaluate it with PCA learned on a different dataset in order to obtain more compact descriptors. Although we cannot test multiple rotations anymore, the performance improves significantly outperforming more sophisticated methods trained over annotated pairs of similar and non-similar patches.

4.3 Nearest neighbors

We further evaluate our descriptor on a nearest neighbor search task using the patch dataset. We randomly select 1,000 query patches and report recall at the top R retrieved patches. This task is performed on a single subset at a time. Results for using Notre Dame as test set and Yosemite as learning set are reported in Table 2. Our kernel descriptors appear to perform the best, while the rotation alignment improves once more.

5. CONCLUSIONS

We proposed a kernel descriptor equipped with continuous encoding and a fast rotation alignment process. Interestingly, it achieves superior performance even compared to methods that employ supervised learning.[1]

Train	Test	*RootSIFT*	*RootSIFT-PCA*	*Simonyan [10]*	*Brown[3]*	$KD_{2,3,1}$	$KD_{3,3,1}$	$KD_{3,2,2}$-*PCA*
ND	Lib	29.64	19.34	**12.42**	16.85	21.58	20.06	13.17
Yos	Lib		19.95	14.58	18.27			**14.53**
ND	Yos	26.69	18.37	10.08	13.55	11.07	9.66	**8.31**
Lib	Yos		19.52	11.18	N/A			**9.65**
Lib	ND	22.06	13.90	7.22	N/A	7.99	7.00	**6.36**
Yos	ND		13.98	6.82	11.98			**6.50**
Mean		26.14	17.51	10.38	15.16	13.55	12.24	**9.75**
Dimensions		128	80	73-77	29-36	105	147	80
Learning		N	US	S	S	N	N	US

Table 1: False positive rate (%) at 95% recall. Learning type: *N*-none, *US*-unsupervised, *S*-supervised.

R	1	5	10	100	1000	10000
RootSIFT	8.5	24.4	33.0	62.9	79.7	90.6
RootSIFT-PCA	8.8	23.9	32.7	61.4	78.4	90.4
$KD_{3,3,1}$ (No rotations)	9.1	24.7	34.6	64.9	80.8	91.3
$KD_{3,3,1}$ (16 rotations)	**8.8**	**26.2**	**37.3**	**68.3**	**84.4**	**93.1**
$KD_{3,3,1} - PCA$	9.4	24.9	35.2	66.4	82.9	92.4

Table 2: Recall computed at R top ranked patches for 1000 randomly selected patch queries on Notredame dataset. Learning for PCA is performed on Yosemite dataset, and the dimensionality is reduced to 80 components.

6. REFERENCES

[1] R. Arandjelovic and A. Zisserman. Three things everyone should know to improve object retrieval. In *CVPR*, 2012.

[2] L. Bo, X. Ren, and D. Fox. Kernel descriptors for visual recognition. In *NIPS*, Dec. 2010.

[3] M. Brown, G. Hua, and S. Winder. Discriminative learning of local image descriptors. *Trans. PAMI*, 33(1):43–57, 2011.

[4] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. Brief: Binary robust independent elementary features. In *ECCV*, Oct. 2010.

[5] L. Fei-Fei and P. Perona. A bayesian hierarchical model for learning natural scene categories. In *CVPR*, 2005.

[6] J. F. Henriques, J. Carreira, R. Caseiro, and J. Batista. Beyond hard negative mining: Efficient detector learning via block-circulant decomposition. In *ICCV*, 2013.

[7] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, Nov. 2004.

[8] T. Ojala, M. Pietikainen, and T. Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *Trans. PAMI*, 24(7):971–987, 2002.

[9] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *NIPS*, 2007.

[10] K. Simonyan, A. Vedaldi, and A. Zisserman. Learning local feature descriptors using convex optimisation. *Trans. PAMI*, 2014.

[11] G. Tolias, T. Furon, and H. Jégou. Orientation covariant aggregation of local descriptors with embeddings. In *ECCV*, Sep. 2014.

[12] A. Vedaldi and A. Zisserman. Efficient additive kernels via explicit feature maps. *Trans. PAMI*, 34(3):480–492, Mar. 2012.

[1]This work was supported by ERC grant VIAMASS no. 336054 and ANR project Fire-ID.

IX Multiple-Kernel Local-Patch Descriptor

Title: Multiple-Kernel Local-Patch Descriptor
Authors: A. Mukundan, G. Tolias, O. Chum
Published at: BMVC 2017

Multiple-Kernel Local-Patch Descriptor

Arun Mukundan
arun.mukundan@cmp.felk.cvut.cz
Giorgos Tolias
giorgos.tolias@cmp.felk.cvut.cz
Ondřej Chum
chum@cmp.felk.cvut.cz

Visual Recognition Group
Czech Technical University in Prague

Abstract

We propose a multiple-kernel local-patch descriptor based on efficient match kernels of patch gradients. It combines two parametrizations of gradient position and direction, each parametrization provides robustness to a different type of patch miss-registration: polar parametrization for noise in the patch dominant orientation detection, Cartesian for imprecise location of the feature point. Even though handcrafted, the proposed method consistently outperforms the state-of-the-art methods on two local patch benchmarks.

1 Introduction

Representing and matching local features is an essential step of several computer vision tasks. It has attracted a lot of attention in the last decades, when local features still were a required step of most approaches. Despite the large focus on Convolutional Neural Networks (CNN) to process whole images, local features still remain important and necessary for tasks such as Structure-from-Motion (SfM) [11], stereo matching [20], or retrieval under severe change in viewpoint or scale [26].

Recently, the focus has shifted from hand-crafted descriptors to CNN-based descriptors. Learning such descriptors relies on large training sets of patches, that are commonly provided as a side-product of SfM [34]. Remarkable performance is achieved on a standard benchmark [5]. However, recent work [6, 27] shows that CNN-based approaches do not necessarily generalize equally well on different tasks or different datasets. Hand-crafted descriptors still appear an attractive alternative.

We build upon the hand-crafted kernel descriptor proposed by Bursuc *et al.* [9] that is shown to have good performance, even compared to learned alternatives. Its few parameters are easily tuned on some validation set, while it is shown to perform well on multiple tasks, as we confirm in our experiments. Post-processing with PCA and power-law normalization are shown beneficial.

Visualizing and analyzing the parametrization of this kernel descriptor allows us to understand its advantages and disadvantages, mainly the undesirable discontinuity around the patch center. We propose to combine multiple parametrizations and kernels to achieve robustness to different types of patch miss-registration. Experimental evaluation shows that the proposed descriptor outperforms all other approaches on two benchmarks designed to compare local-feature descriptors, specifically on the newly introduced HPatches dataset [6], and on the Phototourism benchmark [34].

2 Related work

We review prior work on local descriptors, covering both hand-crafted and learned ones.

Hand-crafted descriptors attracted a lot of attention for a decade and a variety of approaches and methodologies exists. A popular direction is that of gradient histogram-based descriptors, where the most popular representative is SIFT [17]. Different variants focus on pooling regions [16, 19], efficiency [1, 30], invariance [16] or other aspects [14]. Other are based on filter-bank responses [15], patch intensity [10, 25] or ordered intensity [21].

Kernel descriptors based on the idea of Efficient Match Kernels (EMK) [7] encode entities inside a patch (such a gradient, color, *etc*) in a continuous domain, rather than as a histogram. The kernels and their few parameters are often hand-picked and tuned on a validation set. Kernel descriptors are commonly represented by a finite-dimensional explicit feature maps. Quantized descriptors, such as SIFT, can be also interpreted as kernel descriptors [8, 9].

Learned descriptors commonly require annotation at patch level. Therefore, research in this direction is facilitated by the release of datasets that are originate from an SfM system [22, 34]. Such training datasets allow effective learning of local descriptors, and in particular, their pooling regions [29, 34], filter banks [34], transformations for dimensionality reduction [29] or embeddings [23].

Kernelized descriptors are formulated within a supervised framework by Wang *et al*. [33], where image labels enable kernel learning and dimensionality reduction. In this work, we rather focus on minimal learning in the form of discriminatively learned projections. This is several orders of magnitude faster to learn than other learning approaches.

Recently, learning local descriptor is dominated by deep learning. The network architectures are smaller than the corresponding ones performing on images, and use a large amount of training patches. Among representative examples is the work of Simo-Serra *et al*. [28] training with hard positive and negative examples or the work of Zagoruyko [35] where a central-surround representation is found to be immensely beneficial. CNN-based approaches are seen as joint feature, filter bank, and metric learning [12]. Finally, the state of the art consists of shallower architectures with improved ranking loss [4, 5]. Despite obtaining impressive results on a standard benchmark, CNN-based approaches do not generalize well to other datasets and tasks [6, 27].

A post-processing step is common to both hand-crafted and learned descriptors. This post-processing ranges from simple ℓ_2 normalization, PCA dimensionality reduction, to transformations learned on annotated data.

3 Preliminaries

Kernelized descriptors. In general lines we follow the formulation of Bursuc *et al*. [9]. We represent a patch $\mathcal{P}$ as a set of pixels $p \in \mathcal{P}$ and compare two patches $\mathcal{P}$ and $\mathcal{Q}$ via match kernel

$$\mathcal{M}(\mathcal{P},\mathcal{Q}) = \sum_{p\in\mathcal{P}} \sum_{q\in\mathcal{Q}} k(p,q), \tag{1}$$

where kernel $k : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is a similarity function, typically non-linear, comparing two pixels. EMK uses an explicit feature map $\psi : \mathbb{R}^n \to \mathbb{R}^d$ to approximate this result as

$$\mathcal{M}(\mathcal{P},\mathcal{Q}) = \sum_{p\in\mathcal{P}} \sum_{q\in\mathcal{Q}} k(p,q) \approx \sum_{p\in\mathcal{P}} \sum_{q\in\mathcal{Q}} \psi(p)^\top \psi(q) = \sum_{p\in\mathcal{P}} \psi(p)^\top \sum_{q\in\mathcal{Q}} \psi(q). \tag{2}$$

Vector $\mathbf{V}(\mathcal{P}) = \sum_{p\in\mathcal{P}} \psi(p)$ is a *kernelized descriptor* (KD), associated with patch $\mathcal{P}$, used to approximate $\mathcal{M}(\mathcal{P},\mathcal{Q})$, whose explicit evaluation is costly. The approximation is given by a dot product $\mathbf{V}(\mathcal{P})^\top \mathbf{V}(\mathcal{Q})$, where $\mathbf{V}(\mathcal{P}) \in \mathbb{R}^d$. To ensure a unit self similarity, ℓ_2 normalization by a factor γ is introduced. The normalized KD is then given by $\bar{\mathbf{V}}(\mathcal{P}) = \gamma(\mathcal{P})\mathbf{V}(\mathcal{P})$, where $\gamma(\mathcal{P}) = (\mathbf{V}(\mathcal{P})^\top \mathbf{V}(\mathcal{P}))^{-1/2}$.

Kernel k comprises product of kernels that act on scalar pixel attributes

$$k(p,q) = k_1(p_1,q_1)k_2(p_2,q_2)\dots k_n(p_n,q_n), \tag{3}$$

where kernel k_n is pairwise similarity function for scalars and p_n are pixel attributes such as position and gradient orientation. Feature map ψ_n corresponds to kernel k_n and feature map ψ is constructed via Kronecker product of individual feature maps $\psi(p) = \psi_1(p_1) \otimes \psi_2(p_2) \otimes \dots \otimes \psi_n(p_n)$. Due to the mixed product property it holds that $\psi(p)^\top \psi(q) \approx k_1(p_1,q_1)k_2(p_2,q_2)\dots k_n(p_n,q_n)$.

Feature maps. As non-linear kernel for scalars we use the normalized Von Mises probability density function[1], which is used for image [31] and patch [9] representation. It is parametrized by κ controlling the shape of the kernel, where lower κ corresponds to wider kernel. We use a stationary kernel that, by definition, depends only on the difference $\Delta_n = p_n - q_n$, *i.e.* $k_{\mathsf{VM}}(p_n,q_n) := k_{\mathsf{VM}}(\Delta_n)$. We adopt a Fourier series approximation with N frequencies that produces a feature map $\psi_{\mathsf{VM}} : \mathbb{R} \to \mathbb{R}^{2N+1}$. It has the property that $k_{\mathsf{VM}}(p_n,q_n) \approx \psi_{\mathsf{VM}}(p_n)^\top \psi_{\mathsf{VM}}(q_n)$. The reader is encouraged to read prior work for details on these feature maps [32], which are previously used in various contexts [9, 31].

Descriptor post-processing. It is known that further descriptor post-processing [3, 9, 24] is beneficial. In particular, KD is further centered and projected as

$$\hat{\mathbf{V}}(\mathcal{P}) = A^\top(\bar{\mathbf{V}}(\mathcal{P}) - \mu), \tag{4}$$

where $\mu \in \mathbb{R}^d$ and $A \in \mathbb{R}^{d\times d}$ are the mean vector and the projection matrix. These are commonly learned by PCA [13] or with supervision [24]. The final descriptor is always ℓ_2-normalized in the end.

4 Method

In this section we consider different patch parametrizations and kernels that result in different patch similarity. We discuss the benefits of each and propose how to combine them. We further learn descriptor transformation with supervision and provide useful insight on how patch similarity is affected.

Patch attributes. We consider a pixel p to be associated with coordinates p_x, p_y in Cartesian coordinate system, coordinates p_ρ, p_ϕ in polar coordinate system, pixel gradient magnitude p_m, and pixel gradient angle p_θ. Angles p_θ, $p_\phi \in [0,2\pi]$, distance from the center p_ρ is normalized to $[0,1]$, while coordinates p_x, $p_y \in \{1,2,\dots,W\}$ for $W \times W$ patches. In order to use feature map ψ_{VM}, attributes p_ρ, p_x, and p_y are linearly mapped to $[0,\pi]$. The gradient angle is expressed *w.r.t.* the patch orientation, *i.e.* p_θ directly, or *w.r.t.* to the position of the pixel. The latter is given as $p_{\tilde{\theta}} = p_\theta - p_\phi$.

[1] Also known as the periodic normal distribution

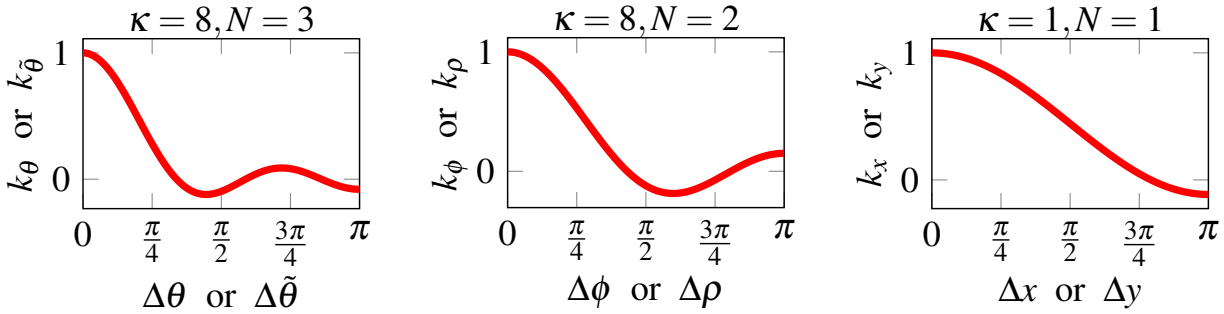


Figure 1: Kernel approximations that we use for pixel attributes. Parameter κ and the number of frequencies N define the final shape. The choice of kernel parameters is guided by [9].

Patch parametrizations. Composing patch kernel k as a product of kernels over different attributes enables easy design of various patch similarities. Correspondingly, this defines different KD. All attributes p_x, p_y, p_ρ, p_θ, p_ϕ, and $p_{\tilde{\theta}}$ are matched by the Von Mises kernel, namely, k_x, k_y, k_ρ, k_θ, k_ϕ, and $k_{\tilde{\theta}}$ parameterized by κ_x, κ_y, κ_ρ, κ_θ, κ_ϕ, and $\kappa_{\tilde{\theta}}$, respectively.

In this work we focus on the two following match kernels over patches. One in *polar* coordinates

$$\mathcal{M}_{\phi\rho\tilde{\theta}}(\mathcal{P},\mathcal{Q}) = \sum_{p\in\mathcal{P}}\sum_{q\in\mathcal{Q}} p_g q_g \sqrt{p_m}\sqrt{q_m} k_\phi(p_\phi,q_\phi) k_\rho(p_\rho,q_\rho) k_{\tilde{\theta}}(p_{\tilde{\theta}},q_{\tilde{\theta}}), \tag{5}$$

and one in *cartesian* coordinates

$$\mathcal{M}_{xy\theta}(\mathcal{P},\mathcal{Q}) = \sum_{p\in\mathcal{P}}\sum_{q\in\mathcal{Q}} p_g q_g \sqrt{p_m}\sqrt{q_m} k_x(p_x,q_x) k_y(p_y,q_y) k_\theta(p_\theta,q_\theta), \tag{6}$$

where $p_g = exp(-p_\rho^2)$ gives more importance to central pixels, in a similar manner to SIFT.

The KD for the two cases are given by

$$\mathbf{V}_{\phi\rho\tilde{\theta}}(\mathcal{P}) = \sum_{p\in\mathcal{P}} p_g p_m \psi_\phi(p_\phi)\otimes\psi_\rho(p_\rho)\otimes\psi_{\tilde{\theta}}(p_{\tilde{\theta}}) = \sum_{p\in\mathcal{P}} p_g\sqrt{p_m}\psi_{\phi\rho\tilde{\theta}}(p) \tag{7}$$

$$\mathbf{V}_{xy\theta}(\mathcal{P}) = \sum_{p\in\mathcal{P}} p_g p_m \psi_x(p_x)\otimes\psi_y(p_y)\otimes\psi_\theta(p_\theta) = \sum_{p\in\mathcal{P}} p_g\sqrt{p_m}\psi_{xy\theta}(p). \tag{8}$$

The $\mathbf{V}_{\phi\rho\tilde{\theta}}$ variant is exactly the one proposed by Bursuc *et al.* [9], considered as a baseline in this work. Different parametrizations result in different patch similarity, which is analyzed in the following. In Figure 1 we present the approximation of kernels used per attribute.

Descriptor post-processing with supervision. Mean vector μ and projection matrix A can be learned in an unsupervised way, *e.g.* by PCA on a sample descriptor set. In such case, matrix A is formed by the eigenvectors as columns. This is the case in prior work, not only for local descriptors [9] but also for global image representation [13]. It was previously observed, and our experiments confirm, that discriminative projection [18] learned on labeled data outperforms post-processing by generative model, such as PCA. The discriminative projection is composed of two parts, a whitening part and a rotation part. The whitening part is obtained from the intraclass (matching pairs) covariance matrix, while the rotation part is the PCA of the interclass (non-matching pairs) covariance matrix in the whitened space. Vector μ is the mean descriptor vector. To reduce the descriptor dimensionality, only eigenvectors corresponding to the largest eigenvalues are used. We refer to this transformation as learned (supervised) whitening (LW) in the rest of the paper.

Visualization of patch similarity. We define pixel similarity $\mathcal{M}(p,q)$ as kernel response between pixels p and q, approximated as $\mathcal{M}(p,q) \approx \psi(p)^\top \psi(q)$. To show a spatial distribution of the influence of pixel p, we define a *patch map* of pixel p. The patch map has the same size as the image patches, for each pixel q of the patch, map $\mathcal{M}(p,q)$ is evaluated for some constant value of q_θ.

For example, in Figure 2 patch maps for different kernels are shown. The position of p is denoted by $\times$ symbol. The value of $p_\theta = 0$ and $q_\theta = 0$ for all spatial locations of q in the top row and $q_\theta = -\pi/8$ in the bottom row. The visualization shows the discontinuity of the pixel similarity impact of the $\mathbf{V}_{\phi\rho\tilde{\theta}}$ descriptor near the center of the patch. This is caused by the polar coordinate system where a small difference in the position near the origin causes large difference in ϕ and $\tilde{\theta}$. Also in the bottom row we see that using the relative gradient direction $\tilde{\theta}$ allows to compensate for imprecision caused by small patch rotation, *i.e.* the most similar pixel is not the one at the location of p with different $\tilde{\theta}$, but a rotated pixel with more similar value of $\tilde{\theta}$. Finally, we observe that the kernel parametrized by Cartesian coordinates and absolute angle of the gradient ($\mathbf{V}_{xy\theta}$, third column) is insensitive to small translations, *i.e.* feature point displacement.

We additionally construct patch maps in the case of descriptor post-processing by a linear transformation, *e.g.* descriptor whitening. Now the contribution of a pixel pair is given by

$$\hat{\mathcal{M}}(p,q) = (A^\top(\psi(p)-\mu))^\top (A^\top(\psi(q)-\mu)) \tag{9}$$

$$= (\psi(p)-\mu)^\top AA^\top(\psi(q)-\mu) \tag{10}$$

$$= \psi(p)^\top AA^\top\psi(q) - \psi(p)^\top AA^\top\mu - \psi(q)^\top AA^\top\mu + \mu^\top AA^\top\mu. \tag{11}$$

The last term is constant and can be ignored, while if A is a rotation matrix then only shifting by μ affects the similarity. After the transformation, the similarity is no longer shift-invariant. The non-linear post-processing, such as power-law normalization or simple ℓ_2 normalization cannot be visualized, as it acts after the pixel aggregation[2].

Figure 3 we shows patch maps for $\mathbf{V}_{\phi\rho\tilde{\theta}}$ in the case of PCA or LW post-processing. PCA is shown to have some small effect on the similarity, while LW significantly changes the derived shape. It implicitly affects the shape of the kernels used; observe that the kernels go wider in the circular direction.

Combining kernel descriptors. We propose to take advantage of both parametrizations $\mathbf{V}_{\phi\rho\tilde{\theta}}$ and $\mathbf{V}_{xy\theta}$, by summing their contribution. This is performed by simple concatenation of the two descriptors. Finally, whitening is jointly learned and dimensionality reduction is performed.

In Figure 4 we show patch maps for the individual and combined representation, before and after applying learned whitening. Observe how the combined one better behaves around the center but also how the final similarity is formed after the whitening.

[2]Details are omitted due to lack of space.

[3]Ten isocontours are sampled uniformly. The similarity is shown in a relative manner and, therefore, the absolute scale is missing (*e.g.* in Figure 2 the maximum value is larger in top row compared to bottom due to $k_\theta(0) > k_\theta(\pi/8)$).

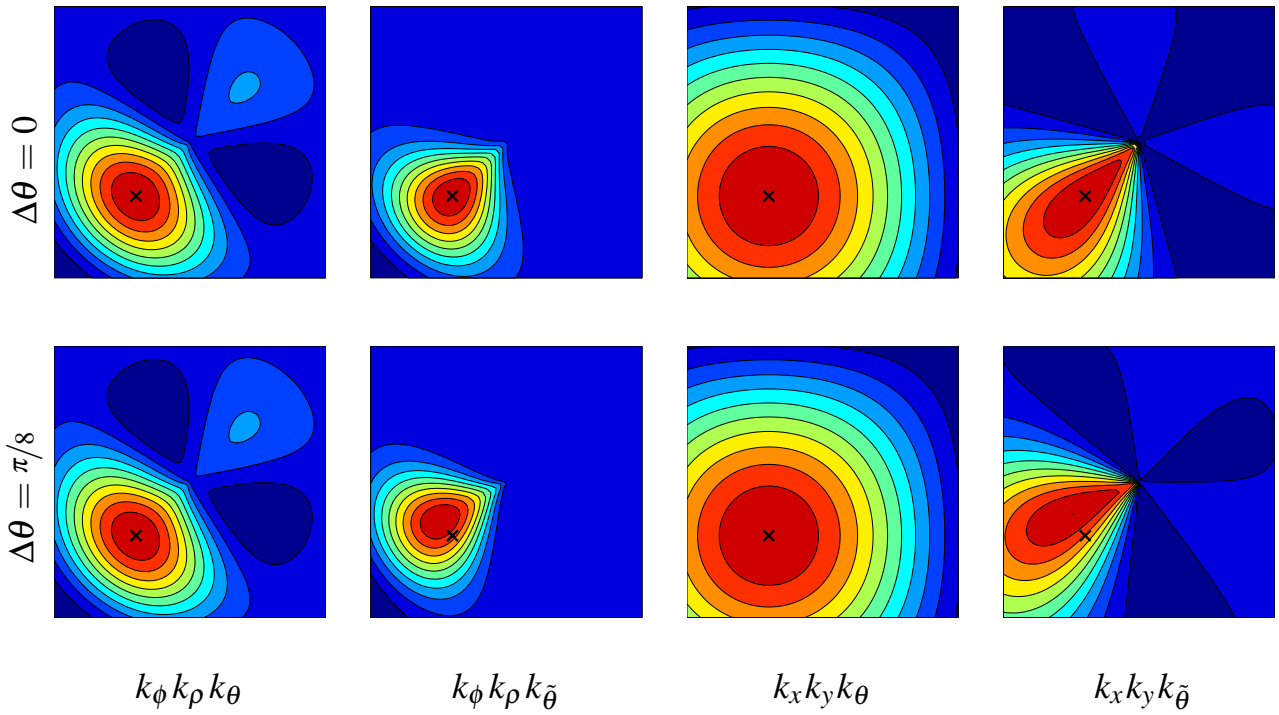


Figure 2: Patch maps for different parametrizations and kernels. We present two parametrizations in polar and two in cartesian coordinates, with absolute or relative gradient angle for each one. $\Delta\theta$ is fixed and pixel p is shown with "$\times$". At the bottom of each column the kernels (patch similarity) approximated are shown.[3]

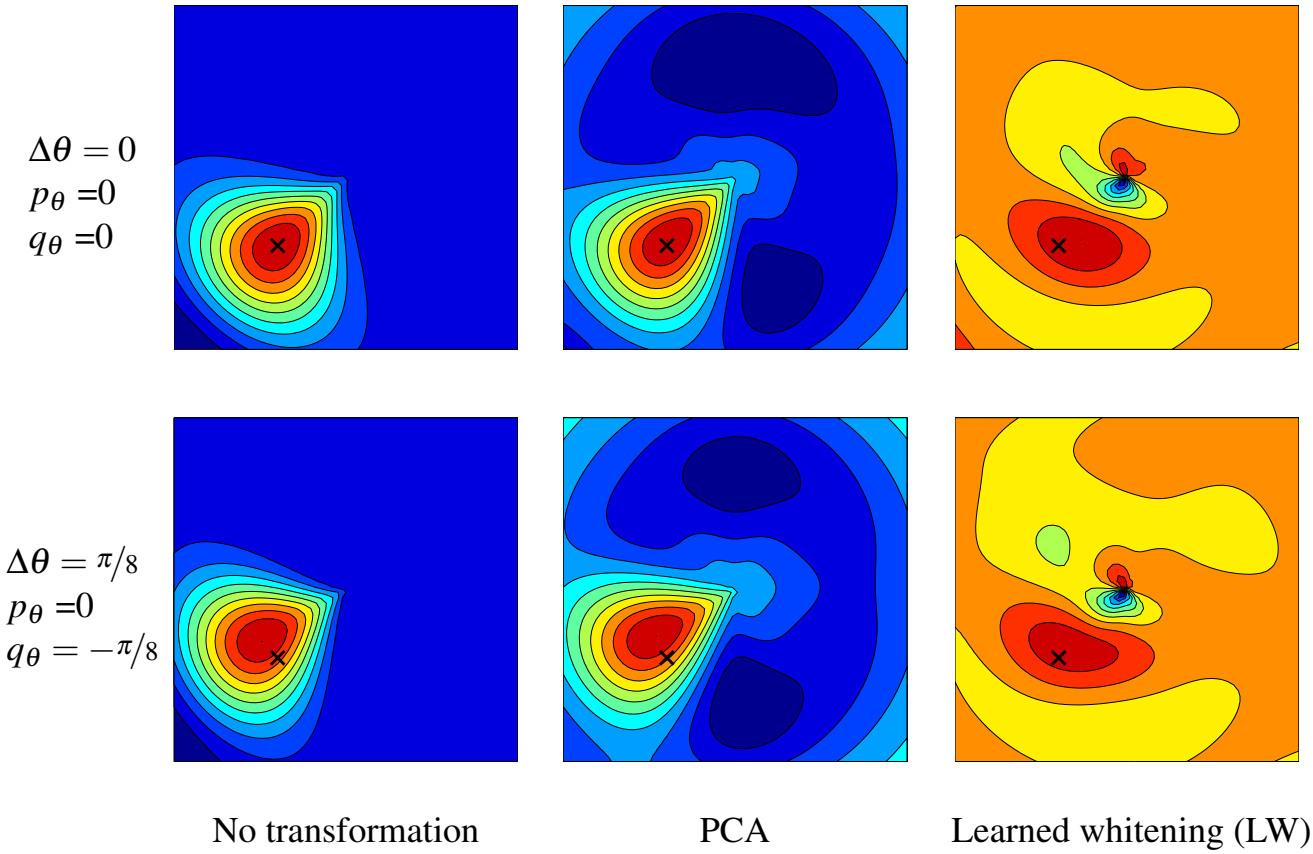


Figure 3: Patch maps for $\phi\rho\tilde{\theta}$ parametrization and kernels. $\Delta\theta$ is fixed by choosing fixed values for p_θ and q_θ. Pixel p is shown with "$\times$". Three different cases are shown: without transformation, with PCA transformation and with transformation by supervised whitening.[3]

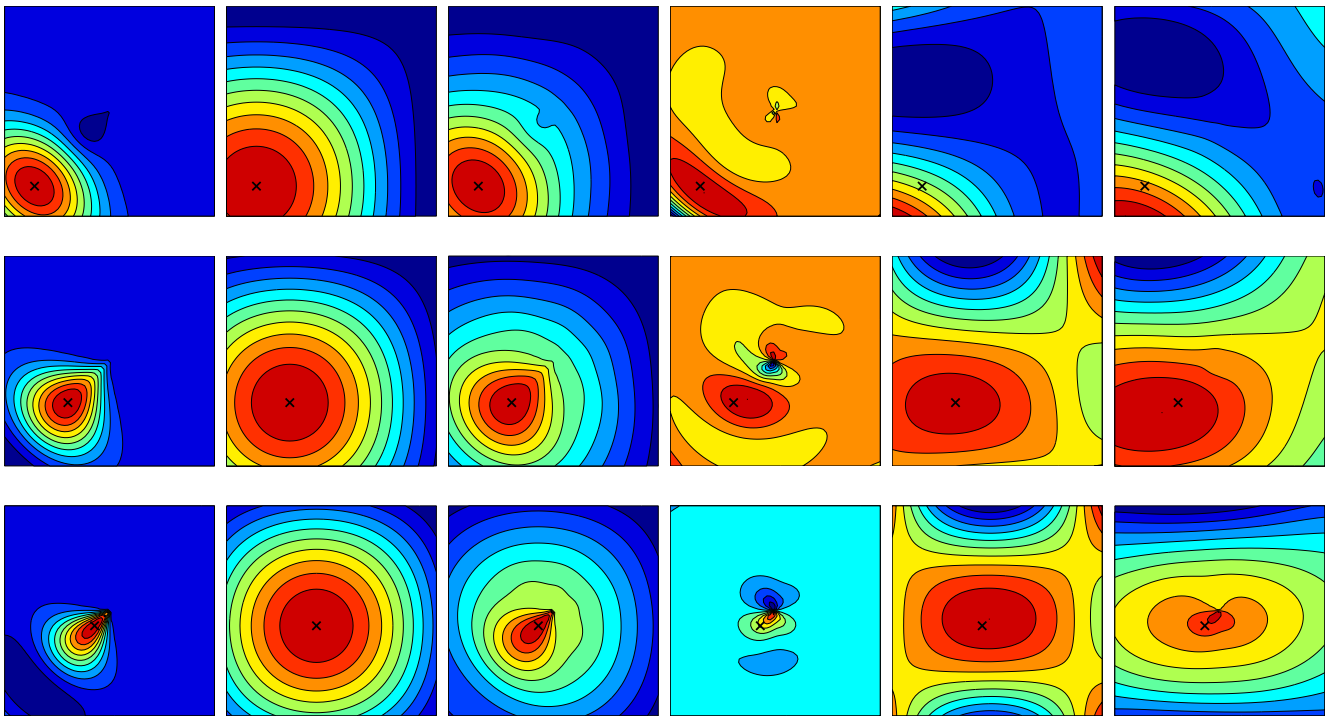


Figure 4: Patch maps for different parametrizations and kernels. We present polar and cartesian parametrization separately, and their combination by descriptor concatenation. We present the case for 3 different pixels p (one pixel per row) shown with "$\times$". $\Delta\theta = 0$ in all examples, in particular $p_\theta = 0$ and $q_\theta = 0$. The cases without descriptor transformation and with transformation by supervised whitening (LW) are shown.[3]

5 Experiments

We evaluate the method on two benchmarks, namely the widely used *Phototourism* (PT) dataset [34], and the recently released *HPatches* (HP) dataset [6]. We first compare the proposed method with the baseline method of Bursuc *et al.* [9] and then with the state-of-the-art methods on the two datasets. In all our experiments with descriptor post-processing the dimensionality is reduced to 128 except for the cases where the input descriptor is already of lower dimension.

Datasets and protocols. The Phototourism dataset contains three sets of patches, namely, Liberty (Li), Notredame (No) and Yosemite (Yo). Additionally, labels are provided to indicate the 3D point that the patch corresponds to, thereby providing supervision. It has been widely used for training and evaluating local descriptors. Performance is measured by the false positive rate at 95% of recall (FPR95). The protocol is to train on one of the three sets and test on the other two. An average over all six combinations is reported.

The HPatches dataset contains local patches of higher diversity, is more realistic, and during evaluation the performance is measured on three tasks: *verification*, *retrieval*, and *matching*. We follow the standard evaluation protocol [6] and report mean Average Precision (mAP). When evaluating on HP, we follow the protocol and learn the whitening on PhotoTourism Liberty, or on a pre-defined split of test and train of the HPatches dataset provided by the authors.

Test			Liberty		Notredame		Yosemite	
Train	*D*	*Mean*	*No*	*Yo*	*Li*	*Yo*	*No*	*Li*
polar [9]	175	22.42	24.34	24.34	16.06	16.06	26.85	26.85
cartes	63	35.87	34.06	34.06	34.10	34.10	39.47	39.47
polar + *cartes*	238	25.37	26.16	26.16	20.04	20.04	29.91	29.91
polar + *PCA* [9]	128	8.30	12.09	13.13	5.16	5.41	7.52	6.49
polar [9] + *LW*	128	7.06	8.55	10.48	4.40	3.94	8.86	6.12
cartes + *LW*	63	15.13	17.31	20.34	10.90	11.85	16.84	13.55
polar + *cartes* + *LW*	128	**5.98**	**7.44**	**9.84**	**3.48**	**3.54**	**6.56**	**5.02**

Table 1: Performance comparison on Phototourism dataset between the baseline approach and our combined descriptor. We further show the benefit of learned whitening (LW) over the standard PCA followed by square-rooting. FPR95 is reported for all methods.

Method	Verification	Matching	Retrieval
polar [9]	80.77	32.51	48.04
cartes	70.67	15.79	30.73
polar + *cartes*	77.97	29.34	44.23
polar + *PCA* [9]	87.11	38.45	54.81
polar [9] + *LW*	88.00	41.91	58.80
cartes + *LW*	85.13	33.77	52.94
polar + *cartes* + *LW*	**88.64**	**43.81**	**61.21**

Table 2: Performance comparison of the baseline approach and our combined descriptor via mAP on HPatches dataset. PCA and LW are learned on a subset of HP.

Comparison with the baseline. The results of the experimental evaluation are shown in Tables 1 and 2 for the PT and HP datasets, respectively. For all compared methods, including the baseline, we observed that in the descriptor post-processing stage, the discriminative whitening (marked LW) outperforms PCA followed by square-rooting (originally proposed in [9]). The difference is observed among 4^{th} and 5^{th} row of Tables 1 and 2.

Polar parametrization with the relative gradient direction (*polar*) significantly outperforms the Cartesian parametrization with the absolute gradient direction (*cartes*). After the descriptor post-processing (*polar* + LW vs. *cartes* + LW), the gap is reduced. The performance of the combined descriptor (*polar* + *cartes*) without descriptor post-processing is worse than the baseline descriptor. That is caused by the fact, that the two descriptors are combined with an equal weight, which is clearly suboptimal. No attempt is made to estimate the mixing parameter explicitly, as this is implicitly done in the post-processing stage. The jointly whitened combination of the two parametrizations (last row of Tables 1 and 2) consistently outperforms the baseline method.

Comparison with the State of the Art. We compare the performance of proposed method with previously published results on Phototourism dataset in Table 3. Our method obtains the best performance, while this is achieved with the supervised whitening which is much faster to learn than CNN descriptors. It only takes less than 10 seconds to compute on a modern computer(4 cores, 2.6Ghz) for the *polar* + *cartes* case on the Phototourism Liberty dataset, as opposed to several hours and GPUs for the deep learning approaches.

Test			Liberty		Notredame		Yosemite	
Train	*D*	*Mean*	*No*	*Yo*	*Li*	*Yo*	*No*	*Li*
DC–S2S [35]	512	9.67	8.79	12.84	4.54	5.58	13.02	13.24
DDESC [28]	128	9.85	8.82	**8.82**	4.54	4.54	16.19	16.19
Matchnet [12]	4096	7.75	6.90	10.77	3.87	5.76	8.39	10.88
TF–M [5]	128	6.47	**7.22**	9.79	**3.12**	3.85	7.08	7.82
polar + cartes + LW	128	**5.98**	7.44	9.84	3.48	**3.54**	**5.02**	**6.56**

Table 3: Performance comparison with the state of the art on Phototourism dataset. FPR95 is reported for all methods and the best score per dataset is shown in bold.

Verification		Matching		Retrieval	
TF–R	81.92	**PCW**	**33.69**	*+TF–R*	40.23
+TF–M	82.69	*+TF–M*	34.29	*+SIFT*	40.36
PCW	**82.94**	*+TF–R*	34.37	*+RSIFT*	43.84
+DC–S2S	83.03	*+DDESC*	35.44	*+DDESC*	44.55
+TF–R	83.24	*+RSIFT*	36.77	**PCW**	**48.26**
PCW⋆	**88.64**	**PCW⋆**	**43.81**	**PCW⋆**	**61.21**

Verification		Matching		Retrieval	
DC–S	70.04	*RSIFT*	27.22	*DC–S2S*	34.76
DC–S2S	78.23	*DC–S2S*	27.69	*DC–S*	34.84
DDESC	79.51	*DDESC*	28.05	*TF–R*	37.69
TF–M	81.90	*TF–R*	30.61	*TF–M*	39.40
TF–R	81.92	*TF–M*	32.64	*DDESC*	39.83
PCW	**82.94**	**PCW**	**33.69**	**PCW**	**48.26**

Table 4: Best performing methods on HP dataset. On the left we compare all methods, while on the right only methods that have **not** used any part of HPatches for training. The "+" refers to ZCA used in [6]. Our method is noted by PCW (*polar* + *cartes* + LW) and shown in bold, while training whitening on a subset of HP is denoted by ⋆. Otherwise it is trained on Liberty (PT). Previously top performing methods are DC-S2S [35], DDESC [28], TF [5], and RSIFT [2]. Top 6 methods per task are ranked and shown. Full list of methods in [6].

The comparison on the HPatches dataset is reported in Table 4. On the left all methods are considered, independently whether the splits of HPatches have been used for training or not. The table on the right compares only those methods that have **not** used any part of HPatches for training. In this case, the post-processing (LW) of our method was learned on Phototourism Liberty, as done in [6] so that the numbers are directly comparable. Note that the proposed method trained on Phototourism Liberty scores high even among the methods that used the split of HPatches in training.

6 Conclusions

We have proposed a multiple-kernel local-patch descriptor combining two parametrizations of gradient position and direction. Each parametrization provides robustness to a different type of patch miss-registration: polar parametrization for noise in the dominant orientation, Cartesian for imprecise location of the feature point. Learning a discriminative whitening implicitly sets the relative weight between the two representations. The proposed method consistently outperforms prior methods on two datasets and three tasks.

Acknowledgments The authors were supported by the MSMT LL1303 ERC-CZ grant, Arun Mukundan was supported by the SGS17/185/OHK3/3T/13 grant.

References

[1] Mitsuru Ambai and Yuichi Yoshida. Card: Compact and real-time descriptors. In *ICCV*, 2011.

[2] Relja Arandjelovic and Andrew Zisserman. Three things everyone should know to improve object retrieval. In *CVPR*, 2012.

[3] Artem Babenko and Victor Lempitsky. Aggregating deep convolutional features for image retrieval. In *ICCV*, 2015.

[4] Vassileios Balntas, Edward Johns, Lilian Tang, and Krystian Mikolajczyk. Pn-net: conjoined triple deep network for learning local image descriptors. In *arXiv*, 2016.

[5] Vassileios Balntas, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. Learning local feature descriptors with triplets and shallow convolutional neural networks. In *BMVC*, 2016.

[6] Vassileios Balntas, Karel Lenc, Andrea Vedaldi, and Krystian Mikolajczyk. Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors. In *CVPR*, 2017.

[7] Liefeng Bo and Cristian Sminchisescu. Efficient match kernels between sets of features for visual recognition. In *NIPS*, 2009.

[8] Liefeng Bo, Xiaofeng Ren, and Dieter Fox. Kernel descriptors for visual recognition. In *NIPS*, December 2010.

[9] Andrei Bursuc, Giorgos Tolias, and Hervé Jégou. Kernel local descriptors with implicit rotation matching. In *ICMR*, 2015.

[10] M. Calonder, Vincent Lepetit, C. Strecha, and Pascal Fua. Brief: Binary robust independent elementary features. In *ECCV*, 2010.

[11] Jan-Michael Frahm, Pierre Fite-Georgel, David Gallup, Tim Johnson, Rahul Raguram, Changchang Wu, Yi-Hung Jen, Enrique Dunn, Brian Clipp, Svetlana Lazebnik, et al. Building rome on a cloudless day. In *ECCV*, 2010.

[12] Xufeng Han, Thomas Leung, Yangqing Jia, Rahul Sukthankar, and Alexander C Berg. Matchnet: Unifying feature and metric learning for patch-based matching. In *CVPR*, 2015.

[13] Hervé Jégou and Ondrej Chum. Negative evidences and co-occurrences in image retrieval: The benefit of PCA and whitening. In *ECCV*, 2012.

[14] Takumi Kobayashi and Nobuyuki Otsu. Image feature extraction using gradient local auto-correlations. In *ECCV*, 2008.

[15] Iasonas Kokkinos and Alan Yuille. Scale invariance without scale selection. In *CVPR*, 2008.

[16] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. A sparse texture representation using local affine regions. *IEEE Trans. PAMI*, 27(8):1265–1278, 2005.

[17] David G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[18] Krystian Mikolajczyk and Jiri Matas. Improving descriptors for fast tree matching by optimal linear projection. In *ICCV*, 2007.

[19] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *IEEE Trans. PAMI*, 27(10):1615–1630, 2005.

[20] Dmytro Mishkin, Jiri Matas, Michal Perdoch, and Karel Lenc. Wxbs: Wide baseline stereo generalizations. In *arXiv*, 2015.

[21] Timo Ojala, Matti Pietikainen, and Topi Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Trans. PAMI*, 24(7):971–987, 2002.

[22] Mattis Paulin, Matthijs Douze, Zaid Harchaoui, Julien Mairal, Florent Perronin, and Cordelia Schmid. Local convolutional features with unsupervised training for image retrieval. In *ICCV*, 2015.

[23] James Philbin, Michael Isard, Josef Sivic, and Andrew Zisserman. Descriptor learning for efficient retrieval. In *ECCV*, 2010.

[24] Filip Radenović, Giorgos Tolias, and Ondřej Chum. CNN image retrieval learns from BoW: Unsupervised fine-tuning with hard examples. In *ECCV*, 2016.

[25] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *ICCV*, 2011.

[26] Johannes Lutz Schönberger, Filip Radenović, Ondrej Chum, and Jan-Michael Frahm. From single image query to detailed 3D reconstruction. In *CVPR*, 2015.

[27] Johannes Lutz Schönberger, Hans Hardmeier, Torsten Sattler, and Marc Pollefeys. Comparative evaluation of hand-crafted and learned local features. In *CVPR*, 2017.

[28] Edgar Simo-Serra, Eduard Trulls, Luis Ferraz, Iasonas Kokkinos, Pascal Fua, and Francesc Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *ICCV*, 2015.

[29] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Learning local feature descriptors using convex optimisation. Technical report, Department of Engineering Science, University of Oxford, 2013.

[30] Engin Tola, Vincent Lepetit, and Pascal Fua. Daisy: An efficient dense descriptor applied to wide-baseline stereo. *IEEE Trans. PAMI*, 32(5):815–830, 2010.

[31] Giorgos Tolias, Andrei Bursuc, Teddy Furon, and Hervé Jégou. Rotation and translation covariant match kernels for image retrieval. *CVIU*, 140:9–20, 2015.

[32] Andrea Vedaldi and Andrew Zisserman. Efficient additive kernels via explicit feature maps. In *CVPR*, 2010.

[33] Peng Wang, Jingdong Wang, Gang Zeng, Weiwei Xu, Hongbin Zha, and Shipeng Li. Supervised kernel descriptors for visual recognition. In *CVPR*, 2013.

[34] Simon Winder and Matthew Brown. Learning local image descriptors. In *CVPR*, 2007.

[35] Sergey Zagoruyko and Nikos Komodakis. Learning to compare image patches via convolutional neural networks. In *CVPR*, 2015.

X Understanding and improving kernel local descriptors

Title: Understanding and improving kernel local descriptors
Authors: A. Mukundan, G. Tolias, A. Bursuc, H. Jégou, O. Chum
Published at: IJCV 2019

International Journal of Computer Vision (2019) 127:1723–1737
https://doi.org/10.1007/s11263-018-1137-8

Understanding and Improving Kernel Local Descriptors

Arun Mukundan[1] · Giorgos Tolias[1] · Andrei Bursuc[2] · Hervé Jégou[3] · Ondřej Chum[1]

Received: 21 February 2018 / Accepted: 21 November 2018 / Published online: 3 December 2018

Abstract
We propose a multiple-kernel local-patch descriptor based on efficient match kernels from pixel gradients. It combines two parametrizations of gradient position and direction, each parametrization provides robustness to a different type of patch misregistration: polar parametrization for noise in the patch dominant orientation detection, Cartesian for imprecise location of the feature point. Combined with whitening of the descriptor space, that is learned with or without supervision, the performance is significantly improved. We analyze the effect of the whitening on patch similarity and demonstrate its semantic meaning. Our unsupervised variant is the best performing descriptor constructed without the need of labeled data. Despite the simplicity of the proposed descriptor, it competes well with deep learning approaches on a number of different tasks.

1 Introduction

Representing and matching local features is an essential step of several computer vision tasks. It has attracted a lot of attention in the last decades, when local features still were a required step of most approaches. Despite the large focus on Convolutional Neural Networks (CNN) to process whole images, local features still remain important and necessary for tasks such as Structure-from-Motion (SfM) (Frahm et al. 2010; Heinly et al. 2015; Schonberger and Frahm 2016), stereo matching (Mishkin et al. 2015), or retrieval under severe change in viewpoint or scale (Schönberger et al. 2015; Zhou et al. 2017).

Communicated by Tae-Kyun Kim, Stefanos Zafeiriou, Ben Glocker, Stefan Leutenegger.

Arun Mukundan
arun.mukundan@cmp.felk.cvut.cz

Giorgos Tolias
giorgos.tolias@cmp.felk.cvut.cz

Andrei Bursuc
andrei.bursuc@valeo.com

Hervé Jégou
rvj@fb.com

Ondřej Chum
ondra.chum@cmp.felk.cvut.cz

[1] VRG, FEE, CTU in Prague, Prague, Czech Republic

[2] Valeo.ai, Strašnice, Paris, France

[3] Facebook AI Research, Prague, Paris, France

Classical approaches involve hand-crafted design of a local descriptor, which has been the practice for more than a decade with some widely used examples (Lowe 2004; Bay et al. 2008; Mikolajczyk and Schmid 2005; Tola et al. 2010; Calonder et al. 2010; Leutenegger et al. 2011). Such descriptors do not require any training data or supervision. This kind of approach allows to easily inject domain expertise, prior knowledge or even the result of a thorough analysis (Dong and Soatto 2015). Learning methods have been also employed in order to learn parts of the hand-crafted design, *e.g.* the pooling regions (Winder and Brown 2007; Simonyan et al. 2014), from the training data.

Recently, the focus has shifted from hand-crafted descriptors to CNN-based descriptors. Learning such descriptors relies on large training sets of patches, that are commonly provided as a side-product of SfM (Winder and Brown 2007). Integrating domain expertise has been mostly so far neglected in this kind of approaches. Nevertheless, remarkable performance is achieved on a standard benchmark (Balntas et al. 2016b; Tian and Wu 2017; Mishchuk et al. 2017). On the other hand, recent work (Balntas et al. 2017; Schönberger et al. 2017) shows that many CNN-based approaches do not necessarily generalize equally well on different tasks or different datasets. Hand-crafted descriptors still appear an attractive alternative.

In this work, we choose to work with a particular family of hand-crafted descriptors, the so called *kernel descriptors* (Bo and Sminchisescu 2009; Bo et al. 2011; Tolias et al. 2015). They provide a quite flexible framework for matching sets, patches in our case, by encoding different properties of the set elements, pixels in our case. In particular, we build

upon the hand-crafted kernel descriptor proposed by Bursuc et al. (2015) that is shown to have good performance, even compared to learned alternatives. Its few parameters are easily tuned on a validation set, while it is shown to perform well on multiple tasks, as we confirm in our experiments.

Further post-processing or descriptor normalization, such as Principal Component Analysis (PCA) and power-law normalization, is shown to be effective on different tasks (Delhumeau et al. 2013; Bursuc et al. 2015; Mikolajczyk and Matas 2007; Taira et al. 2016). We combine our descriptor with such post-processing that is learned from the data in unsupervised or supervised ways. We show how to reduce the estimation error and significantly improve results even without any supervision.

The hand-crafted nature and simplicity of our descriptor allows to visualize and analyze its parametrization, and finally understand its advantages and disadvantages. It leads us to propose a simple combination of parametrizations each offering robustness to different types of patch miss-registrations. Interestingly, the same analysis is possible even for the learned post-processing. We observe that its effect on the patch similarity is semantically meaningful. The feasibility of such analysis and visualization is an advantage or our approach, and hand-crafted approaches in general, compared to CNN-based methods. Several insightful ablation and visualization studies (Zeiler and Fergus 2014; Yosinski et al. 2015; Mahendran and Vedaldi 2016; Bau et al. 2017) reveal what a CNN has learned. This typically provides only a partial view, *i.e.* for a small number of neurons, on their behavior, while our approach enables visualization of the overall learned similarity in a general way that is not restricted to particular examples.

This work is an extension of our earlier conference publication (Mukundan et al. 2017). In addition to the earlier version, we propose unsupervised whitening with shrinkage, give extra insight about its effect on patch similarity, present extended comparisons of different whitening variants and provide a proof justifying the absence of regularized concatenation.

The manuscript is organized as follows. Related work is discussed in Sect. 2, and background knowledge for kernel descriptors is presented in Sect. 3. Our descriptor, the different whitening variants, and their interpretation are described in Sect. 4. Finally, the experimental validation on two patch benchmarks is presented in Sect. 5.

2 Related Work

We review prior work on local descriptors, covering both hand-crafted and learned ones.

2.1 Hand-Crafted Descriptors

Hand-crafted descriptors have dominated the research landscape and a variety of approaches and methodologies exists. There are different variants on descriptors building features from filter-bank responses (Bay et al. 2008; Brown et al. 2005; Kokkinos and Yuille 2008; Oliva and Torralba 2001; Schmid and Mohr 1997), pixel gradients (Lowe 2004; Mikolajczyk and Schmid 2005; Tola et al. 2010; Ambai and Yoshida 2011), pixel intensities (Shechtman and Irani 2007; Calonder et al. 2010; Leutenegger et al. 2011; Rublee et al. 2011), ordering or ranking of pixel intensities (Ojala et al. 2002; Heikkila et al. 2009), local edge shape (Forssén and Lowe 2007). Some approaches focus on particular aspects of the local descriptors, such as a injecting invariance in the patch descriptor (Ojala et al. 2002; Lazebnik et al. 2005; Ahonen et al. 2009; Taira et al. 2016), computational efficiency (Tola et al. 2010; Ambai and Yoshida 2011), binary descriptors (Calonder et al. 2010; Leutenegger et al. 2011; Alahi et al. 2012).

A popular direction is that of gradient histogram-based descriptors, where the most popular representative is SIFT p_g (Lowe 2004). SIFT is a long-standing top performer on multiple benchmarks and tasks across the years. Multiple improvements for SIFT have been subsequently proposed: PCA-SIFT (Ke and Sukthankar 2004), ASIFT (Yu and Morel 2009), OpponentSIFT (van de Sande et al. 2010), 3D-SIFT (Scovanner et al. 2007), RootSIFT (Arandjelovic and Zisserman 2012), DSP-SIFT (Dong and Soatto 2015), *etc.* A simple and effective improvement of SIFT is brought by the RootSIFT descriptor (Arandjelovic and Zisserman 2012), which uses Hellinger kernel as similarity measure. DSP-SIFT (Dong and Soatto 2015) counters the aliasing effects caused by the binned quantization in SIFT by pooling gradients over multiple scales instead of only the scale selected by SIFT. Our *kernelized descriptor* also deals with quantization artifacts by embedding each pixel in a continuous space and the aggregating pixels per patch by sum-pooling.

Kernel descriptors based on the idea of Efficient Match Kernels (EMK) (Bo and Sminchisescu 2009) encode entities inside a patch (such a gradient, color, *etc.*) in a continuous domain, rather than as a histogram. The kernels and their few parameters are often hand-picked and tuned on a validation set. Kernel descriptors are commonly represented by a finite-dimensional explicit feature maps (Vedaldi and Zisserman 2012). Quantized descriptors, such as SIFT, can be also interpreted as kernel descriptor (Bursuc et al. 2015; Bo et al. 2010). Furthermore, the widely used RootSIFT descriptor (Arandjelovic and Zisserman 2012) can be also thought of as an explicit feature map from the original SIFT space to the RootSIFT space, such that the Hellinger kernel is *linearised*, *i.e.* the linear kernel (*i.e.* dot product) in RootSIFT space is equivalent to the Hellinger kernel in the original SIFT

space. In this case, the feature mapping is performed by ℓ_1-normalization and square-rooting, without any expansion in dimensionality.

In this work we build upon EMK by integrating multiple pixel attributes in the patch descriptor. Unlike EMK which relies on features from random projections that require subsequent learning, we leverage instead explicit feature maps to approximate a kernel behavior directly. These representations can be further improved by minimal learning.

2.2 Learned Descriptors

Learned descriptors commonly require annotation at patch level. Therefore, research in this direction is facilitated by the release of datasets that originate from an SfM system (Winder and Brown 2007; Paulin et al. 2015). Such training datasets allow effective learning of local descriptors, and in particular, their pooling regions (Winder and Brown 2007; Simonyan et al. 2014), filter banks (Winder and Brown 2007), transformations for dimensionality reduction (Simonyan et al. 2014) or embeddings (Philbin et al. 2010).

Kernelized descriptors are formulated within a supervised framework by Wang et al. (2013), where image labels enable kernel learning and dimensionality reduction. In this work, we rather focus on learning discriminative projections with minimal or no supervision. This is several orders of magnitude faster to learn than other learning approaches.

Recently, local descriptor learning is dominated by deep learning. The proposed network architectures mimic the ones for full-image processing. They have fewer parameters, however they still use a large amount of training patches.

Among representative examples is the work of Simo-Serra et al. (2015) training with hard positive and negative examples or the work of Zagoruyko and Komodakis (2015) where a central-surround representation is found to be immensely beneficial. Such CNN-based approaches are seen as joint feature, filter bank, and metric learning (Han et al. 2015) since both the convolutional filters, patch descriptor and metrics are learned end-to-end. Going further towards an end-to-end pipeline for patch detection and description, LIFT (Yi et al. 2016) advances a multi-step architecture with several spatial transformer modules (Jaderberg et al. 2015) that detects interest points and crops them, identifies their dominant orientation and rotates them accordingly and finally extract a patch descriptor. Paulin et al. (2017) propose a deep patch descriptor from unsupervised learning. They consider a convolutional kernel network (Mairal et al. 2014) with feature maps compatible with the Gaussian kernel and which require layer-wise training.

Recent works in deep patch descriptors lean towards more compact architectures with more carefully designed training strategies and loss functions. Balntas et al. (2016a, b) advance shallower architectures with improved triplet ranking loss (Balntas et al. 2016a, b). In L2-Net (Tian and Wu 2017) supervision is imposed on intermediate feature maps, the loss function integrates multiple attributes, while sampling of training data is done progressively to better balance positive and negative pairs at each step. In HardNet (Mishchuk et al. 2017), extend L2-Net with a loss that mimics Lowe's matching criterion by maximizing the distance between the closest positive and closest negative example in the batch. HardNet is currently a top perfomer on most benchmarks. Despite obtaining impressive results on standard benchmarks, the generalization of CNN-based local descriptors to other datasets is not always the case (Schönberger et al. 2017).

2.3 Post-Processing

A post-processing step is common to both hand-crafted and learned descriptors. This post-processing ranges from simple ℓ_2 normalization, PCA dimensionality reduction, to transformations learned on annotated data (Radenović et al. 2016; Brown et al. 2011; Balntas et al. 2017; Jégou and Chum 2012).

3 Preliminaries

3.1 Kernelized Descriptors

In general lines we follow the formulation of Bursuc et al. (2015). We represent a patch $\mathcal{P}$ as a set of pixels $p \in \mathcal{P}$ and compare two patches $\mathcal{P}$ and $\mathcal{Q}$ via a match kernel

$$\mathcal{M}(\mathcal{P}, \mathcal{Q}) = \sum_{p \in \mathcal{P}} \sum_{q \in \mathcal{Q}} k(p, q), \tag{1}$$

where kernel $k : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is a similarity function, typically non-linear, comparing two pixels or their corresponding feature vectors. The evaluation of this match kernel is costly as it computes exhaustively similarities between all pairs of pixels from the two sets. Match kernel $\mathcal{M}(\mathcal{P}, \mathcal{Q})$ can be approximated with EMK (Bo and Sminchisescu 2009). It uses an explicit feature map $\psi : \mathbb{R}^n \to \mathbb{R}^d$ to approximate this result as

$$\begin{aligned} \mathcal{M}(\mathcal{P}, \mathcal{Q}) &= \sum_{p \in \mathcal{P}} \sum_{q \in \mathcal{Q}} k(p, q) \\ &\approx \sum_{p \in \mathcal{P}} \sum_{q \in \mathcal{Q}} \psi(p)^\top \psi(q) \\ &= \sum_{p \in \mathcal{P}} \psi(p)^\top \sum_{q \in \mathcal{Q}} \psi(q). \end{aligned} \tag{2}$$

Vector $\mathbf{V}(\mathcal{P}) = \sum_{p\in\mathcal{P}} \psi(p)$ is a *kernelized descriptor* (KD), associated with patch $\mathcal{P}$, used to approximate $\mathcal{M}(\mathcal{P}, \mathcal{Q})$, whose explicit evaluation is costly. The approximation is given by a dot product $\mathbf{V}(\mathcal{P})^\top \mathbf{V}(\mathcal{Q})$, where $\mathbf{V}(\mathcal{P}) \in \mathbb{R}^d$. To ensure a unit self similarity, ℓ_2-normalization by a factor γ is introduced. The normalized KD is then given by $\bar{\mathbf{V}}(\mathcal{P}) = \gamma(\mathcal{P})\mathbf{V}(\mathcal{P})$, where $\gamma(\mathcal{P}) = (\mathbf{V}(\mathcal{P})^\top \mathbf{V}(\mathcal{P}))^{-1/2}$.

Kernel k comprises products of kernels, each kernel acting on a different scalar pixel attribute

$$k(p, q) = k_1(p_1, q_1)k_2(p_2, q_2)\ldots k_n(p_n, q_n), \tag{3}$$

where kernel k_n is pairwise similarity function for scalars and p_n are pixel attributes such as position and gradient orientation. Feature map ψ_n corresponds to kernel k_n and feature map ψ is constructed via Kronecker product of individual feature maps $\psi(p) = \psi_1(p_1) \otimes \psi_2(p_2) \otimes \ldots \otimes \psi_n(p_n)$. It is straightforward to show that

$$\psi(p)^\top \psi(q) \approx k_1(p_1, q_1)k_2(p_2, q_2)\ldots k_n(p_n, q_n). \tag{4}$$

3.2 Feature Maps

As non-linear kernel for scalars we use the normalized Von Mises probability density function,[1] which is used for image (Tolias et al. 2015) and patch (Bursuc et al. 2015) representations. It is parametrized by κ controlling the shape of the kernel, where lower κ corresponds to wider kernel, *i.e.* less selective kernel. We use a stationary (shift invariant) kernel that, by definition, depends only on the difference $\Delta_n = p_n - q_n$, *i.e.* $k_{\mathsf{VM}}(p_n, q_n) := k_{\mathsf{VM}}(\Delta_n)$. We approximate this probability density function with Fourier series with N frequencies that produces a feature map $\psi_{\mathsf{VM}} : \mathbb{R} \to \mathbb{R}^{2N+1}$. It has the property that

$$k_{\mathsf{VM}}(p_n, q_n) \approx \psi_{\mathsf{VM}}(p_n)^\top \psi_{\mathsf{VM}}(q_n). \tag{5}$$

In particular we approximate the Fourier series by the sum of the first N terms as

$$k_{\mathsf{VM}}(\Delta_n) \approx \sum_{i=0}^{N} \gamma_i \cos(i\Delta_n). \tag{6}$$

The feature map $\psi_{\mathsf{VM}}(p_n)$ is designed as follows:

$$\psi_{\mathsf{VM}}(p_n) = \Big(\sqrt{\gamma_0}, \sqrt{\gamma_1}\cos(p_n), \ldots, \sqrt{\gamma_N}\cos(Np_n), \sqrt{\gamma_1}\sin(p_n), \ldots, \sqrt{\gamma_N}\sin(Np_n)\Big)^\top. \tag{7}$$

This vector has $2N + 1$ components. It is now easy to show that the inner product of two feature maps is approximating the kernel . That is,

$$\begin{aligned}\psi_{\mathsf{VM}}(p_n)^\top \psi_{\mathsf{VM}}(q_n) &= \gamma_0 + \sum_{i=1}^{N} \gamma_i(\cos(ip_n)\cos(iq_n) \\ &\quad + \sin(ip_n)\sin(iq_n)) \\ &= \sum_{i=0}^{N} \gamma_i \cos(i(p_n - q_n)) \\ &\approx k_{\mathsf{VM}}(\Delta_n).\end{aligned} \tag{8}$$

The reader is encouraged to read prior work for details on these feature maps (Vedaldi and Zisserman 2010; Chum 2015), which are previously used in various contexts (Tolias et al. 2015; Bursuc et al. 2015).

3.3 Descriptor Post-Processing

It is known that further descriptor post-processing (Radenović et al. 2016; Babenko and Lempitsky 2015; Bursuc et al. 2015) is beneficial. In particular, KD is further centered and projected as

$$\hat{\mathbf{V}}(\mathcal{P}) = A^\top(\bar{\mathbf{V}}(\mathcal{P}) - \mu), \tag{9}$$

where $\mu \in \mathbb{R}^d$ and $A \in \mathbb{R}^{d\times d}$ are the mean vector and the projection matrix. These are commonly learned by PCA (Jégou and Chum 2012) or with supervision (Radenović et al. 2016). The final descriptor is always ℓ_2-normalized in the end.

4 Method

In this section we consider different patch parametrizations and kernels that result in different patch similarity. We discuss the benefits of each and propose how to combine them. We further learn descriptor transformation with or without supervision and provide useful insight on how patch similarity is affected.

4.1 Patch Attributes

We consider a pixel p to be associated with coordinates p_x, p_y in Cartesian coordinate system, coordinates p_ρ, p_ϕ in polar coordinate system, pixel gradient magnitude p_m, and pixel gradient angle p_θ. Angles p_θ, $p_\phi \in [0, 2\pi]$, distance from the center p_ρ is normalized to $[0, 1]$, while coordinates p_x, $p_y \in \{1, 2, \ldots, W\}$ for $W \times W$ patches. In order to use feature map ψ_{VM}, attributes p_ρ, p_x, and p_y are linearly mapped to $[0, \pi]$. The gradient angle is expressed *w.r.t.* the patch orientation, *i.e.* p_θ directly, or *w.r.t.* to the position of the pixel. The latter is given as $p_{\tilde{\theta}} = p_\theta - p_\phi$.

[1] Also known as the periodic normal distribution.

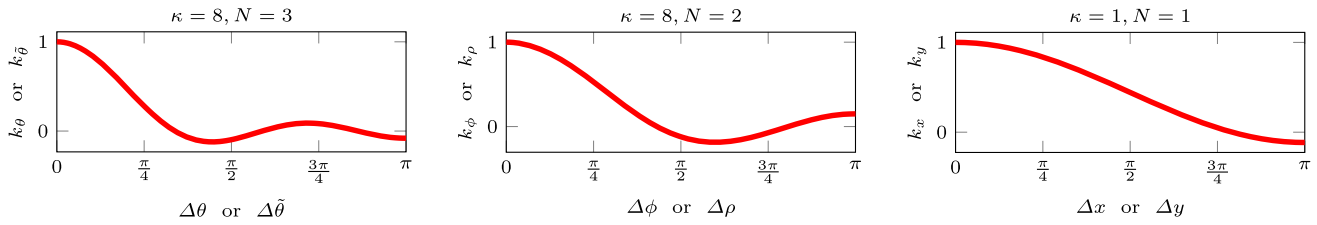


Fig. 1 Kernel approximations that we use for pixel attributes. Parameter κ and the number of frequencies N define the final shape. The choice of kernel parameters is guided by Bursuc et al. (2015)

4.2 Patch Parametrizations

Composing patch kernel k as a product of kernels over different attributes enables easy design of various patch similarities. Correspondingly, this defines different KD. All attributes p_x, p_y, p_ρ, p_θ, p_ϕ, and $p_{\tilde{\theta}}$ are matched by the Von Mises kernel, namely, $k_x, k_y, k_\rho, k_\theta, k_\phi$, and $k_{\tilde{\theta}}$ parameterized by κ_x, κ_y, κ_ρ, κ_θ, κ_ϕ, and $\kappa_{\tilde{\theta}}$, respectively. In a similar manner to SIFT, we apply a Gaussian mask by $p_g = \exp(-p_\rho^2)$ which gives more importance to central pixels.

In this work we focus on the two following match kernels over patches. One in *polar* coordinates

$$\mathcal{M}_{\phi\rho\tilde{\theta}}(\mathcal{P}, \mathcal{Q}) = \sum_{p\in\mathcal{P}} \sum_{q\in\mathcal{Q}} p_g q_g \sqrt{p_m}\sqrt{q_m} k_\phi(p_\phi, q_\phi) \\ k_\rho(p_\rho, q_\rho) k_{\tilde{\theta}}(p_{\tilde{\theta}}, q_{\tilde{\theta}}), \tag{10}$$

and one in *Cartesian* coordinates

$$\mathcal{M}_{xy\theta}(\mathcal{P}, \mathcal{Q}) = \sum_{p\in\mathcal{P}} \sum_{q\in\mathcal{Q}} p_g q_g \sqrt{p_m}\sqrt{q_m} k_x(p_x, q_x) \\ k_y(p_y, q_y) k_\theta(p_\theta, q_\theta). \tag{11}$$

The KD for the two cases are given by

$$\begin{aligned} \mathbf{V}_{\phi\rho\tilde{\theta}}(\mathcal{P}) &= \sum_{p\in\mathcal{P}} p_g \sqrt{p_m} \psi_\phi(p_\phi) \otimes \psi_\rho(p_\rho) \otimes \psi_{\tilde{\theta}}(p_{\tilde{\theta}}) \\ &= \sum_{p\in\mathcal{P}} p_g \sqrt{p_m} \psi_{\phi\rho\tilde{\theta}}(p) \end{aligned} \tag{12}$$

$$\begin{aligned} \mathbf{V}_{xy\theta}(\mathcal{P}) &= \sum_{p\in\mathcal{P}} p_g \sqrt{p_m} \psi_x(p_x) \otimes \psi_y(p_y) \otimes \psi_\theta(p_\theta) \\ &= \sum_{p\in\mathcal{P}} p_g \sqrt{p_m} \psi_{xy\theta}(p). \end{aligned} \tag{13}$$

The $\mathbf{V}_{\phi\rho\tilde{\theta}}$ variant is exactly the one proposed by Bursuc et al. (2015), considered as a baseline in this work. Different parametrizations result in different patch similarity, which is analyzed in the following. In Fig. 1 we present the approximation of kernels used per attribute.

4.3 Post-Processing Learned with or w/o Supervision

We detail different ways to learn the projection matrix A of (9) to perform the descriptor post-processing. Let us consider a learning set of patches $\mathbb{P}$ and the corresponding set of descriptors $V_\mathbb{P} = \{V(\mathcal{P}),\ \mathcal{P} \in \mathbb{P}\}$. Let C be the covariance matrix of $V_\mathbb{P}$. Vector μ is the mean descriptor vector, and different ways to compute A are as follows.

Supervised whitening We further assume that supervision is available in the form of pairs of matching patches. This is given by set $\mathbb{M} = \{(\mathcal{P}, \mathcal{Q}) \in \mathbb{P} \times \mathbb{P},\ \mathcal{P} \sim \mathcal{Q}\}$, where $\sim$ denotes matching patches. We follow the work of Mikolajczyk and Matas (2007) to learn discriminative projections using the available supervision. The discriminative projection is composed of two parts, a whitening part and a rotation part. The whitening part is obtained from the intraclass (matching pairs) covariance matrix $C_\mathbb{M}$, while the rotation part is the PCA of the interclass (non-matching pairs) covariance matrix in the whitened space. We set the interclass one to be equal to C as this is dominated by non-matching pairs, while the intraclass one is given by

$$C_\mathbb{M} = \sum_{(\mathcal{P},\mathcal{Q})\in\mathbb{M}} (V(\mathcal{P}) - V(\mathcal{Q}))\,(V(\mathcal{P}) - V(\mathcal{Q}))^\top. \tag{14}$$

The projection matrix is now given by

$$A = C_\mathbb{M}^{-1/2} \mathrm{eig}(C_\mathbb{M}^{-1/2} C C_\mathbb{M}^{-1/2}), \tag{15}$$

where eig denotes the eigenvectors of a matrix into columns. To reduce the descriptor dimensionality, only eigenvectors corresponding to the largest eigenvalues are used. The same holds for all cases that we perform PCA in the rest of the paper. We refer to this transformation as supervised whitening ($\mathrm{W_S}$).

Unsupervised whitening There is no supervision in this case and the projection is learned via PCA on set $V_\mathbb{P}$. In particular, projection matrix is given by

$$A = \mathrm{eig}(C)\mathrm{diag}(\lambda_1^{-1/2}, \ldots, \lambda_d^{-1/2})^\top, \tag{16}$$

where diag denotes a diagonal matrix with the given elements on its diagonal, and λ_i is the i-th eigenvalue of matrix C. This method is called PCA whitening and we denote simply by Jégou and Chum (2012).

Unsupervised whitening with shrinkage We extend the PCA whitening scheme by introducing parameter t controlling the extent of whitening and the projection matrix becomes

$$A = \text{eig}(C)\text{diag}(\lambda_1^{-t/2}, \ldots, \lambda_d^{-t/2})^\top, \tag{17}$$

where $t \in [0, 1]$, with $t = 1$ corresponding to the standard PCA whitening and $t = 0$ to simple rotation without whitening.

Equivalently, $t = 0$ imposes the covariance matrix to be identity. We call this method attenuated PCA whitening and denote it by W_{UA}.

The aforementioned process resembles covariance estimation with shrinkage (Ledoit and Wolf 2004a, b). The sample covariance matrix is known to be a noise estimator, especially when the available samples are not sufficient relatively to the number of dimensions (Ledoit and Wolf 2004b). Ledoit and Wolf (2004b) propose to replace this by a linear combination of the sample covariance matrix and a structured estimator. Their solution is well conditioned and is shown to reduce the effect of noisy estimation in eigen decomposition. The imposed condition is simply that all variances are the same and all covariances are zero. The shrunk covariance is

$$\tilde{C} = (1 - \beta)C + \beta \mathbf{I}_d, \tag{18}$$

where $\mathbf{I}_d$ is the identity matrix and β the shrinking parameter. This process "shrinks" extreme (too large or too small) eigenvalues to intermediate ones. In our experiments we show that a simple tuning of parameter β performs well across different context and datasets. The projection matrix is now

$$A = \text{eig}(C)\text{diag}((\alpha\lambda_1 + \beta)^{-1/2}, \ldots, (\alpha\lambda_d + \beta)^{-1/2})^\top, \tag{19}$$

where $\alpha = 1 - \beta$. We call this method PCA whitening with shrinkage and denote it by W_{US}. We set parameter β equal to the i-th eigenvalue. A method similar to ours is used in the work of Brown et al. (2011), but does not allow dimensionality reduction since descriptors are projected back to the original space after the eigenvalue clipping.

4.4 Visualizing and Understanding Patch Similarity

We define pixel similarity $\mathcal{M}(p, q)$ as kernel response between pixels p and q, approximated as $\mathcal{M}(p, q) \approx \psi(p)^\top \psi(q)$. To show a spatial distribution of the influence of pixel p, we define a *patch map* of pixel p (fixed p_x, p_y, and p_θ). The patch map has the same size as the image patches; for each pixel q of the patch, map $\mathcal{M}(p, q)$ is evaluated for some constant value of q_θ.

For example, in Fig. 2 patch maps for different kernels are shown. The position of p is denoted by $\times$ symbol. Then, $p_\theta = 0$, while $q_\theta = 0$ for all spatial locations of q in the top row and $q_\theta = -\pi/8$ in the bottom row. This example shows the toy patches and their gradient angles in arrows to be more explanatory. The toy patches are directly defined by p_θ, and q_θ. Only p_θ and q_θ are used in later examples, while the toy patches are skipped from the figures.

The example in Fig. 2 reveals a discontinuity near the center of the patch when pixel similarity is given by $\mathbf{V}_{\phi\rho\tilde{\theta}}$ descriptor. It is caused by the polar coordinate system where a small difference in the position near the origin causes large difference in ϕ and $\tilde{\theta}$. The patch maps reveal weaknesses of kernel descriptors, such the aforementioned discontinuity, but also advantages of each parametrization. It is easy to observe that the kernel parametrized by Cartesian coordinates and absolute angle of the gradient ($\mathbf{V}_{xy\theta}$, third column) is insensitive to small translations, *i.e.* feature point displacement. Moreover, in the bottom row we see that using the relative gradient direction $\tilde{\theta}$ allows to compensate for imprecision caused by small patch rotation, *i.e.* the most similar pixel is not the one at the location of p with different $\tilde{\theta}$, but a rotated pixel with more similar value of $\tilde{\theta}$. This effect is further analyzed in Fig. 3. The final similarity involves the product of two kernels that both depend on angle ϕ. They are both maximized at the same point if $\Delta\theta = 0$, otherwise not. The larger $\Delta\theta$ is, the maximum value moves further (in the patch) from p.

We additionally construct patch maps in the case of descriptor post-processing by a linear transformation, *e.g.* descriptor whitening. Now the contribution of a pixel pair is given by

$$\begin{aligned}\hat{\mathcal{M}}(p, q) &= (A^\top(\psi(p) - \mu))^\top (A^\top(\psi(q) - \mu)) \\ &= (\psi(p) - \mu)^\top AA^\top(\psi(q) - \mu) \\ &= \psi(p)^\top AA^\top \psi(q) - \psi(p)^\top AA^\top \mu \\ &\quad - \psi(q)^\top AA^\top \mu + \mu^\top AA^\top \mu.\end{aligned} \tag{20}$$

The last term is constant and can be ignored. If A is a rotation matrix then the similiarity is affected just by shifting by μ. After the transformation, the similarity is no longer shift-invariant. Non-linear post-processing, such as power-law normalization or simple ℓ_2 normalization cannot be visualized, as it acts after the pixel aggregation.

4.5 Combining Kernel Descriptors

We propose to take advantage of both parametrizations $\mathbf{V}_{\phi\rho\tilde{\theta}}$ and $\mathbf{V}_{xy\theta}$, by summing their contribution. This is performed

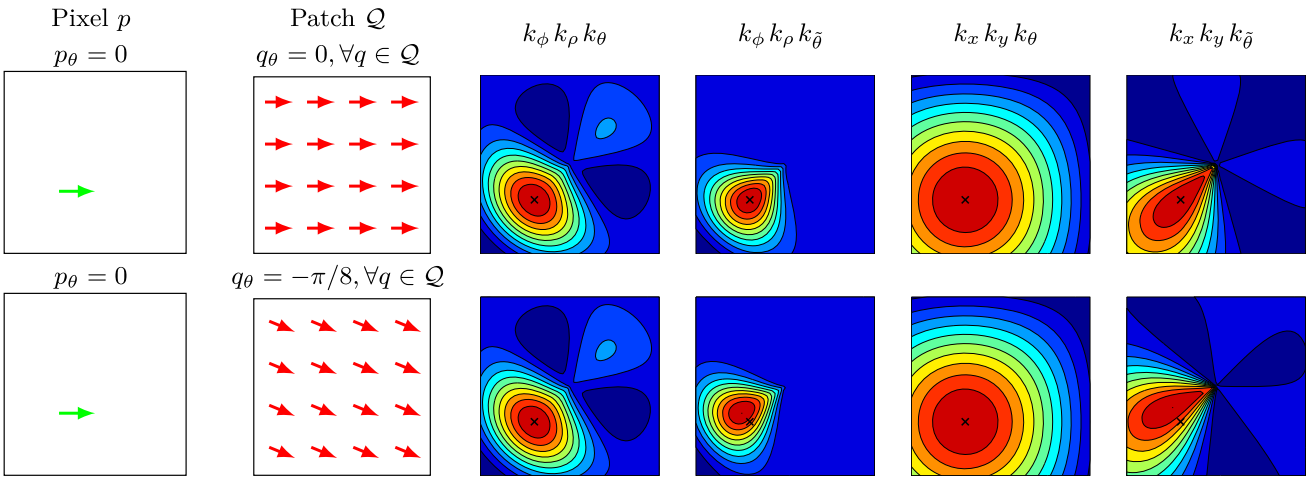


Fig. 2 Patch maps for different parametrizations and kernels. We present two parametrizations in polar and two in cartesian coordinates, with absolute or relative gradient angle for each one. The similarity between each pixel of patch $\mathcal{Q}$ and a single pixel p is shown over patch $\mathcal{Q}$. All pixels in $\mathcal{Q}$ have the same gradient angle, which is shown in red arrows. The position of pixel p is shown with "×" on the patch maps. We show examples for $\Delta\theta$ equal to 0 (top) and $\pi/8$ (bottom). At the top of each column the kernels that are used (patch similarity) are shown. The similarity is shown in a relative manner and, therefore, the absolute scale is missing. Ten isocontours are sampled uniformly and shown in different color (Color figure online)

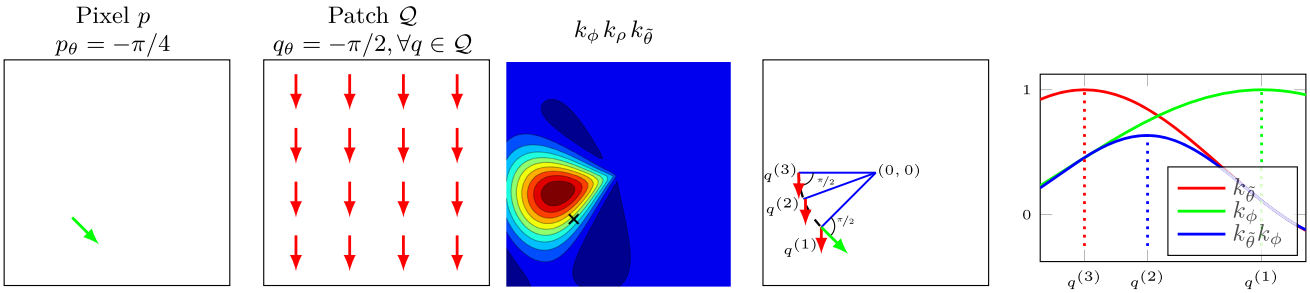


Fig. 3 Patch map with polar parametrization $k_\phi k_\rho k_{\tilde{\theta}}$ for $\Delta\theta = \pi/4$ and the pair of toy pixel and patch on the left. The example explains why the kernel undergoes shifting away from the position of pixel p. The diagram of the 4th column overlays pixel p and 3 pixels of patch $\mathcal{Q}$ with the same distance from the center as p. On the rightmost plot, we illustrate $k_{\tilde{\theta}}$ $(p_{\tilde{\theta}}, q_{\tilde{\theta}})$, k_ϕ (p_ϕ, q_ϕ) for pixels q with $q_\rho = p_\rho$ (on the black dashed circle). $k_{\tilde{\theta}}$ is maximized at $q^{(3)}$, k_ϕ at $q^{(1)}$, and their product at $q^{(2)}$

by simple concatenation of the two descriptors. Finally, whitening is jointly learned and dimensionality reduction is performed.

In Fig. 4 we show patch maps for the individual and combined representation, for different pixels p. Observe how the combined one better behaves around the center. The combined descriptor inherits reasonable behavior around the patch center and insensitivity to position misalignment from the Cartesian parametrization, while insensitivity to dominant orientation misalignment from the polar parametrization, as shown earlier.

4.6 Understanding the Whitened Patch Similarity

We learn the different whitening variants of Sect. 4.3 and visualize their patch maps in Fig. 5. All examples shown are for $\Delta\theta = 0$ but gradient angles p_θ and q_θ jointly vary. We initially observe that the similarity is shift invariant only in the fist column of patch maps where no whitening is applied. This is expected by definition. Projecting by matrix A does not allow to reconstruct the shift invariant kernels anymore; the similarity does not only depend on $\Delta\theta$, which is 0, but also on p_θ and q_θ.

The patch similarity learned by whitening exhibits an interesting property. The shape of the 2D similarity becomes anisotropic and gets aligned with the orientation of the gradient. Equivalently, it becomes perpendicular to the edge on which the pixel lies. This is a semantically meaningful effect. It prevents over-counting of pixel matching along aligned edges of the two patches. In the case of a blob detector this can provide tolerance to errors in the scale estimation, *i.e.* the similarity remains large towards the direc-

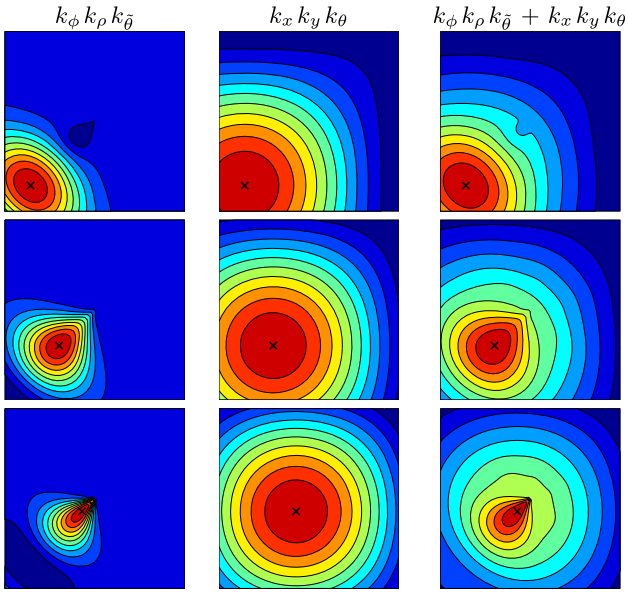


Fig. 4 Patch maps for different pixels and parametrizations and their concatenation. We present two parametrizations in polar and Cartesian coordinates, with relative and absolute gradient angle, respectively. $\Delta\theta$ is fixed to be 0 (individual values of p_θ and q_θ do not matter due to shift invariance) and pixel p is shown with "×". Note the behaviour around the centre in the concatenated case. Ten isocontours are sampled uniformly and shown in different color (Color figure online)

tion that the blob edges shift in case of scale estimation error.

We presume that this is learned by pixels with similar gradient angle that co-occur frequently. A similar effect is captured by both the supervised and the unsupervised whitening with covariance shrinkage, it is, though, less evident in the case of $\mathrm{W_{US}}$. Moreover, we see that it is mostly the Cartesian parametrization that allows this kind of deformation.

According to our interpretation, supervised whitening p_g (Mikolajczyk and Matas 2007; Radenović et al. 2016) owes its success to covariance estimation that is more noise free. The noise removal comes from supervision, but we show that standard approaches for well-conditioned and accurate covariance estimation have similar effect on the patch similarity even without supervision. The observation that different parametrizations allow for different types of co-occurrences to be captured is related to other domains too. For instance. CNN-based image retrieval exhibits improvements after whitening (Babenko and Lempitsky 2015), but this is very unequal between average and max pooling. However, observing the differences is not as easy as in our case with the visualized patch similarity.

Finally, we obtain slices from the 2D patch maps and present the 1D similarity kernels in Fig. 6. It is the similarity between pixel p and all pixels $q \in \mathcal{Q}$ that lie on the vertical and horizontal lines drawn on the patch map at the left of Fig. 6. We present the case for which $p_\theta = 0$ and $q_\theta = 0, \forall q \in \mathcal{Q}$ (top row in Fig. 5). The gradient angle is fully horizontal in this case and the 2D similarity kernel tends to get aligned with that, while the chosen slices are aligned in this fashion too. In our experiments we show that all $\mathrm{W_S}$, $\mathrm{W_{UA}}$, and $\mathrm{W_{US}}$ provide significant performance improvements. However, herein, we observe that the underlined patch similarity demonstrates some differences. Supervised whitening $\mathrm{W_S}$ is not a decreasing function, which might be an outcome of over-fitting to the training data. This is further validated in our experiments. Finally, $\mathrm{W_{UA}}$ and $\mathrm{W_S}$ are not maximized on point p, which does not seem a desired property. $\mathrm{W_{US}}$ is maximized similarly to the raw descriptor without any post-processing.

Patch maps are a way to visualize and study the general shape of the underlined similarity function. In a similar manner, we visualize the kernel responses for a particular pair of patches to reflect which are the pixels contributing the most to the patch similarity. This is achieved by assigning strength $\sum_{q\in\mathcal{Q}} \hat{\mathcal{M}}(p,q)$ to pixel p. In cases without whitening, $\mathcal{M}(p,q)$ is used. We present such heat maps in Fig. 7. Whitening significantly affects the contribution of most pixels. The over-counting phenomenon described in Sect. 4.6 is also visible; some of the long edges are suppressed.

5 Experiments

We evaluate our descriptor on two benchmarks, namely the widely used *Phototourism* (PT) dataset (Winder and Brown 2007), and the recently released *HPatches* (HP) dataset (Balntas et al. 2017). We first show the impact of the shrinkage parameters in unsupervised whitening, and then compare with the baseline method of Bursuc et al. (2015) on top of which we build our descriptor. We examine the generalization properties of whitening when learned on PT but tested on HP, and finally compare against state-of-the-art descriptors on both benchmarks. In all our experiments with descriptor post-processing the dimensionality is reduced to 128, while the combined descriptor original has 238 dimensions, except for the cases where the input descriptor is already of lower dimension. Our experiments are conducted with a Matlab implementation of the descriptor, which takes 5.6 ms per patch for extraction on a single CPU on a 3.5 GHz desktop machine. A GPU implementation reduces time to 0.1 ms per patch on an Nvidia Titan X.

5.1 Datasets and Protocols

The Phototourism dataset contains three sets of patches, namely, Liberty (Li), Notredame (No) and Yosemite (Yo). Additionally, labels are provided to indicate the 3D point that the patch corresponds to, thereby providing supervision.

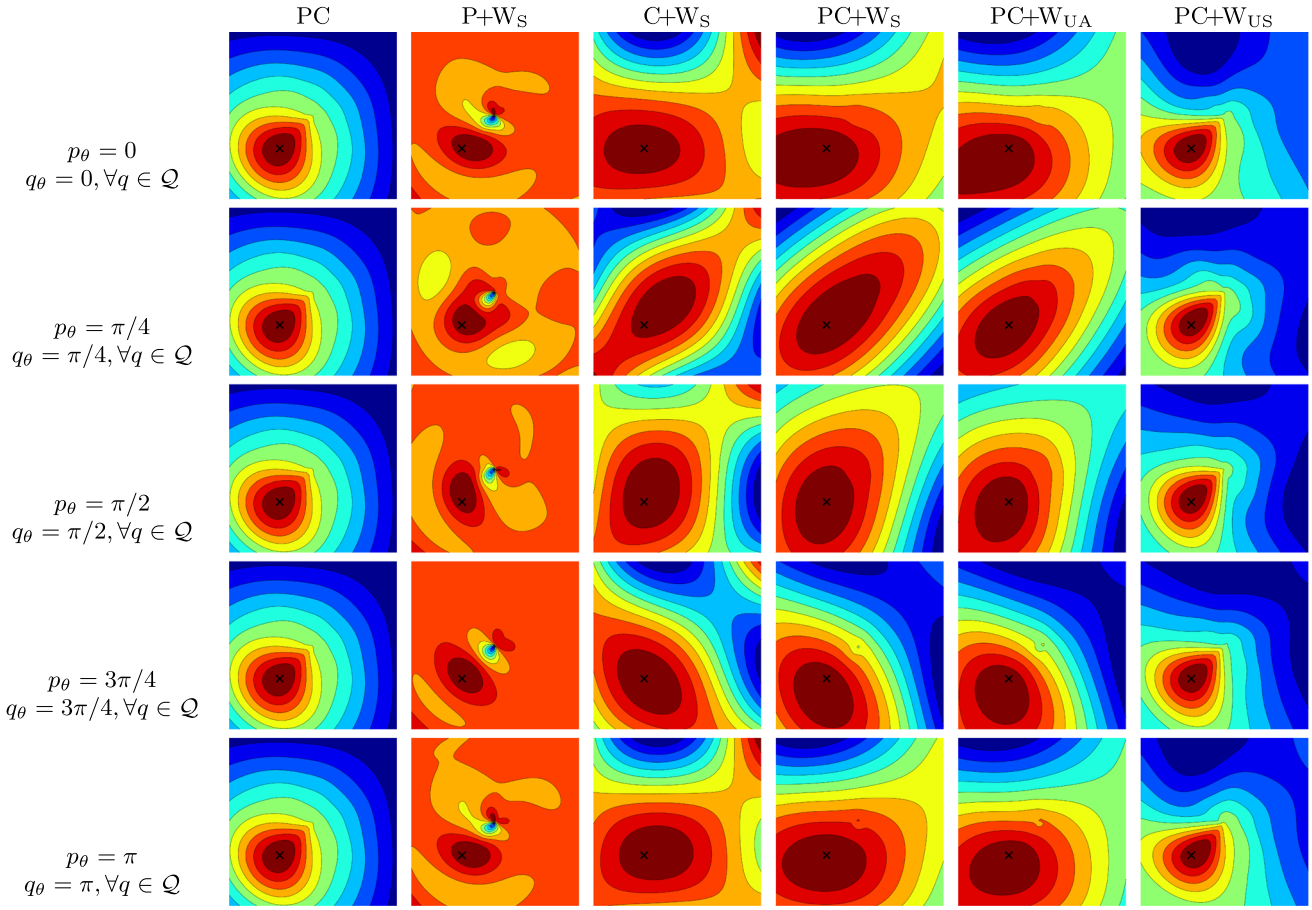


Fig. 5 Patch maps for different parametrizations, their concatenation, different post-processing methods, and varying p_θ and q_θ, while $\Delta\theta$ is always 0. Pixel p is shown with "×". P: polar parametrization, C: cartesian parametrization, W$_{UA}$ is shown for $t = 0.7$ and W$_{US}$ for $\beta = \lambda_{40}$. Whitening is learned on Liberty dataset. Observe that the similarity is no more shift invariant after the whitening and how the shape follows the angle of the gradients. Ten isocontours are sampled uniformly and shown in different color (Color figure online)

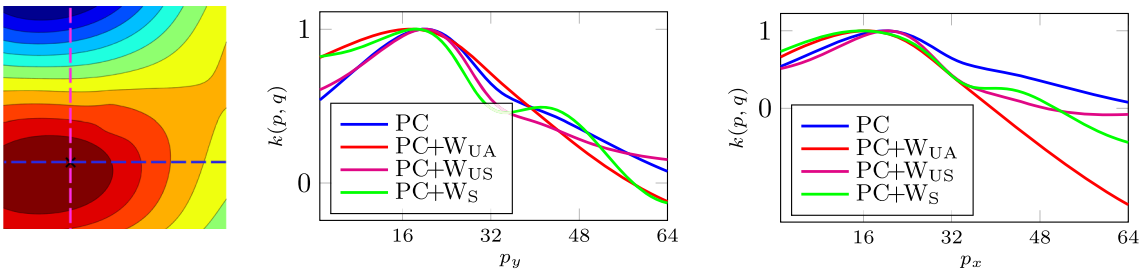


Fig. 6 Visualizing 1D slices of a patch map. Showing similarity $k(p, q)$ for all pixels q with $q_x = p_x$ (middle figure) and $q_y = p_y$ (right figure). It corresponds to 1D similarity across the dashed lines (magenta and blue, respectively) of the patch map on the left. The particular patch map on the left is only chosen as an illustrative example. We show similarity for the first row and for columns 1, 4, 5 and 6 of patch maps from Fig. 5. Pixel p has $p_x = 20$ and $p_y = 20$ with the origin considered at the bottom left corner (Color figure online)

It has been widely used for training and evaluating local descriptors. Performance is measured by the false positive rate at 95% of recall (FPR95). The protocol is to train on one of the three sets and test on the other two. An average over all six combinations is reported.

The HPatches dataset contains local patches of higher diversity, is more realistic, and during evaluation the performance is measured on three tasks: *verification*, *retrieval*, and *matching*. We follow the standard evaluation protocol (Balntas et al. 2017) and report mean Average Precision (mAP).

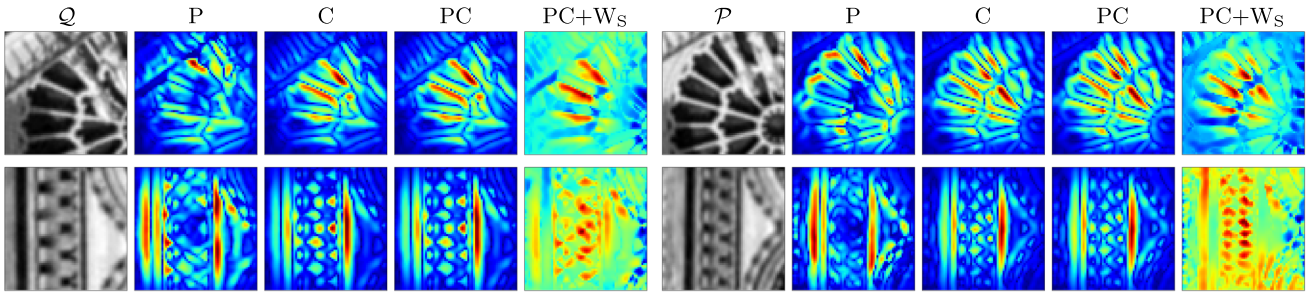


Fig. 7 Positive patch pairs (patches $\mathcal{Q}$ and $\mathcal{P}$) and the corresponding heat maps for polar (P), Cartesian (C), combined (PC), and whitened combined (PC+W$_S$) parametrization. Red (blue) corresponds to maximum (minimum) value. Heat maps on the left side correspond to $\sum_{p\in\mathcal{P}} \mathcal{M}(p,q)$, while the ones on the right side to $\sum_{q\in\mathcal{Q}} \mathcal{M}(p,q)$. In the case of PC+W$_S$, $\hat{\mathcal{M}}(p,q)$ is used instead of $\mathcal{M}(p,q)$ (Color figure online)

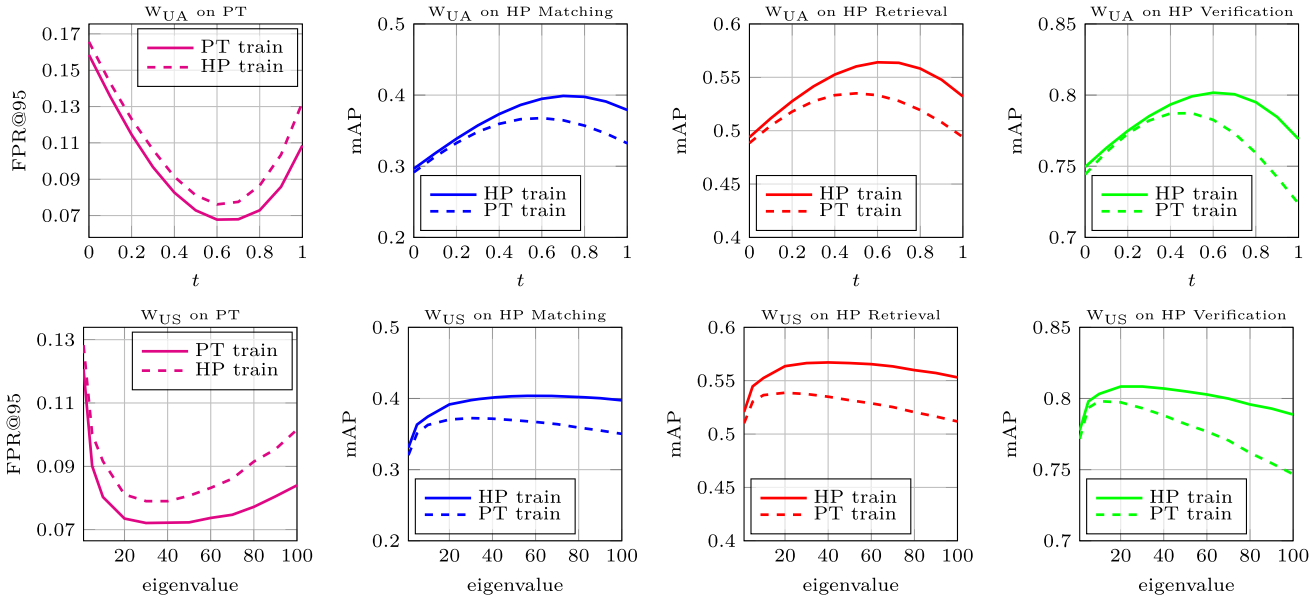


Fig. 8 Impact of the shrinkage parameter for unsupervised whitening when trained on the same or different dataset. We evaluate performance on Photo Tourism and HPatches datasets versus shrinkage parameter t for the attenuated whitening W$_{UA}$ (top row), and versus shrinkage parameter $\beta = \lambda_i$ for whitening with shrinkage W$_{US}$ (bottom row)

We follow the common practice and use models learned on Liberty of PT to compare descriptors that have not used HP during learning. We evaluate on all 3 train/test splits and report the average performance. All reported results on HP (our and other descriptors) are produced by our own evaluation by using the provided framework, and descriptors.[2]

5.2 Impact of the Shrinkage Parameter

We evaluate the impact of the shrinkage parameter involved in the unsupervised whitening. It is t for W$_{UA}$ and $\beta = \lambda_i$ for W$_{US}$. Results are presented in Fig. 8 for evaluation on PT and HP dataset, while the whitening is learned on the same or different dataset. The performance is stable for a range of values, which makes it easy to tune in a robust way across cases and datasets. In the rest of our experiments we set $t = 0.7$ and $\beta = \lambda_{40}$. In Fig. 9 we show the eigenvalues used by W, W$_{UA}$, and W$_{US}$. The contrast between the larger and smaller eigenvalues is decreased.

5.3 Comparison with the Baseline

We compare the combined descriptor against the different parametrizations when used alone. The experimental evaluation is shown in Table 1 for the PT dataset. The baseline is followed by PCA and square-rooting, as originally proposed in Bursuc et al. (2015). We did not consider the square-rooting variant in our analysis in Sect. 4 because

[2] L2Net and HardNet descriptors were provided by the authors of HardNet (Mishchuk et al. 2017).

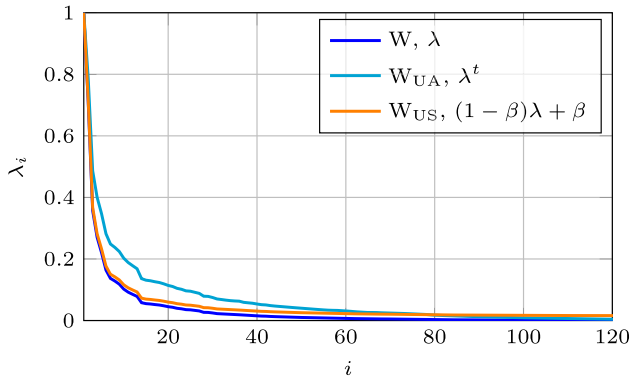


Fig. 9 Eigenvalues for standard PCA whitening, attenuated whitening ($t = 0.7$) and whitening with shrinkage ($\beta = \lambda_{40}$). We normalize so that the maximum eigenvalue is 1. First 120 eigenvalues (out of 238) are shown

such non-linearity does not allow to visualize the underlined patch similarity. Supervised whitening on top of the combined descriptor performs the best. Unsupervised whitening significantly improves too, while it does not require any labeling of the patches.

Polar parametrization with the relative gradient direction (*polar*) significantly outperforms the Cartesian parametrization with the absolute gradient direction (*cartes*). After the descriptor post-processing (*polar* + W_S vs. *cartes* + W_S), the gap is reduced. The performance of the combined descriptor (*polar* + *cartes*) without descriptor post-processing is worse than the baseline descriptor. That is caused by the fact, that the two descriptors are combined with an equal weight, which is clearly suboptimal. No attempt is made to estimate the mixing parameter explicitly. It is implicitly included in the post-processing stage (see Appendix A).

Figure 10 presents patch similarity histograms for matching and non-matching pairs, showing how their separation is improved by the final descriptor.

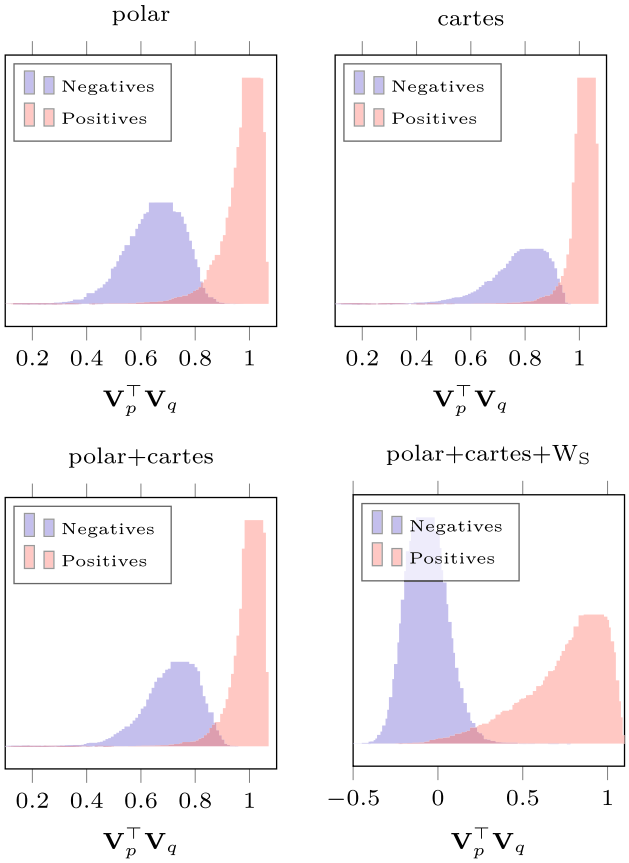


Fig. 10 Histograms of patch similairity for positive and negative patch pairs. Histograms are constructed from 50K matching and 50K non-matching pairs from Notredame dataset

We perform an experiment with synthetic patch transformations to test the robustness of different parametrizations. The whole patch is synthetically rotated or translated by appropriately transforming pixel position and gradient angle in the case of rotation. A fixed amount of rotation/translation

Table 1 Performance comparison on Phototourism dataset between the baseline approach and our combined descriptor. We further show the benefit of learned whitening (W_S) over the standard PCA followed by square-rooting, as well as the other variants that do additional regularization (W_{UA}, W_{US}) without supervision. FPR95 is reported for all methods

Test			Liberty		Notredame		Yosemite	
Train	D	Mean	No	Yo	Li	Yo	Li	No
polar Bursuc et al. (2015)	175	22.42	24.34	24.34	16.06	16.06	26.85	26.85
cartes	63	35.87	34.06	34.06	34.10	34.10	39.47	39.47
polar + *cartes*	238	25.37	26.16	26.16	20.04	20.04	29.91	29.91
polar +PCA+SQRT Bursuc et al. (2015)	128	8.30	12.09	13.13	5.16	5.41	7.52	6.49
polar Bursuc et al. (2015)+W_S	128	7.06	8.55	10.48	4.40	3.94	8.86	6.12
cartes + W_S	63	15.13	17.31	20.34	10.90	11.85	16.84	13.55
polar + *cartes* + W_S	128	**5.94**	**7.46**	**9.85**	**3.45**	**3.55**	6.47	**4.89**
polar + *cartes* + W_{UA}	128	6.79	10.59	11.17	3.80	4.36	**5.58**	5.16
polar + *cartes* + W_{US}	128	7.22	10.61	11.14	4.27	4.46	6.75	6.09

Bold values indicate the best performance

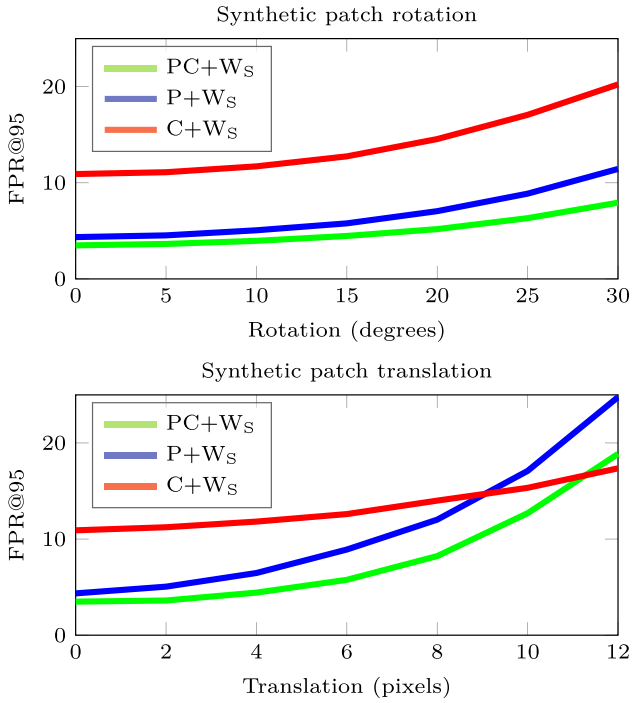


Fig. 11 Performance on PT (training on Liberty , testing on Notredame) when one patch of each pair undergoes synthetic rotation or translation

Table 2 Generalization of different whitening approaches. Mean Average Precision(mAP) for 3 tasks of HP, namely Retrieval (R), Matching (M), and Verification (V). The whitening is learned on PT or HP. We denote supervised by *Sup*

Name	Train	Sup.	R	M	V
polar + *cartes*	*N/A*	*N/A*	45.23	29.68	77.78
polar + *cartes* + W_{UA}	*PT*	*No*	52.78	36.46	77.31
polar + *cartes* + W_{US}	*PT*	*No*	**53.50**	**37.16**	**78.81**
polar + *cartes* + W_S	*PT*	*Yes*	49.66	32.58	75.82
polar + *cartes* + W_{UA}	*HP*	*No*	56.36	39.88	80.06
polar + *cartes* + W_{US}	*HP*	*No*	56.71	40.13	80.70
polar + *cartes* + W_S	*HP*	*Yes*	**61.79**	**44.40**	**83.50**

Bold values indicate the best performance

is performed for one patch of each pair of the PT dataset and results are presented in Fig. 11. It is indeed verified that the Cartesian parametrization is more robust to translations, while the polar one to rotations. The joint one finally partially enjoys the benefits of both.

5.4 Generalization of Whitening

We learn the whitening on PT or HP (supervised and unsupervised) and evaluate the performance on HP. We present results in Table 2. Whitening always improves the performance of the raw descriptor. The unsupervised variant is superior when learning it on an independent dataset. It generalizes better, implying over-fitting of the supervised one (recall the observations of Fig. 6). Learning on HP (the corresponding training part per split) with supervision significantly helps. Note that PT contains only patches detected by DoG, while HP uses a combination of detectors.

Table 3 Performance comparison with the state of the art on Phototourism dataset. We report FPR@95 averaged over 6 dataset combinations for supervised (left) and unsupervised (right) approaches. The whitening for our descriptor is learned on the corresponding training part of PT for each combination

Name	D	$FPR@95$
Supervised		
Brown et al. (2011)	29–36	15.36
Trzcinski et al. (2012)	128	17.08
Simonyan et al. (2014)	73–77	10.38
DC-S2S Zagoruyko and Komodakis (2015)	512	9.67
DDESC Simo-Serra et al. (2015)	128	9.85
Matchnet Han et al. (2015)	4096	7.75
TF-M Balntas et al. (2016b)	128	6.47
L2Net+ Tian and Wu (2017)	128	2.22
HardNet+ Mishchuk et al. (2017)	128	**1.51**
polar + *cartes* + W_S (our)	128	5.98
Unsupervised		
RootSIFT	128	26.14
RootSIFT + PCA + SQRT Bursuc et al. (2015)	80	17.51
polar + PCA + SQRT Bursuc et al. (2015)	128	8.30
polar + *cartes* + W_{UA} (our)	128	**6.79**
polar + *cartes* + W_{US} (our)	128	7.21

Bold values indicate the best performance

5.5 Comparison with the State of the Art

We compare the performance of the proposed descriptor with previously published results on Phototourism dataset. Results are shown in Table 3. Our method obtains the best performance among the unsupervised/hand-crafted approaches by a large margin. Overall, it comes right after the two very recent CNN-based descriptors, namely L2Net (Tian and Wu 2017) and HardNet (Mishchuk et al. 2017). The advantage of our approach is the low cost of the learning. It takes less than a minute; about 45 s to extract descriptors of Liberty and about 10 s to compute the projection matrix. CNN-based competitors require several hours or days of training.

The comparison on the HPatches dataset is reported in Fig. 12. We use the provided descriptors and framework to evaluate all approaches by ourselves. For the descriptors that require learning, the model that is learned on Liberty-PT is used. Our unsupervised descriptor is the top performing hand-crafted variant by a large margin. Overall, it is always outperformed by HardNet, L2Net, while on verification is it

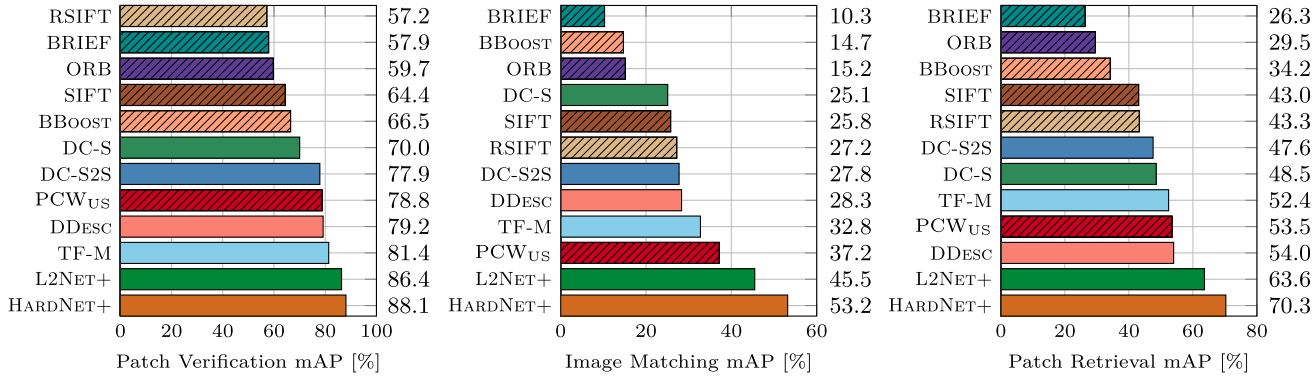


Fig. 12 Performance comparison on HP benchmark. The learning, whenever applicable, is performed on Liberty of PT dataset. Descriptors that do not require any supervision in the form of labeled patches, *i.e.* hand-crafted or unsupervised, are shown in striped bars. Our descriptor is denoted by PCW_{US} (P = *polar*, C = *cartes*)

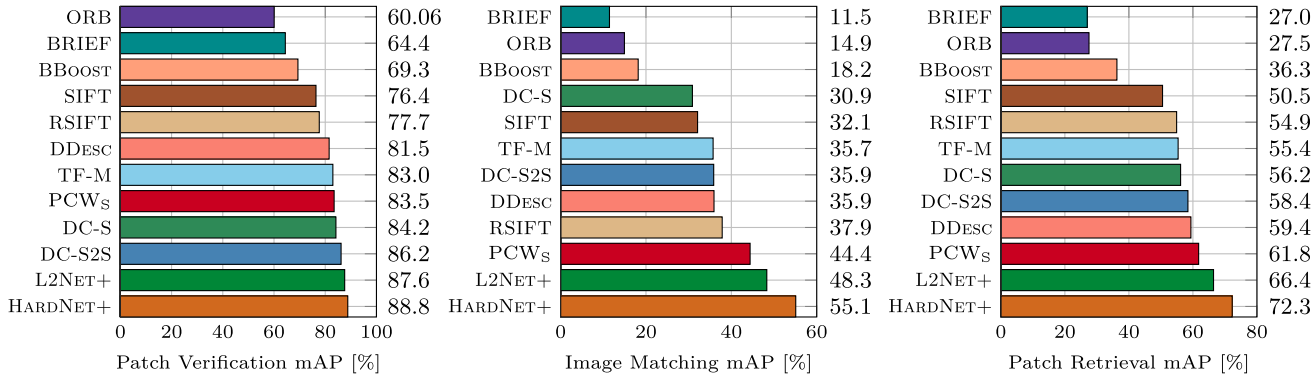


Fig. 13 Performance comparison on HP benchmark when post-processing all descriptors with supervised whitening W_{S} which is learned on HP. The initial learning of the descriptor, whenever applicable, is performed on Liberty of PT dataset. Our descriptor uses the whitening learned on HP and does not use the PT dataset at all. Our descriptor is denoted by PCW_{S} (P = *polar*, C = *cartes*)

additionally outperformed by DDesc and TF-M. Verification is closer to the learning task (loss) involved in the learning of these CNN-based methods.

Finally, we learn supervised whitening W_{S} for all other descriptors, post-process them, and present results in Fig. 13. The projection matrix is learned on HP, in particular the training part of each split. Supervised whitening W_{S} consistently boosts the performance of all descriptors, while this comes at a minimal extra cost compared to the initial training of a CNN descriptor. Our descriptor comes 3rd at 2 out of 3 tasks. Note that it uses the whitening learned on HP (similarly to all other descriptors of this comparison), but does not use the PT dataset at all. All CNN-based descriptors train their parameters on Liberty-PT which is costly, while the overall learning of our descriptor is again in the order of a single minute.

6 Conclusions

We have proposed a multiple-kernel local-patch descriptor combining two parametrizations of gradient position and direction. Each parametrization provides robustness to a different type of patch miss-registration: polar parametrization for noise in the dominant orientation, Cartesian for imprecise location of the feature point. We have performed descriptor whitening and have shown that its effect on patch similarity is semantically meaningful. The lessons learned from analyzing the similarity after whitening can be exploited for further improvements of kernel, or even CNN-based, descriptors. Learning the whitening in a supervised or unsupervised way boosts the performance. Interestingly, the latter generalizes better and sets the best so far performing hand-crafted descriptor that is competing well even with CNN-based descriptors. Unlike the currently best performing CNN-based approaches, the proposed descriptor is easy to implement and interpret.

Acknowledgements The authors were supported by the OP VVV funded Project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics" and MSMT LL1303 ERC-CZ Grant. Arun Mukundan was supported by the CTU student Grant SGS17/185/OHK3/3T/13. We would like to thank Karel Lenc and Vassileios Balntas for their valuable

help with the HPatches benchmark, and Dmytro Mishkin for providing the L2Net and HardNet descriptors for the HPatches dataset.

Appendix: A Regularized Concatenation

When combining the descriptors of different parametrization by concatenation we use both with equal contribution, *i.e.* the final similarity is equal to $k_\phi k_\rho k_{\tilde{\theta}} + k_x k_y k_\theta$. In the case of the raw descriptors this is clearly suboptimal. One would rather regularize by $k_\phi k_\rho k_{\tilde{\theta}} + w k_x k_y k_\theta$ and search for the optimal value of scalar w. We are about to prove that this is not necessary in the case of post-processing by supervised whitening, where the optimal regularization is included in the projection matrix.

We denote a set of descriptors without regularized concatenation by $V_{\mathbb{P}}$ when $w = 1$, while the $V_{\mathbb{P}}^{(w)}$ when $w \neq 1$. It holds that $V_{\mathbb{P}}^{(w)} = \{WV(\mathcal{P}),\ \mathcal{P} \in \mathbb{P}\}$, where W is a diagonal matrix with ones on the dimensions corresponding to the first descriptors (for $k_\phi\ k_\rho\ k_{\tilde{\theta}}$), and has all the rest elements of the diagonal equal to w. The covariance matrix of $V_{\mathbb{P}}$ is C, while of $V_{\mathbb{P}}^{(w)}$ it is $C^{(w)} = WCW^\top$.

Learning the supervised whitening on $V_{\mathbb{P}}$ as in (15) produces projection matrix

$$A = C_{\mathbb{M}}^{-1/2} \mathrm{eig}\left(C_{\mathbb{M}}^{-1/2} C C_{\mathbb{M}}^{-1/2}\right), \tag{21}$$

while learning it on $V_{\mathbb{P}}^{(w)}$ produces projection matrix

$$A^{(w)} = {C_{\mathbb{M}}^{(w)}}^{-1/2} \mathrm{eig}\left({C_{\mathbb{M}}^{(w)}}^{-1/2} C^{(w)} {C_{\mathbb{M}}^{(w)}}^{-1/2}\right). \tag{22}$$

Cholesky decomposition of C gives

$$C = U^\top U = LL^\top, \tag{23}$$

which leads to the Cholesky decomposition

$$C^{(w)} = WU^\top UW^\top = WLL^\top W^\top. \tag{24}$$

Using (24) allows us to rewrite (22) as

$$\begin{aligned} A^{(w)} &= (L^\top W^\top)^{-1} \mathrm{eig}((WU^\top)^{-1} WCW^\top (L^\top W^\top)^{-1}) \\ &= (W^\top)^{-1}(L^\top)^{-1} \mathrm{eig}({U^\top}^{-1} W^{-1} WCW^\top (W^\top)^{-1}(L^\top)^{-1}) \\ &= (W^\top)^{-1}(L^\top)^{-1} \mathrm{eig}({U^\top}^{-1} C (L^\top)^{-1}) \\ &= (W^\top)^{-1}(L^\top)^{-1} \mathrm{eig}(C_{\mathbb{M}}^{-1/2} C C_{\mathbb{M}}^{-1/2}) \\ &= (W^\top)^{-1} A. \end{aligned} \tag{25}$$

Whitening descriptor $V(\mathcal{P}) \in V_{\mathbb{P}}$ with matrix A is performed by

$$\hat{V}(\mathcal{P}) = A^\top (V(\mathcal{P}) - \mu), \tag{26}$$

while whitening descriptor $V(\mathcal{P})^{(w)} \in V_{\mathbb{P}}^{(w)}$ with matrix $A^{(w)}$ is performed by

$$\begin{aligned} \hat{V}(\mathcal{P})^{(w)} &= {A^{(w)}}^\top (WV(\mathcal{P}) - W\mu) \\ &= A^\top W^{-1} (WV(\mathcal{P}) - W\mu) \\ &= \hat{V}(\mathcal{P}). \end{aligned} \tag{27}$$

No matter what the regularization parameter is, the descriptor is identical after whitening. We conclude that there is no need to perform such regularized concatenation.

References

Ahonen, T., Matas, J., He, C., & Pietikäinen, M. (2009). Rotation invariant image description with local binary pattern histogram fourier features. In *Scandinavian conference on image analysis* (pp. 61–70). Berlin.

Alahi, A., Ortiz, R., & Vandergheynst, P. (2012). Reak: fast retina keypoint. In *CVPR*.

Ambai, M., & Yoshida, Y. (2011). Card: Compact and real-time descriptors. In *ICCV*.

Arandjelovic, & R., Zisserman, A., (2012). Three things everyone should know to improve object retrieval. In *CVPR*.

Babenko, A., & Lempitsky, V. (2015). Aggregating deep convolutional features for image retrieval. In *ICCV*.

Balntas, V., Johns, E., Tang, L., & Mikolajczyk, K. (2016). PN-Net: Conjoined triple deep network for learning local image descriptors. arXiv preprint arXiv:1601.05030

Balntas, V., Riba, E., Ponsa, D., & Mikolajczyk, K. (2016). Learning local feature descriptors with triplets and shallow convolutional neural networks. In *BMVC*.

Balntas, V., Lenc, K., Vedaldi, A., & Mikolajczyk, K. (2017). Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors. In *CVPR*.

Bau, D., Zhou, B., Khosla, A., Oliva, A., & Torralba, A. (2017). Networkdissection: Quantifying interpretabilityof deep visual representations. In *CVPR* (pp. 3319–3327). IEEE.

Bay, H., Ess, A., Tuytelaars, T., & Van Gool, L. (2008). Speeded-up robust features (SURF). *CVIU*, *110*(3), 346–359.

Bo, L., Ren, X., & Fox, D. (2010). Kernel descriptors for visual recognition. In *NIPS*.

Bo, L., Ren, X., & Fox, D. (2011). Depth kernel descriptors for object recognition. In *IROS*.

Bo, L., & Sminchisescu, C. (2009). Efficient match kernels between sets of features for visual recognition. In *NIPS*.

Brown, M., Hua, G., & Winder, S. (2011). Discriminative learning of local image descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *33*(1), 43–57.

Brown, M., Szeliski, R., & Winder, S. (2005). Multi-image matching using multi-scale oriented patches. *CVPR*, *1*, 510–517.

Bursuc, A., Tolias, G., & Jégou, H. Kernel. (2015). local descriptors with implicit rotation matching. In *ICMR*.

Calonder, M., Lepetit, V., Strecha, C., & Fua, P. (2010). Brief: Binary robust independent elementary features. In *ECCV*.

Chum, O. (2015). Low dimensional explicit feature maps. In *ICCV*.

Delhumeau, J., Gosselin, P. H., Jégou, H., & Pérez, P. (2013). Revisiting the VLAD image representation. In *ACM multimedia*.

Dong, J., & Soatto, S. (2015). Domain-size pooling in local descriptors: Dsp-sift. In *CVPR*.

Forssén, P.E., & Lowe, D.G. (2007). Shape descriptors for maximally stable extremal regions. In *IEEE 11th international conference on computer vision, 2007. ICCV 2007* (pp. 1–8). IEEE

Frahm, J. M., Fite-Georgel, P., Gallup, D., Johnson, T., Raguram, R., Wu, C., et al. (2010). Building rome on a cloudless day. In *ECCV*.
Han, X., Leung, T., Jia, Y., Sukthankar, R., & Berg, A. C. (2015). Matchnet: Unifying feature and metric learning for patch-based matching. In *CVPR*.
Heikkila, M., Pietikainen, M., & Schmid, C. (2009). Description of interest regions with local binary patterns. *Pattern Recognition*, *42*(3), 425–436.
Heinly, J., Schonberger, J. L., Dunn, E., & Frahm, J. M. (2015). Reconstructing the world* in six days*(as captured by the yahoo 100 million image dataset). In *CVPR*.
Jaderberg, M., Simonyan, K., & Zisserman, A., et al. (2015). Spatial transformer networks. In*NIPS* (pp. 2017–2025)
Jégou, H., & Chum, O. (2012). Negative evidences and co-occurrences in image retrieval: The benefit of PCA and whitening. In *ECCV*.
Ke, Y., & Sukthankar, R. (2004). PCA-SIFT: a more distinctive representation for local image descriptors. In *CVPR* (pp. 506–513).
Kokkinos, I., & Yuille, A. (2008). Scale invariance without scale selection. In *CVPR*.
Lazebnik, S., Schmid, C., & Ponce, J. (2005). A sparse texture representation using local affine regions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*(8), 1265–1278.
Ledoit, O., & Wolf, M. (2004). Honey, i shrunk the sample covariance matrix. *The Journal of Portfolio Management*, *30*(4), 110–119.
Ledoit, O., & Wolf, M. (2004). A well-conditioned estimator for large-dimensional covariance matrices. *Journal of Multivariate Analysis*, *88*(2), 365–411.
Leutenegger, S., Chli, M., & Siegwart, R. Y. Brisk. (2011). Binary robust invariant scalable keypoints. In *ICCV*.
Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *IJCV*, *60*(2), 91–110.
Mahendran, A., & Vedaldi, A. (2016). Visualizing deep convolutional neural networks using natural pre-images. *IJCV*, *120*(3), 233–255.
Mairal, J., Koniusz, P., Harchaoui, Z., & Schmid, C. (2014). Convolutional kernel networks. In *NIPS* (pp. 2627–2635).
Mikolajczyk, K., & Matas, J. (2007). Improving descriptors for fast tree matching by optimal linear projection. In *ICCV*.
Mikolajczyk, K., & Schmid, C. (2005). A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*(10), 1615–1630.
Mishchuk, A., Mishkin, D., Radenovic, F., & Matas, J. (2017). Working hard to know your neighbor's margins: Local descriptor learning loss. In *NIPS*.
Mishkin, D., Matas, J., Perdoch, M., & Lenc, K. (2015). WxBS: Wide baseline stereo generalizations. arXiv preprint arXiv:1504.06603
Mukundan, A., Tolias, G., & Chum, O. (2017). Multiple-kernel local-patch descriptor. In *BMVC*.
Ojala, T., Pietikainen, M., & Maenpaa, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *24*(7), 971–987.
Oliva, A., & Torralba, A. (2001). Modeling the shape of the scene: A holistic representation of the spatial envelope. *IJCV*, *42*(3), 145–175.
Paulin, M., Douze, M., Harchaoui, Z., Mairal, J., Perronin, F., & Schmid, C. (2015). Local convolutional features with unsupervised training for image retrieval. In *ICCV*.
Paulin, M., Mairal, J., Douze, M., Harchaoui, Z., Perronnin, F., & Schmid, C. (2017). Convolutional patch representations for image retrieval: An unsupervised approach. *ICCV*, *121*(1), 149–168.
Philbin, J., Isard, M., Sivic, J., & Zisserman, A. (2010). Descriptor learning for efficient retrieval. In *ECCV*.
Radenović, F., Tolias, G., & Chum, O. (2016). CNN image retrieval learns from BoW: Unsupervised fine-tuning with hard examples. In *ECCV*.
Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. (2011). Orb: An efficient alternative to sift or surf. In *ICCV*.
Schmid, C., & Mohr, R. (1997). Local grayvalue invariants for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *19*(5), 530–535.
Schonberger, J. L., & Frahm, J. M. (2016). Structure-from-motion revisited. In *CVPR*.
Schönberger, J. L., Hardmeier, H., Sattler, T., & Pollefeys, M. (2017). Comparative evaluation of hand-crafted and learned local features. In *CVPR*.
Schönberger, J. L., Radenović, F., Chum, O., & Frahm, J. M. (2015). From single image query to detailed 3D reconstruction. In *CVPR*.
Scovanner, P., Ali, S., & Shah, M. (2007). A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th ACM international conference on multimedia* (pp. 357–360).
Shechtman, E., & Irani, M. (2007). Matching local self-similarities across images and videos. In *CVPR* (p. (pp. 1–8). IEEE.
Simo-Serra, E., Trulls, E., Ferraz, L., Kokkinos, I., Fua, P., & Moreno-Noguer, F. (2015). Discriminative learning of deep convolutional feature point descriptors. In *ICCV*.
Simonyan, K., Vedaldi, A., & Zisserman, A. (2014). Learning local feature descriptors using convex optimisation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *36*(8), 1573–1585.
Taira, H., Torii, A., & Okutomi, M. (2016). Robust feature matching by learning descriptor covariance with viewpoint synthesis. In *ICPR*.
Tian, B. F. Y., & Wu, F. (2017). L2-net: Deep learning of discriminative patch descriptor in euclidean space. In *CVPR*.
Tola, E., Lepetit, V., & Fua, P. (2010). Daisy: An efficient dense descriptor applied to wide-baseline stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *32*(5), 815–830.
Tolias, G., Bursuc, A., Furon, T., & Jégou, H. (2015). Rotation and translation covariant match kernels for image retrieval. *CVIU*, *140*, 9–20.
Trzcinski, T., Christoudias, M., Lepetit, V., & Fua, P. (2012). Learning image descriptors with the boosting-trick. In *NIPS*
van de Sande, K. E. A., Gevers, T., & Snoek, C. G. M. (2010). Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *32*(9), 1582–1596.
Vedaldi, A., & Zisserman, A. (2010). Efficient additive kernels via explicit feature maps. In *CVPR*.
Vedaldi, A., & Zisserman, A. (2012). Efficient additive kernels via explicit feature maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *34*, 480–492.
Wang, P., Wang, J., Zeng, G., Xu, W., Zha, H., & Li, S. (2013). Supervised kernel descriptors for visual recognition. In *CVPR*.
Winder, S., & Brown, M. (2007). Learning local image descriptors. In *CVPR*.
Yi, K. M., Trulls, E., Lepetit, V., & Fua, P. (2016). Lift: Learned invariant feature transform. In *ECCV* (pp. 467–483). Springer.
Yosinski, J., Clune, J., Nguyen, A., Fuchs, T., & Lipson, H. (2015). Understanding neural networks through deep visualization. arXiv preprint arXiv:1506.06579
Yu, G., & Morel, J. M. (2009). A fully affine invariant image comparison method. In *ICASSP*. (pp. 1597–1600). IEEE.
Zagoruyko, S., & Komodakis, N. (2015). Learning to compare image patches via convolutional neural networks. In *CVPR*.
Zeiler, M. D., & Fergus, R. (2014). Visualizing and understanding convolutional networks. In *ECCV*.
Zhou, L., Zhu, S., Shen, T., Wang, J., Fang, T., & Quan, L. (2017). Progressive large scale-invariant image matching in scale space. In *ICCV*.

XI Explicit Spatial Encoding for Deep Local Descriptors

Title: Explicit Spatial Encoding for Deep Local Descriptors
Authors: A. Mukundan, G. Tolias, O. Chum
Published at: CVPR 2019

Explicit Spatial Encoding for Deep Local Descriptors

Arun Mukundan Giorgos Tolias Ondřej Chum
Visual Recognition Group, FEE, CTU in Prague

Abstract

We propose a kernelized deep local-patch descriptor based on efficient match kernels of neural network activations. Response of each receptive field is encoded together with its spatial location using explicit feature maps. Two location parametrizations, Cartesian and polar, are used to provide robustness to a different types of canonical patch misalignment. Additionally, we analyze how the conventional architecture, i.e. *a fully connected layer attached after the convolutional part, encodes responses in a spatially variant way. In contrary, explicit spatial encoding is used in our descriptor, whose potential applications are not limited to local-patches. We evaluate the descriptor on standard benchmarks. Both versions, encoding* 32×32 *or* 64×64 *patches, consistently outperform all other methods on all benchmarks. The number of parameters of the model is independent of the input patch resolution.*

1. Introduction

Local feature extraction and representation is still an essential part of a number computer vision applications across many different problems. A common and well performing procedure is a sequence of three steps: local feature detection [16, 27, 29, 6, 25], local patch rectification into a canonical form, and finally a descriptor construction from the canonical patch [28, 6, 25, 14]. The desired property of the local patch descriptors is that Euclidean distance or a dot product between two descriptors indicates whether they are matching, *i.e*. the local features are coming approximately from the same surface of a 3D scene. The descriptor methods have shifted from hand-crafted to currently the most successful convolutional neural network (CNN) based approaches [58, 46, 5, 30, 49, 33].

Fully convolutional neural networks takes an image or a patch as input and produces a tensor, where a vector at each spatial location can be seen as a detector response over its receptive field. In the case of variable-sized, or non-aligned input, such as images, the response tensor is transformed into a descriptor typically by some form of global pooling [40, 19, 39], which discards geometric information. The global pooling is analogous to bags-of-features [13, 48] or descriptor aggregation [37, 18]. In the case of aligned input of fixed size, such as rectified image patches, the tensor is vectorized and further processed. Vectorization has similar interpretation to vectorizing spatial bins in SIFT [25]. Commonly, the vectorized tensor is processed by a single fully-connected (FC) layer [30, 49], that can be either interpreted as learned affine (linear and bias) transformation of the space, *e.g*. whitening and dimensionality reduction, or as spatially dependent embedding with efficient match kernels (EMK) [9, 11] (see Section 3.2). The key contribution of this work is a CNN module that explicitly models the spatial information of a rectified patch. Its applicability is not limited to local descriptors.

Two rectified patches coming from matching local features are far from being identical in general. The difference has two sources, namely appearance change in imaging process and geometric misalignment. The former comes from different light conditions, non-planarity of the surface, imaging artifacts, *etc*. The latter is caused by the detected feature covering slightly different area of the 3D surface, or incorrect rectification of the patch. These are consequences of either the appearance changes or of insufficient geometric invariance of the detector, *i.e*. affine invariant detector acting on projectively transformed surface.

Prior work on hand-crafted feature descriptors has shown that it is beneficial to explicitly address the geometric misalignment. Some of the approaches handling this are soft assignment of gradients to bins in SIFT and continuous spatial encoding by kernel methods in different [11] or multiple [32] coordinate systems.

CNNs are powerful in modeling the appearance variance, while weak in modeling the geometric displacement (at least with a single FC layer). Recent methods propose different ways of incorporating spatial information in a CNN [34, 24], but their application field is different than local descriptors. In this work, we propose to model the geometric misalignment by efficient match kernels that explicitly encode the spatial positions of the responses. To encode the spatial information, kernel-based explicit feature maps are used in a similar fashion to hand-crafted features [11, 32]. This can be seen as a transition from soft

binning, *i.e*. overlapping receptive fields, to continuous efficient match kernels. In contrast to models with an FC layer, with efficient match kernels the number of model parameters does not grow with increased resolution of the input patch, *i.e*. the models for 32×32 patch input has the same number of parameters as the model for 64×64. The applications of the proposed descriptor go beyond that of local-patches, *e.g*. tasks where encoding spatial position is essential [24, 34].

The rest of the paper is organized as follows. The related work is discussed in Section 2. Conventional deep local descriptors and the proposed ones is discussed in Section 3. Implementation details are detailed in Section 4. Finally, we present and discuss our experiments on standard benchmarks in Section 5.

2. Related work

In this section, we review prior work related to hand-crafted and learned descriptors of local features.

2.1. Hand-crafted descriptors

There are numerous approaches to hand-craft local descriptors. The variants are based on different types of processing of the input patch, such as filter-bank responses [6, 10, 22, 36, 43], pixel gradients [25, 28, 50, 1], pixel intensities [45, 12, 23, 41] and ordering or ranking of pixel intensities [35, 17]. The most prominent direction has been that of gradient histograms, an approach followed also by the most popular hand-crafted local descriptor, namely SIFT [25]. Several improvements and extensions exist in the literature [20, 57, 21, 44, 2, 14]. The RootSIFT [2] variant efficiently estimates Hellinger distance and became a standard choice in approaches and tasks.

Kernel descriptors are derived from the concept of efficient match kernels [9] and form a flexible way to design descriptors with the desired invariant properties. Kernel descriptors have been proposed not only for local patches [11] but also as a global image descriptor [8, 7]. The kernel descriptor of Bursuc *et al*. [11] was shown to outperform learned descriptors at that time.

2.2. Learning descriptors

Structure-from-Motion and datasets such as Photo-Tourism [55] gave rise to learned local descriptors. The learned part varies from their pooling regions [55, 47] and filter banks [55] to transformations for dimensionality reduction [47] and embeddings [38].

Learning is also applied to kernelized descriptors as in the supervised framework by Wang *et al*. [54]. The local descriptors in their case are not used separately but directly aggregated into a global image representation, while supervision comes at image level. Kernel local descriptors are combined with supervised learning in the form of discriminative projections in the work of Mukundan *et al*. [32]. Our work is inspired by theirs; we use the same kernel-based position encoding, but on top of convoltutional activations instead of pixel attributes.

2.3. Deep learning of descriptors

The interest in local descriptor learning is lately dominated by deep learning [46, 58, 15, 56, 5, 3]. All examples in the literature use architectures that consist of a sequence of common CNN layers, similar to the ones of generic computer vision tasks, such as object recognition, but less deep and with fewer parameters in total. They typically require a large amount of training data in the form of local patch pairs or triplets. Some of the contributions are about mining hard training samples [46, 30, 26], different loss functions [5], different architectures [49] or training jointly with the local feature detector [56].

Two of the most recent and successful deep local descriptors are L2-Net [49] and HardNet [30]. L2-Net applies the loss function to intermediate feature maps too and the loss function integrates multiple attributes. HardNet extends L2-Net by sampling the hardest within batch samples and currently constitutes the state-of-the-art descriptor. Their common characteristic, which is shared among all ancestors, is that they are using common CNN layers in their architecture. As a consequence, spatial information of convolutional feature maps is not explicitly encoded, but only processed with a standard FC layer.

3. Method

We initially present the current typical architecture for deep local descriptors in the literature. Then, we provide a different perspective that formulates such descriptors as match kernels. It allows us to point out how the encoding of convolutional feature maps is performed in a translation variant way, but without explicitly encoding the spatial information. Finally, we present our novel deep local descriptor which is derived through the same match kernel framework and improves exactly this drawback. We get inspired by hand-crafted kernel descriptors to incorporate explicit position encoding into deep networks for local descriptors. An overview of the proposed descriptor is shown in Figure 1,

3.1. Deep local descriptors

Conventional architectures for deep local descriptors consist of a sequence of convolutional layers, producing translation invariant feature maps, and a final FC layer. We denote the descriptor extraction process by function $\psi : \mathbb{R}^{N \times N} \rightarrow \mathbb{R}^D$, where N is the size of the input patch and D the dimensionality of the final descriptor. Descriptor

for patch $a \in \mathbb{R}^{N\times N}$ is given by $\psi(a) \in \mathbb{R}^D$ or equivalently $\boldsymbol{\psi}_a$ to simplify the notation.

We denote the convolutional part of the network, *i.e.* a *Fully Convolutional Network* (FCN), by function $\phi : \mathbb{R}^{N\times N} \to \mathbb{R}^{n\times n\times d}$. Size n of the resulting feature map is related to input size N and the architecture of the network. Feature map $\phi(a)$, equivalently denoted by $\boldsymbol{\phi}_a$, is a 3D tensor of activations, which we also view as a 2D grid of d-dimensional vectors. We call these vectors *convolutional descriptors* and use $\boldsymbol{\phi}_a^p$ to denote the vector with coordinates $p = (i, j)$ on the $n \times n$ grid, *i.e.* $p \in [n]^2$ [1]. Each convolutional descriptor corresponds to a region of the input patch a that is equal to the receptive field size of the feature map.

The standard practice is to vectorize 3D tensor $\boldsymbol{\phi}_a$ and feed it to an FC layer with parameters that consist of matrix $W \in \mathbb{R}^{D\times(n\times n\times d)}$ and bias $\mathbf{w} \in \mathbb{R}^D$. The final descriptor is constructed as

$$\boldsymbol{\psi}_a = W \operatorname{vec}(\boldsymbol{\phi}_a) + \mathbf{w}, \tag{1}$$

where vec denotes tensor vectorization. A local descriptor is typically ℓ_2-normalized, which is equivalently achieved by introducing a normalization factor $\gamma_a = 1/\sqrt{\boldsymbol{\psi}_a^\top \boldsymbol{\psi}_a}$ producing descriptor $\hat{\boldsymbol{\psi}}_a = \gamma_a \boldsymbol{\psi}_a$.

Similarity (or distance) between patches a and b is estimated with inner product (or Euclidean distance) $\hat{\boldsymbol{\psi}}_a^\top \hat{\boldsymbol{\psi}}_b$. The ℓ_2-normalized descriptor is always used to compare patches, but we often use $\boldsymbol{\psi}_a$ (and not $\hat{\boldsymbol{\psi}}_a$) simply to specify which descriptor variant is used. Several deep local descriptors in the recent literature, namely L2Net [49], HardNet [30], and GeoDesc [26] follow such an architecture and can be formulated in the same way.

3.2. A match-kernel perspective

We provide an alternative, but equivalent, construction of deep local descriptors. We consider matrix W as a concatenation of n^2 matrices, *i.e.*

$$W = \begin{pmatrix} W_{(1,1)}^\top \\ \vdots \\ W_{(i,j)}^\top \\ \vdots \\ W_{(n,n)}^\top \end{pmatrix}^\top, \tag{2}$$

where $W_p \in \mathbb{R}^{D\times d}$. Descriptor in (1) can be now written as

$$\boldsymbol{\psi}_a = \sum_{p\in[n]^2} W_p \boldsymbol{\phi}_a^p + \mathbf{w}', \tag{3}$$

where $\mathbf{w}' = \mathbf{w}/n^2$. Moreover, patch similarity becomes

$$\begin{aligned} \hat{\boldsymbol{\psi}}_a^\top \hat{\boldsymbol{\psi}}_b &\propto \sum_{p,q\in[n]^2} \left(W_p \boldsymbol{\phi}_a^p + \mathbf{w}'\right)^\top \left(W_q \boldsymbol{\phi}_b^q + \mathbf{w}'\right) \\ &= \sum_{p,q\in[n]^2} g_{fc}(\boldsymbol{\phi}_a^p, p)^\top g_{fc}(\boldsymbol{\phi}_b^q, q), \end{aligned} \tag{4}$$

where $g_{fc} : \mathbb{R}^d \times [n]^2 \to \mathbb{R}^D$ is a function that encodes a convolutional descriptor in a translation variant way, depending on its position in the $n \times n$ grid. The match kernel formulation in (4) interprets deep local descriptor similarity as similarity accumulation for all pairs of positions on the $n \times n$ grid. It reveals that matching between convolutional descriptors in $\boldsymbol{\phi}_a$ and $\boldsymbol{\phi}_b$ is performed in a translation variant way. The *encoding function* g in the case of conventional deep local descriptors is

$$g_{fc}(\mathbf{v}, p) = W_p \mathbf{v} + \mathbf{w}', \tag{5}$$

where matrix W_p and $\mathbf{w}'$ come from the parameters of the FC layer. In this work, we propose a new encoding function g, not restricted to standard CNN architecture (layers), that explicitly encodes position p on the 2D grid.

3.3. Position encoding

Explicit feature maps [53] are used to encode the position. Let $f : \mathbb{R} \to \mathbb{R}^{2s+1}$ be a feature map, where s is a design choice defining the dimensionality of the embedding. Such a feature map defines a shift invariant kernel $K : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ with kernel signature k, so that $K(\alpha, \beta) = k(\alpha - \beta)$

$$f(\alpha)^\top f(\beta) = K(\alpha, \beta) = k(\alpha - \beta). \tag{6}$$

The kernel K (or the feature map f) is constructed to approximate the Von Mises kernel [51].

We propose encoding function $g_{xy} : \mathbb{R}^d \times [n]^2 \to \mathbb{R}^{D(2s+1)^2}$ given by

$$g_{xy}(\boldsymbol{\phi}_a^p, p) = \boldsymbol{\phi}_a^p \otimes f(x_p) \otimes f(y_p), \tag{7}$$

where $\otimes$ is the Kronecker product and x_p and y_p provide the coordinates of position p in a Cartesian coordinate system [2]. It is a joint encoding of the convolutional descriptor and the explicit representation of its position. It is inspired by the work of Mukundan *et al.* [32] who propose a hand-crafted local descriptor that encodes pixel gradients with their positions in the patch. Similarity of two such encodings is given by

$$g_{xy}(\boldsymbol{\phi}_a^p, p)^\top g_{xy}(\boldsymbol{\phi}_b^q, q) = \boldsymbol{\phi}_a^{p\top} \boldsymbol{\phi}_b^q \cdot k(x_p - x_q) \cdot k(y_p - y_q). \tag{8}$$

[1] $[i] = \{1 \dots i\}$ and $[i]^2 = [i] \times [i]$

[2] For $p = (i, j)$, $x_p = i$ and $y_p = j$.

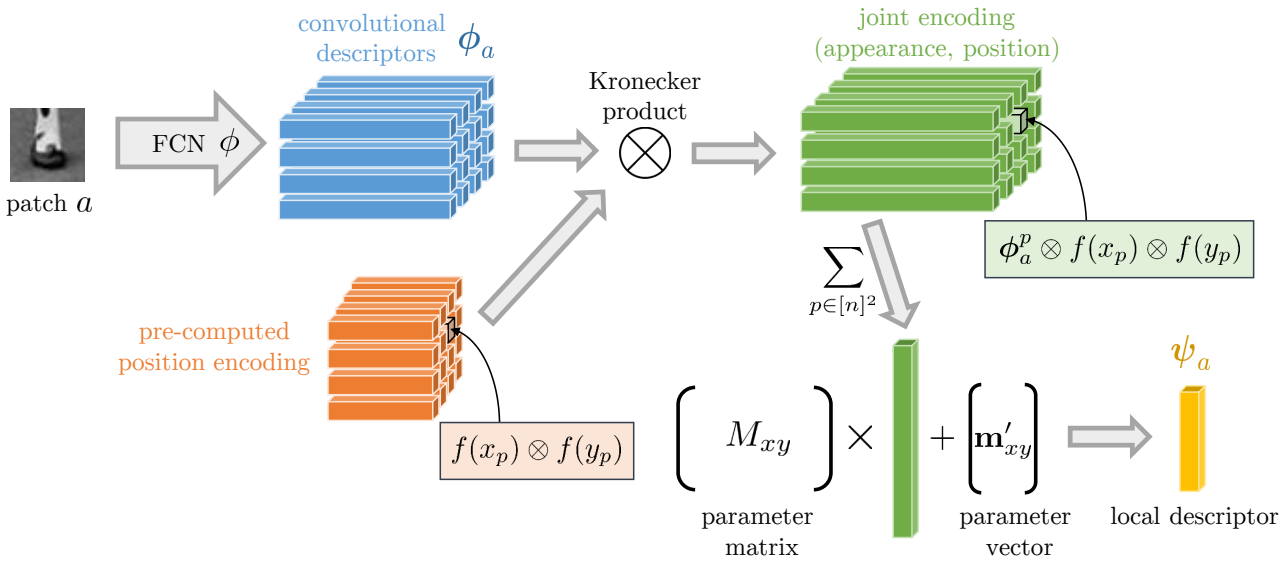


Figure 1. Overview of extraction process for the proposed descriptor. We present the case of ψ^{xy} (10), while other variants are performed in a similar way. $\mathbf{m}'_{xy} = n^2\mathbf{m}_{xy}$.

It is equivalent to the product of descriptor similarity and similarity of positions on the Cartesian grid.

Following the paradigm of descriptor whitening of hand-crafted descriptors [32, 4], we propose the final local descriptor

$$\boldsymbol{\psi}_a^{xy} = \sum_{p\in[n]^2} w_p M_{xy} g_{xy}(\boldsymbol{\phi}_a^p, p) + \mathbf{m}_{xy} \tag{9}$$

$$= M_{xy}\left(\sum_{p\in[n]^2} w_p g_{xy}(\boldsymbol{\phi}_a^p, p)\right) + n^2\mathbf{m}_{xy}, \tag{10}$$

where $M_{xy} \in \mathbb{R}^{D\times d(2s+1)^2}$ and $\mathbf{m}_{xy} \in \mathbb{R}^D$ are parameters to be learned during training, while $w_p = \exp(-\rho_p^2)$ is a weight giving importance according to the distance ρ_p from the center of the patch. Note that in contrast to (3) the same matrix, *i.e.* M_{xy}, is used for all convolutional descriptors. As a result the number of required parameters is reduced and multiplication by M_{xy} can be efficiently performed after the summation (10). In analogy to the encoding of position in a Cartesian coordinate system, we additionally propose the encoding w.r.t. a polar coordinate system[3] by

$$g_{\rho\theta}(\boldsymbol{\phi}_a^p, p) = \boldsymbol{\phi}_a^p \otimes f(\rho_p) \otimes f(\theta_p), \tag{11}$$

[3]For $p = (i, j)$, $\rho_p = \sqrt{(i-c)^2 + (j-c)^2}$ and $\theta_p = \tan^{-1}\frac{j-c}{i-c}$, where $c = (n+1)/2$.

and the corresponding descriptor

$$\boldsymbol{\psi}_a^{\rho\theta} = \sum_{p\in[n]^2} w_p M_{\rho\theta} g_{\rho\theta}(\boldsymbol{\phi}_a^p, p) + \mathbf{m}_{\rho\theta}. \tag{12}$$

Different parameterizations, *i.e.* using different coordinate system, provide tolerance to different kinds of misalignment between patches. Cartesian offers tolerance to translation misalignment, while polar offers tolerance to rotation and scale misalignment. To benefit from both types of tolerance, we further use the combined encoding that uses the two coordinate systems and is produced by concatenation of the previous encoding. It is defined as function $g_c : \mathbb{R}^d \times \mathbb{R}^d \times [n]^2 \to \mathbb{R}^{2D(2s+1)^2}$ given by

$$g_c(\boldsymbol{\phi}_a^p, \tilde{\boldsymbol{\phi}}_a^p, p) = \left((\boldsymbol{\phi}_a^p \otimes f(x_p) \otimes f(y_p))^\top, (\tilde{\boldsymbol{\phi}}_a^p \otimes f(\rho_p) \otimes f(\theta_p))^\top\right)^\top \tag{13}$$

where $\tilde{\phi}$ is used to show that the two encodings do not need to rely on the same FCN ϕ. Subscript c refers to the combined coordinate system, but we skip $xy\rho\theta$ to simplify the notation. The final descriptor proposed in this work is

$$^\star\boldsymbol{\psi}_a^c = \sum_{p\in[n]^2} w_p M_c g_c(\boldsymbol{\phi}_a^p, \tilde{\boldsymbol{\phi}}_a^p, p) + \mathbf{m}_c \tag{14}$$

where $M_c \in \mathbb{R}^{D\times 2d(2s+1)^2}$, and left superscript $\star$ is used to denote that a separate FCN is used for each encoding, correspondingly coordinate system.

Convolutional part ϕ

Conv. layer	Param. matrix shape	# Parameters
1	[1, 32, 3, 3]	288
2	[32, 32, 3, 3]	9,216
3	[32, 64, 3, 3]	18,432
4	[64, 64, 3, 3]	36,864
5	[64, 128, 3, 3]	73,728
6	[128, 128, 3, 3]	147,456
Total		285,984

HardNet	$N = 32$	$N = 64$
$\boldsymbol{\phi}$	285,984	285,984
FC	1,048,576	4,194,304
Total	1,334,560	4,480,288

ψ^{xy}	N={32, 64}	
	s=1	s=2
$\boldsymbol{\phi}$	285,984	285,984
M, $\mathbf{m}$	147,584	409,728
Total	433,568	695,712

ψ^{c}	N={32, 64}	
	s=1	s=2
$\boldsymbol{\phi}$	285,984	285,984
M, $\mathbf{m}$	295,040	819,328
Total	581,024	1,105,312

$^{\star}\psi^{c}$	N={32, 64}	
	s=1	s=2
$\boldsymbol{\phi}$	285,984	285,984
$\tilde{\boldsymbol{\phi}}$	285,984	285,984
M, $\mathbf{m}$	295,040	819,328
Total	867,008	1,391,296

Table 1. Number of parameters for different models. The convolutional part ϕ has identical architecture for all models. Cases where both ϕ and $\tilde{\phi}$ appear use a separate convolutional part for the Cartesian and the polar descriptor. These specifications correspond to $d = 128$, and $D = 128$. The resulting n is equal to 8 and 16 for N equal to 32 and 64, respectively. We report M and $\mathbf{m}$ due to limited space, but we refer to M_{xy}, and M_c according to the respective table, and similarly for $\mathbf{m}$. Descriptor $\boldsymbol{\psi}^{\rho\theta}$ has identical requirements as descriptor $\boldsymbol{\psi}^{xy}$. The parameter requirements of our descriptor remain unchanged for different patch size N.

4. Implementation details

In this section, we provide implementation details that concern the efficiency of the aggregation, describe the different architectures and their required number of parameters, and finally discuss the training procedure.

4.1. Efficient aggregation

We describe the implementation details for variant ψ_a^{xy}, but these hold for other variants too in the same way. Vectors $w_p f(x_p) \otimes f(y_p) \in \mathbb{R}^{(2s+1)^2}$ that encode positions $p \in [n]^2$ are fixed for the 2D grid of size $n \times n$. Thus, we pre-compute and store them in matrix $F \in \mathbb{R}^{n^2 \times (2s+1)^2}$. We reshape 3D tensor $\boldsymbol{\phi}_a$ into matrix $\Phi \in \mathbb{R}^{n^2 \times d}$. Given these two matrices and due to the linearity of matrix to vector multiplication we can re-write the descriptor as

$$\begin{aligned}
\boldsymbol{\psi}_a^{xy} &= \sum_{p \in [n]^2} w_p M_{xy} g_{xy}(\boldsymbol{\phi}_a^p, p) + \mathbf{m}_{xy}, \\
&= M_{xy} \left(\sum_{p \in [n]^2} w_p g_{xy}(\boldsymbol{\phi}_a^p, p) \right) + n^2 \mathbf{m}_{xy}, \qquad (15) \\
&= M_{xy} \left(\sum_{p \in [n]^2} w_p \boldsymbol{\phi}_a^p \otimes f(x_p) \otimes f(y_p) \right) + n^2 \mathbf{m}_{xy}, \\
&= M_{xy} \operatorname{vec}(\Phi^\top F) + n^2 \mathbf{m}_{xy}. \qquad (16)
\end{aligned}$$

Multiplication $\Phi^\top F$ makes the computation memory efficient because it avoids explicit storing of the Kronecker product for each p. To evaluate (15), the memory requirements are $n^2 d(2s+1)^2$ numbers, while to evaluate (16), only $n^2(d + (2s+1)^2)$ numbers are allocated. Using setup $d = 128$ and $s = 2$ in our experiments, the memory requirements are reduced by a factor of 20.9.

4.2. Architecture

We use the HardNet+ [30] architecture for the convolution part, since HardNet+ achieves state-of-the-art performance on all benchmarks. We also use it a baseline to compare with.

The statistics of the convolutional part ϕ are described in Table 1 (left). Each convolutional layer is followed by batch normalization and ReLU, while no bias is used. Table 1 (right) provides the total number of parameters for HardNet+ and our networks, namely, plain polar or Cartesian encoding with different dimensionality of the explicit feature maps ($s = 1$ and $s = 2$ frequencies used), and the joint encoding with a common (ϕ) or separate (ϕ and $\tilde{\phi}$) convolutional part. Note that for the joint encoding with separate convolutional parts and $s = 2$ frequencies, the proposed network needs roughly the same number of parameters as HardNet+ with input patch of size 32×32 pixels ($N = 32$). In all other settings of the proposed architecture, the number of parameters is significantly reduced. Importantly, the number of parameters for larger patch sizes (such as 64×64), that provide better performance, the number of parameters stays fixed for the proposed architecture. For Hardnet+, the number of parameters of the FC layer increases by a factor of 4 for 64×64 input patches.

4.3. Training

We would like to highlight the contribution of the explicit spatial encoding and to provide direct comparison to the current state-of-the-art descriptor construction. To avoid changing many things at the same time, we follow exactly the same training procedure as HardNet+, which we briefly review below.

The network is trained with the triplet loss defined as

$$\ell(\hat{\psi}_{an}, \hat{\psi}_{pos}, \hat{\psi}_{neg}) = [1 - ||\hat{\psi}_{an} - \hat{\psi}_{pos}|| + ||\hat{\psi}_{an} - \hat{\psi}_{neg}||]_+, \quad (17)$$

acting on a triplet formed by an anchor, a positive (matching to the anchor), and a negative (non-matching to the anchor) descriptor. A batch of size 1024 patches is constructed from 512 pairs of anchor-positive descriptors. Regarding a particular pair in the batch, the positive descriptors of all other pairs are considered as candidate negatives. Finally, the one with the smallest Euclidean distance to the anchor within the batch is chosen as a hard negative to form a triplet.

We use Stochastic Gradient Descent (SGD) to perform the training. The total training set consists of 2 million anchor-positive pairs and the training lasts 10 epochs. Data augmentation is employed by random patch rotation, scaling and flipping. The learning rate is set to 10, and linearly decays to zero withing 10 epochs. Momentum is equal to 0.9 and weight decay to 10^{-4}. Random orthogonal initialization is used for the weights of the network [42]. The method is implemented in the PyTorch framework.

5. Experiments

We first describe the datasets and the evaluation protocols used in our experiments, and then present qualitative results showing the impact of the training on patch similarity. Finally we present the results achieved by different variants of our descriptor and show a comparison with the state of the art.

5.1. Datasets and protocols.

We use two publicly available patch datasets, namely *PhotoTourism* (PT) [55] and *HPatches* (HP) [4]. We use the former for both training and evaluation, while the latter only for evaluation when training on PT to show the generalization ability of the descriptor.

The PT dataset consists of following 3 separate sets, Liberty, Notredame and Yosemite. Each consists of local features detected with the Difference-of-Gaussians (DoG) detector and verified through an SfM pipeline. Each set comprises about half a million 64×64 patches, associated with a discrete label which is the outcome of SfM verification. The test set consists of 100k pairs of patches corresponding to the same (positive) 3D point, and an equal number corresponding to different (negative) 3D points. The metric used to measure performance is the *false positive rate at 95% of recall* (FPR@95). Models are trained on one set and tested on the other two, and the mean of 6 scores is reported.

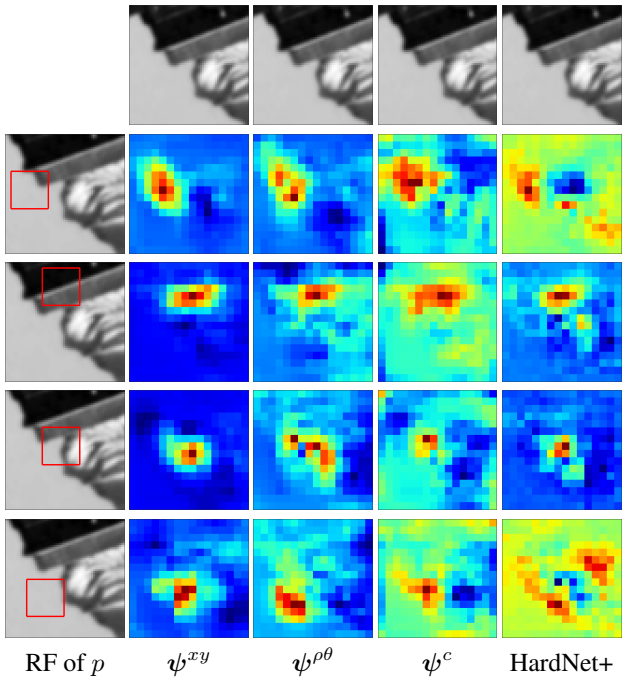


Figure 2. Visualization of similarity between position p of the $n \times n$ grid (rows) on a patch and another whole patch for different methods (columns). Heat-maps are normalized to $[0, 1]$ with red corresponding to the maximum similarity. Red box is used to depict the receptive field (RF) of p.

The HP dataset contains patches of higher diversity and is more realistic. Evaluation is performed on three different tasks, namely verification, retrieval, and matching. Despite the fact that we do not train on HP, we evaluate on all 3 train/test splits and report the average performance to allow future comparisons. We follow the common practice and train our descriptor on Liberty of PT to evaluate on HP.

We repeat each experiment three times, with different random seeds to initialize the parameters, and report mean and standard deviation of the 3 runs. We followed this policy for all variants and datasets.

Recently, larger and more diverse datasets [31, 26] have been introduced to improve local descriptor training. These are shown to improve the performance of state-of-the-art descriptors even by simply replacing the training dataset. We have not included them in our experiments but expect the impact to be similar on our descriptor too.

5.2. Visualizing patch similarity.

We construct encodings $g(\mathbf{v}, p)$, before aggregation, for our descriptors and for the conventional case and construct a similarity map to analyze the impact of the po-

Test			Liberty		Notredame		Yosemite	
Train	# Parameters	Mean	No	Yo	Li	Yo	Li	No
HardNet+ †	1,334,560	1.51	1.49	2.51	0.53	0.78	1.96	1.84
HardNet+	1,334,560	1.43 $\pm$ 0.02	1.25 $\pm$ 0.03	2.35 $\pm$ 0.03	0.48 $\pm$ 0.01	0.74 $\pm$ 0.02	2.15 $\pm$ 0.01	1.61 $\pm$ 0.10
$^{\star}\boldsymbol{\psi}^{c}, s=1$	867,008	1.53 $\pm$ 0.03	1.27 $\pm$ 0.03	2.31 $\pm$ 0.08	0.48 $\pm$ 0.02	0.82 $\pm$ 0.05	2.58 $\pm$ 0.08	1.72 $\pm$ 0.09
$^{\star}\boldsymbol{\psi}^{c}, s=2$	1,391,296	1.36 $\pm$ 0.01	1.14 $\pm$ 0.03	2.16 $\pm$ 0.10	0.42 $\pm$ 0.01	0.73 $\pm$ 0.02	2.18 $\pm$ 0.07	1.51 $\pm$ 0.12

Table 2. Performance comparison of the proposed descriptors with the state-of-the-art descriptor HardNet+ on the PhotoTourism dataset. Performance is measured via FPR@95. We repeat each experiment/training 3 times and report mean performance and standard deviation. Patch size is $N = 32$. †: Reported in the original work.

sition encoding. We present such visualization in Figure 2. We pick position p and compute similarity $g_{fc}(\boldsymbol{\phi}_a^p, p)^\top g_{fc}(\boldsymbol{\phi}_b^q, q), \forall q \in [n]^2$, for the conventional case, and $(M_c g_c(\boldsymbol{\phi}_a^p, p) + \mathbf{m}_c)^\top (M_c g_c(\boldsymbol{\phi}_b^q, q) + \mathbf{m}_c), \forall q \in [n]^2$ for ours in the case of the combined descriptor. We observe how all architectures, including the conventional one, result in large similarity values near p.

5.3. Results and comparisons.

We train and evaluate different variants of the proposed descriptor. If not otherwise stated, we use input patches of size equal to 32×32, which is the standard practice for deep local descriptors. We further examine the case of 64×64 input patches. We always set $d = 128$ and $D = 128$. The dimensionality of the feature maps is controlled by s which we set equal to 1 or 2 in our experiments.

Reproducing HardNet+. Our implementation, training procedure, and training hyper-parameters are based on HardNet+ [4]. We reproduce its training and report our own results, proving that our benefit is not an outcome of implementation details. We report both the achieved performed in the original publication and our reproduced ones in all the comparisons.

Baselines for ablation study. We train and test the following two baselines to see the impact of the position encoding. First, we train a descriptor that encodes convolutional descriptors in $\boldsymbol{\phi}_a$ in a translation invariant way, *i.e.* no position encoding at all. It is implemented by spatial sum pooling on $\boldsymbol{\phi}_a$ and given by

$$\boldsymbol{\psi}_a^{\text{sum}} = \sum_{p \in [n]^2} \boldsymbol{\psi}_a^p. \tag{18}$$

[4] https://github.com/DagnyT/hardnet

The dimensionality of $\boldsymbol{\psi}_a^{\text{sum}}$ is equal to d and not D in this case. However, $d = D = 128$, making this descriptor directly comparable to all others.

Second, we train a descriptor that encodes the spatial information simply by concatenation, *i.e.* vectorization of $\boldsymbol{\phi}_a$, which does not provide any tolerance to position misalignments. It is given by

$$\boldsymbol{\psi}_a^{\text{cat}} = \text{vec}\, \boldsymbol{\psi}_a \tag{19}$$

Impact of position encoding. We compare our descriptor with HardNet+ on PT and show results in Table 2. Conceptually it is a comparison between the conventional architecture that uses an FC layer to "feed" the convolutional descriptors to, and our kernel-based approach to explicitly encode the spatial information. Our descriptors (with $s = 2$) slightly outperforms HardNet+ while it has roughly the same number of parameters. Even the variant with fewer parameters ($s = 1$) performs similarly.

A more thorough comparison, examining the impact of the explicit spatial encoding, is performed on HP and presented in Figure 4. Firstly, we evaluate $\boldsymbol{\psi}^{\text{SUM}}$ as part of an ablation study. It is translation invariant that totally discards the spatial information. It does not require additional parameters other than the ones for FCN ϕ. It has significantly lower performance compared to all the other descriptors. We additionally tried including multiplication by matrix M_{sum} in (18) and did not notice performance improvements. Descriptor $\boldsymbol{\psi}^{\text{CAT}}$ is another case not requiring additional parameters. It is translation variant in a "rigid" way, whose tolerance to translation misalignment is restricted to the amount that the large receptive field offers. Despite the very large dimensionality, it is not a top performer. Even

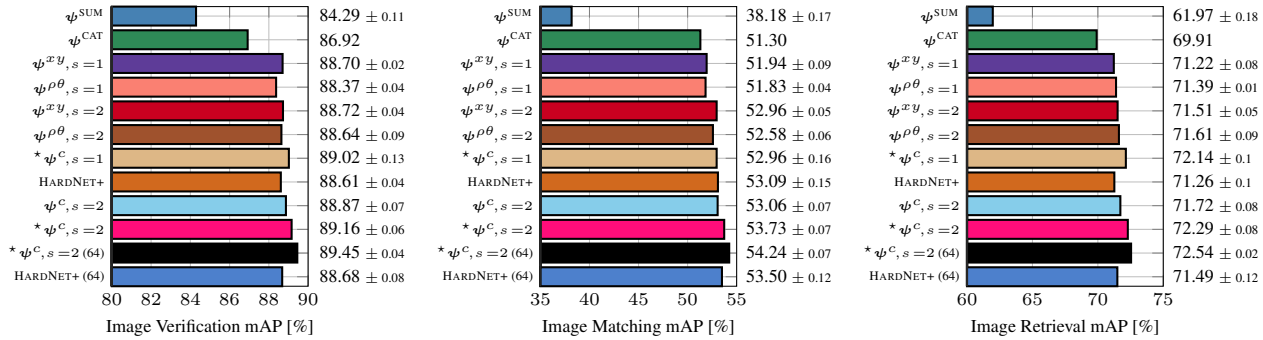


Figure 3. Performance comparison on the HPatches benchmark. The training is performed on the Liberty set of PhotoTourism dataset for all descriptors and with identical setup. Performance is measured via mean Average Precision (mAP). We repeat each experiment/training 3 times and report mean performance and standard deviation (with the exception of ψ^{CAT} that due to very high dimensionality was trained only once). All descriptors have 128 dimensions, with the exception of ψ^{CAT} which has 8192. The methods are sorted w.r.t. the required number of parameters (top is the least demanding, *i.e.* less parameters). All methods are trained and tested with patch size $N = 32$ unless when (64) is reported.

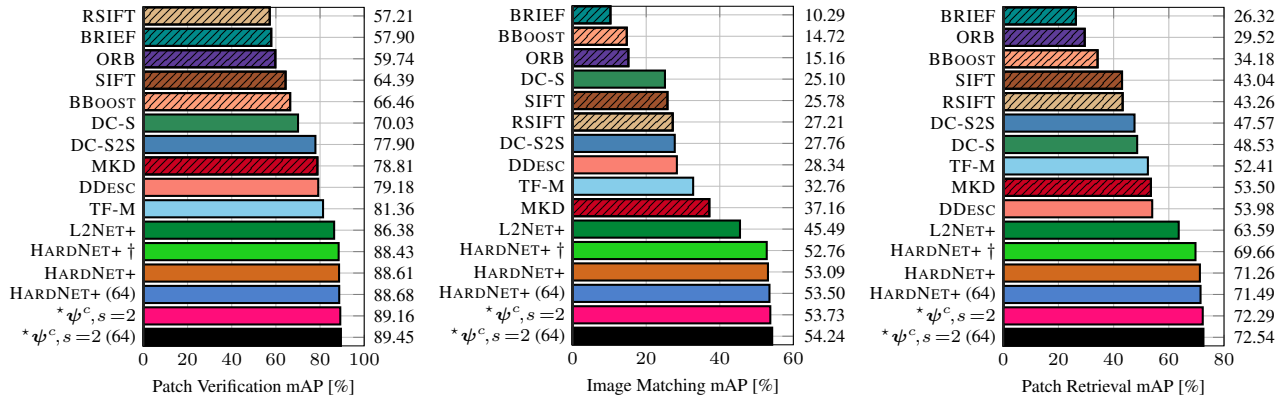


Figure 4. Performance comparison with the state of the art on the HPatches benchmark. The learning for learned descriptors is performed on the Liberty set of PhotoTourism dataset. Hand-crafted descriptors are shown with striped bars. Performance is measured via mean Average Precision (mAP). The performance of our descriptor is the mean of 3 repetitions of each experiment/training. All methods are trained and tested with patch size $N = 32$ unless when (64) is reported. †: Reported in the original work.

our light-weight variant with as few as 127k additional parameters (excluding ϕ) recovers most of the performance loss due to lack of spatial information, *i.e.* w.r.t.ψ^{SUM} . This result suggests that the common choice of an FC layer for deep local descriptors might be over-parametrized. It is not the best performing either. Our variant $^{\star}\psi^{c}, s=2$ is consistently the top performing one on all tasks.

Comparison with the state of the art. We finally present a comparison to the state of the art on HP in Figure 4. The comparison includes a set of hand-crafted and learned local descriptors, namely RSIFT [2], SIFT [25], BRIEF [12], BBoost [52], ORB [41], MKD [32], DeepCompare [58], DDesc [46], TFeat [5], L2Net [49] and HardNet [30]. The proposed descriptor achieves the best performance with a 128D descriptor on all 3 tasks consistently.

6. Conclusions

We interpret conventional convolutional local descriptors as efficient match kernels and show that they learn spatially variant encoding through that last FC layer. We design a novel local descriptor that explicitly encodes the spatial information. We use a combined position parametrization handling different sources of geometric misalignment. It achieves the same performance as state-of-the-art descriptors with fewer parameters and consistently outperforms them on all standard patch benchmarks with the same number of parameters.

Acknowledgments This work was supported by the GAČR grant 19-23165S, the OP VVV funded project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics" and the CTU student grant SGS17/185/OHK3/3T/13.

References

[1] Mitsuru Ambai and Yuichi Yoshida. Card: Compact and real-time descriptors. In *ICCV*, 2011. 2

[2] Relja Arandjelovic and Andrew Zisserman. Three things everyone should know to improve object retrieval. In *CVPR*, 2012. 2, 8

[3] Vassileios Balntas, Edward Johns, Lilian Tang, and Krystian Mikolajczyk. Pn-net: conjoined triple deep network for learning local image descriptors. In *arXiv*, 2016. 2

[4] Vassileios Balntas, Karel Lenc, Andrea Vedaldi, and Krystian Mikolajczyk. Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors. In *CVPR*, 2017. 4, 6

[5] Vassileios Balntas, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. Learning local feature descriptors with triplets and shallow convolutional neural networks. In *BMVC*, 2016. 1, 2, 8

[6] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (SURF). *CVIU*, 110(3):346–359, 2008. 1, 2

[7] Liefeng Bo, Kevin Lai, Xiaofeng Ren, and Dieter Fox. Object recognition with hierarchical kernel descriptors. In *CVPR*, 2011. 2

[8] Liefeng Bo, Xiaofeng Ren, and Dieter Fox. Kernel descriptors for visual recognition. In *NIPS*, December 2010. 2

[9] Liefeng Bo and Cristian Sminchisescu. Efficient match kernels between sets of features for visual recognition. In *NIPS*, 2009. 1, 2

[10] Matthew Brown, Richard Szeliski, and Simon Winder. Multi-image matching using multi-scale oriented patches. In *CVPR*, volume 1, pages 510–517, 2005. 2

[11] Andrei Bursuc, Giorgos Tolias, and Hervé Jégou. Kernel local descriptors with implicit rotation matching. In *ICMR*, 2015. 1, 2

[12] Micheal Calonder, Vincent Lepetit, Cristoph Strecha, and Pascal Fua. Brief: Binary robust independent elementary features. In *ECCV*, 2010. 2, 8

[13] Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cedric Bray. Visual categorization with bags of keypoints. In *ECCV Workshop Statistical Learning in Computer Vision*, 2004. 1

[14] Jingming Dong and Stefano Soatto. Domain-size pooling in local descriptors: Dsp-sift. In *CVPR*, 2015. 1, 2

[15] Xufeng Han, Thomas Leung, Yangqing Jia, Rahul Sukthankar, and Alexander C Berg. Matchnet: Unifying feature and metric learning for patch-based matching. In *CVPR*, 2015. 2

[16] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Alvey vision conference*, volume 15, page 50, 1988. 1

[17] Marko Heikkila, Matti Pietikainen, and Cordelia Schmid. Description of interest regions with local binary patterns. *Pattern recognition*, 42(3):425–436, 2009. 2

[18] Hervé Jégou, Florent Perronnin, Matthijs Douze, Jorge Sánchez, Patrick Pérez, and Cordelia Schmid. Aggregating local descriptors into compact codes. In *IEEE Trans. PAMI*, September 2012. 1

[19] Yannis Kalantidis, Clayton Mellina, and Simon Osindero. Cross-dimensional weighting for aggregated deep convolutional features. *ECCVW*, 2016. 1

[20] Yan Ke and Rahul Sukthankar. PCA-SIFT: a more distinctive representation for local image descriptors. In *CVPR*, pages 506–513, June 2004. 2

[21] Theo Gevers Koen van de Sande and Cees Snoek. Evaluating color descriptors for object and scene recognition. *IEEE Trans. PAMI*, 32(9):1582–1596, 2010. 2

[22] Iasonas Kokkinos and Alan Yuille. Scale invariance without scale selection. In *CVPR*, 2008. 2

[23] Stefan Leutenegger, Margarita Chli, and Roland Y Siegwart. Brisk: Binary robust invariant scalable keypoints. In *ICCV*, 2011. 2

[24] Rosanne Liu, Joel Lehman, Piero Molino, Felipe Petroski Such, Eric Frank, Alex Sergeev, and Jason Yosinski. An intriguing failing of convolutional neural networks and the coordconv solution. In *NIPS*, 2018. 1, 2

[25] David G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004. 1, 2, 8

[26] Zixin Luo, Tianwei Shen, Lei Zhou, Siyu Zhu, Runze Zhang, Yao Yao, Tian Fang, and Long Quan. Geodesc: Learning local descriptors by integrating geometry constraints. In *ECCV*, 2018. 2, 3, 6

[27] Krystian Mikolajczyk and Cordelia Schmid. Scale and affine invariant interest point detectors. *IJCV*, 60(1):63–86, 2004. 1

[28] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *IEEE Trans. PAMI*, 27(10):1615–1630, 2005. 1, 2

[29] Krystian Mikolajczyk, Tinne Tuytelaars, Cordelia Schmid, Andrew Zisserman, Jiri Matas, F. Schaffalitzky, T. Kadir, and Luc Van Gool. A comparison of affine region detectors. *IJCV*, 65(1/2):43–72, 2005. 1

[30] Anastasiia Mishchuk, Dmytro Mishkin, Filip Radenovic, and Jiri Matas. Working hard to know your neighbor's margins: Local descriptor learning loss. In *NIPS*, 2017. 1, 2, 3, 5, 8

[31] Rahul Mitra, Nehal Doiphode, Utkarsh Gautam, Sanath Narayan, Shuaib Ahmed, Sharat Chandran, and Arjun Jain. A large dataset for improving patch matching. In *arXiv*, 2018. 6

[32] Arun Mukundan, Giorgos Tolias, Andrei Bursuc, Hervé Jégou, and Ondrej Chum. Understanding and improving kernel local descriptors. *IJCV*, 2019. 1, 2, 3, 4, 8

[33] Hyeonwoo Noh, Andre Araujo, Jack Sim, Tobias Weyand, and Bohyung Han. Large-scale image retrieval with attentive deep local features. In *ICCV*, 2017. 1

[34] David Novotny, Samuel Albanie, Diane Larlus, and Andrea Vedaldi. Semi-convolutional operators for instance segmentation. In *ECCV*, 2018. 1, 2

[35] Timo Ojala, Matti Pietikainen, and Topi Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Trans. PAMI*, 24(7):971–987, 2002. 2

[36] Aude Oliva and Antonio Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *IJCV*, 42(3):145–175, 2001. 2

[37] Florent Perronnin, Jorge Sánchez, and Thomas Mensink. Improving the fisher kernel for large-scale image classification. In *ECCV*, September 2010. 1

[38] James Philbin, Michael Isard, Josef Sivic, and Andrew Zisserman. Descriptor learning for efficient retrieval. In *ECCV*, 2010. 2

[39] Filip Radenović, Giorgos Tolias, and Ondřej Chum. Fine-tuning cnn image retrieval with no human annotation. In *IEEE Trans. PAMI*, 2018. 1

[40] Ali S Razavian, Josephine Sullivan, Stefan Carlsson, and Atsuto Maki. Visual instance retrieval with deep convolutional networks. *ITE Trans. on Media Technology and Applications*, 2016. 1

[41] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *ICCV*, 2011. 2, 8

[42] Andrew M Saxe, James L McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013. 6

[43] Cordelia Schmid and Roger Mohr. Local grayvalue invariants for image retrieval. *IEEE Trans. PAMI*, 19(5):530–535, 1997. 2

[44] Paul Scovanner, Saad Ali, and Mubarak Shah. A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th ACM International Conference on Multimedia*, pages 357–360, 2007. 2

[45] Eli Shechtman and Michal Irani. Matching local self-similarities across images and videos. In *CVPR*, pages 1–8. IEEE, 2007. 2

[46] Edgar Simo-Serra, Eduard Trulls, Luis Ferraz, Iasonas Kokkinos, Pascal Fua, and Francesc Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *ICCV*, 2015. 1, 2, 8

[47] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Learning local feature descriptors using convex optimisation. *IEEE Trans. PAMI*, 36(8):1573–1585, 2014. 2

[48] Josef Sivic and Andrew Zisserman. Video Google: A text retrieval approach to object matching in videos. In *ICCV*, 2003. 1

[49] Bin Fan Yurun Tian and Fuchao Wu. L2-net: Deep learning of discriminative patch descriptor in euclidean space. In *CVPR*, 2017. 1, 2, 3, 8

[50] Engin Tola, Vincent Lepetit, and Pascal Fua. Daisy: An efficient dense descriptor applied to wide-baseline stereo. *IEEE Trans. PAMI*, 32(5):815–830, 2010. 2

[51] Giorgos Tolias, Andrei Bursuc, Teddy Furon, and Hervé Jégou. Rotation and translation covariant match kernels for image retrieval. *CVIU*, 140:9–20, 2015. 3

[52] Tomasz Trzcinski, Mario Christoudias, Vincent Lepetit, and Pascal Fua. Learning image descriptors with the boosting-trick. In *NIPS*, 2012. 8

[53] Andrea Vedaldi and Andrew Zisserman. Efficient additive kernels via explicit feature maps. In *CVPR*, 2010. 3

[54] Peng Wang, Jingdong Wang, Gang Zeng, Weiwei Xu, Hongbin Zha, and Shipeng Li. Supervised kernel descriptors for visual recognition. In *CVPR*, 2013. 2

[55] Simon Winder and Matthew Brown. Learning local image descriptors. In *CVPR*, 2007. 2, 6

[56] Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, and Pascal Fua. Lift: Learned invariant feature transform. In *ECCV*, pages 467–483. Springer, 2016. 2

[57] Guoshen Yu and Jean-Michel Morel. A fully affine invariant image comparison method. In *ICASSP*, pages 1597–1600. IEEE, 2009. 2

[58] Sergey Zagoruyko and Nikos Komodakis. Learning to compare image patches via convolutional neural networks. In *CVPR*, 2015. 1, 2, 8

XII Asymmetric Feature Maps with Application to Sketch Based Retrieval

Title: Asymmetric Feature Maps with Application to Sketch Based Retrieval
Authors: G. Tolias, O. Chum
Published at: CVPR 2017

Asymmetric Feature Maps with Application to Sketch Based Retrieval

Giorgos Tolias Ondřej Chum
Visual Recognition Group, Faculty of Electrical Engineering, Czech Technical University in Prague
{giorgos.tolias,chum}@cmp.felk.cvut.cz

Abstract

We propose a novel concept of asymmetric feature maps (AFM), which allows to evaluate multiple kernels between a query and database entries without increasing the memory requirements. To demonstrate the advantages of the AFM method, we derive a short vector image representation that, due to asymmetric feature maps, supports efficient scale and translation invariant sketch-based image retrieval. Unlike most of the short-code based retrieval systems, the proposed method provides the query localization in the retrieved image. The efficiency of the search is boosted by approximating a 2D translation search via trigonometric polynomial of scores by 1D projections. The projections are a special case of AFM. An order of magnitude speed-up is achieved compared to traditional trigonometric polynomials. The results are boosted by an image-based average query expansion, exceeding significantly the state of the art on standard benchmarks.

1. Introduction

Efficient match kernel [3] is a popular choice in applications evaluating complex similarity measures on large collections of objects, where an object is a set of elements. This includes local feature descriptors [3, 5] and image retrieval with short descriptors [38][1].

In efficient match kernel, all elements of the sets are mapped to a finite feature map [27, 39]. An inner product of the feature maps approximates evaluation of a specific kernel, defining similarity of the set elements. We propose an extension to this concept. In the asymmetric feature map, the query uses a different embedding than the database objects. The query embedding defines the kernel that is evaluated between the query and the database entries. Thus, multiple kernels can be evaluated while the memory requirements for the database remains the same (up to a scalar per kernel) as for a single kernel to be evaluated. The embeddings are obtained via joint kernel feature map optimization, which significantly improves the quality of kernel approximation for a fixed dimensionality of the feature map.

[1]The authors were supported by the MSMT LL1303 ERC-CZ grant.

The application domain of AFM is wide, in particular any method using efficient match kernel benefits from AFM. We evaluate the AFM on a sketch-based retrieval application. Sketch-based retrieval has received less attention than image retrieval and still remains challenging. Instead of a real image, the query consists of an abstract binary sketch. This allows the user to quickly outline an object, *e.g.* by a finger on a tablet or smart phone, and search for relevant images (see Figure 1). The progress in this area has more or less followed the footsteps of traditional image retrieval. The first systems employed global descriptors [8]. Then, the Bag-of-Words paradigm with local descriptors and feature quantization [17, 16, 30] was adopted.

Due to the absence of textural cues on the query side, the image representations are shape based. Bridging the representation gap between hand-drawn sketches and real images is one of the challenges making the task difficult. Matching based on shape information has been addressed previously. For instance, in object recognition and detection [2, 17, 23], a costly online matching is performed, which prevented the methods to scale to large image collections. Recent methods manage to index million [7] to billion [36] images for sketch-based retrieval, at the cost of sacrificed invariance to geometric transformations.

To demonstrate the impact of the AFM, we propose a short vector image representation allowing to index large image collections for sketch-based search. Scale and translation invariant real-time search allows to process an order of millions of images per one processor thread. The AFM based method achieves state-of-the-art results on standard benchmarks. The method runs at speed comparable to previously published approaches tailored to sketch-based

Figure 1. Scale and translation invariant query-by-sketch retrieval. An example of sketch queries and top-retrieved images with the sketch localization overlaid in green color.

search. Compared with methods based on efficient match kernel [38], the proposed method achieves order of magnitude speed-up. Unlike most of the methods using low-dimensional descriptors, the proposed method delivers localization of the object in both scale and space. The scale invariance is achieved by evaluating multiple kernels without the need to store multiple representations for database images. The translation invariance and object localization is provided by an efficient similarity evaluation on a 2D grid of translations. Namely, the four main contributions of this work are as follows. (1) Asymmetric explicit feature maps allowing the use of multiple kernel functions without constructing multiple representations for database items are proposed. (2) A joint kernel approximation approach for multiple kernels is derived, generalizing a recent approach of low dimensional explicit feature maps (LDFM) [9]. (3) The scoring through trigonometric polynomial introduced in [38] is further extended and a significant speed-up of its evaluation is proposed. (4) State-of-the-art sketch-based image retrieval based on the AFM, which is further boosted by query expansion which acts, not on the edge maps as standard sketch matching, but on the original images.

The rest of paper is organized as follows. Related work is discussed in Section 2 and the necessary background is presented in Section 3. Sections 4 and 5 describe our contributions on asymmetric explicit feature maps and on sketch retrieval, respectively, while the retrieval procedure and the experimental evaluation are analyzed in Section 6.

2. Related work

The most similar work to ours is the approach of Tolias *et al.* [38], where the trigonometric polynomial scores were introduced in the context of image retrieval (see Section 3.3 for technical details). Shape properties of local features, such as dominant orientation or position, are jointly encoded with the SIFT descriptor. Despite initially assuming aligned objects, their kernel descriptor comes with an efficient way to compute similarity over multiple image transformations. Compared to their method, asymmetric feature maps introduced in our paper: i) reduce the memory requirements of multi-scale search by roughly a factor of 3, and ii) achieve an order of magnitude speed-up through approximate translation search. The trigonometric polynomials have been also used by Bursuc *et al.* [5] in the context of rotation invariant feature descriptors. The descriptor has recently shown competitive results with CNN based approaches [1].

Since we demonstrate the advantages of AFM on sketch based retrieval, we provide a brief review of relevant literature on this topic. The line of research that focuses on sketches includes recognition [14, 40] or retrieval [24] of sketches. This paper addresses sketch-based image retrieval, which tries to match sketch queries to real images from a large collection. Following successful examples of traditional image retrieval, sketch-based methods employ global image representation [8, 29] or local descriptors and the Bag-of-Words model. In the latter case, representative methods employ local descriptors that are traditionally used on images [16, 30] or proposed particularly for this task [15, 28, 18, 6]. Some examples are HOG descriptors which are adapted for sketch retrieval [18] and were recently extended to capture color [4], symmetry-aware and flip invariant descriptors [6], and descriptors based on local contour fragments [28]. Generic approaches performing learning of discriminative features have been shown effective for sketch retrieval too [33].

Chamfer matching appears to be a good similarity measure for object shapes [37]. Recent attempts focus on Chamfer matching approximations in order to increase scalability. Cao *et al.* [7] binarize the distance transform map and manage to index two million images. However, their approach completely lacks invariance. The same holds for the work of Sun *et al.* [36] who increase the scale of the indexed collection up to one billion. Despite the achievement of scalability, rough approximations of Chamfer matching sacrifice accuracy. Recently, Parui and Mittal [25] proposed a similarity invariant approach able to index up to one million images. Their solution is based on dynamic programming to match chains of contour lines, while the main drawback is the costly off-line indexing.

3. Background

We briefly review the necessary background, which includes efficient match kernels [3], explicit feature maps [39] and efficient trigonometric polynomial scores [38].

3.1. Efficient Match Kernels

In many situations, an object is described by a set of measurements $\mathcal{P} = \{p \in \mathbb{R}^d\}$. Employing a mapping $\Psi : \mathbb{R}^d \to \mathbb{R}^D$ to the elements of $\mathcal{P}$, the set representation of efficient match kernels is defined as

$$\mathbf{V}(\mathcal{P}) = \sum_{p \in \mathcal{P}} \Psi(p). \tag{1}$$

Then, a dot product between the set representation yields the similarity between sets

$$\mathcal{S}(\mathcal{P}, \mathcal{Q}) = \mathbf{V}(\mathcal{P})^\top \mathbf{V}(\mathcal{Q}) = \sum_{p \in \mathcal{P}} \sum_{p \in \mathcal{Q}} \Psi(p)^\top \Psi(q). \tag{2}$$

Normalized similarity is computed by cosine similarity [38], *i.e.*, dot product of ℓ_2 normalized vectors,

$$\bar{\mathcal{S}}(\mathcal{P}, \mathcal{Q}) = \frac{\mathbf{V}(\mathcal{P})^\top \mathbf{V}(\mathcal{Q})}{\sqrt{\mathbf{V}(\mathcal{P})^\top \mathbf{V}(\mathcal{P})} \sqrt{\mathbf{V}(\mathcal{Q})^\top \mathbf{V}(\mathcal{Q})}}, \tag{3}$$

while another choice is to normalize by the set cardinality [3]. Herein, the cosine similarity is adopted ensuring self-similarity is normalized to one. A number of image representations, such as BOW [35, 11], Fisher vectors [26], or VLAD [20], can be interpreted as efficient match kernels.

3.2. Explicit feature maps

Let $K(p, q)$ be a one-dimensional (p is now scalar) positive definite stationary kernel [32] $K : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$. The value of a stationary kernel by definition depends only on the difference $\lambda = p - q$,

$$K(p, q) = K(p, p - \lambda) = k(\lambda), \tag{4}$$

where $k(\lambda)$ is a signature of kernel $K(p, q)$. Due to Bochner's theorem, kernel signature k can be written as

$$k(\lambda) = \int_0^\infty \alpha(\omega) \cos(\omega\lambda)\, \mathrm{d}\omega, \tag{5}$$

where $\alpha(\omega) : \mathbb{R}_0^+ \to \mathbb{R}_0^+$. The kernel signature is approximated by sum over a finite set Ω of frequencies

$$\hat{k}(\lambda) \approx \sum_{\omega \in \Omega} \alpha_\omega \cos(\omega\lambda), \tag{6}$$

where $\alpha_\omega \in \mathbb{R}_0^+$. Applying the trigonometric identity

$$\cos(p - q) = \cos(p)\cos(q) + \sin(p)\sin(q) \tag{7}$$

gives rise to feature map (or feature embedding) $\Psi_\omega : \mathbb{R} \to \mathbb{R}^2$ defined as

$$\Psi_\omega(p) = \left(\sqrt{\alpha_\omega}\cos(\omega p),\ \sqrt{\alpha_\omega}\sin(\omega p)\right)^\top. \tag{8}$$

The inner product of two such vectors reconstructs the terms of equation (6) since $\Psi_\omega(p)^\top \Psi_\omega(q) = \alpha_\omega \cos(\omega(p - q))$. Let the feature map $\Psi(p) : \mathbb{R} \to \mathbb{R}^D$ be constructed as a concatenation of $\Psi_\omega(p)$ for all $\omega \in \Omega$. Now, the inner product

$$\Psi(p)^\top \Psi(q) = \hat{k}(p - q) \approx K(p, q) \tag{9}$$

evaluates the approximation of the kernel signature (6) and hence approximates the original kernel K. The choice of the number of frequencies $|\Omega|$ determines the quality of the approximation and the dimensionality of the embedding. The dimensionality is $2|\Omega|$, or $2|\Omega| - 1$ if $0 \in \Omega$[2].

Feature map construction. We mention in detail (and compare) two approaches to construct the explicit feature maps. We do not consider random feature maps [27], which approximate the integral in (5) using Monte-Carlo methods. Such feature maps provide a poor approximation for low-dimensional feature maps.

Vedaldi and Zisserman [39] propose the following approximation to a kernel signature $k(\lambda)$ on an interval $\lambda \in [-\Lambda, \Lambda]$. First, a periodic function g with period 2Λ is constructed, so that $g(\lambda) = k(\lambda)$ for $\lambda \in [-\Lambda, \Lambda]$. The feature map is then efficiently obtained by approximating periodic g using harmonic frequencies only. This approach has been shown sub-optimal [9]. Further, the periodic function g is not even guaranteed to be positive definite.

A convex optimization approach is proposed by Chum [9]. The input domain of $\hat{k}(\lambda)$ is discretized to finite set $Z \subset [0, \Lambda]$. The quality of the approximation is measured at points in Z as, for example, an ℓ_∞ norm

$$C_\infty(k, \hat{k}) = \max_{\lambda \in Z} |k(\lambda) - \hat{k}(\lambda)|. \tag{10}$$

The set of frequencies $\Omega \subset \bar{\Omega}$ are selected from a pool of frequencies $\bar{\Omega}$, and corresponding weights $\alpha_\omega \geq 0$, $\omega \in \bar{\Omega}$ jointly through a solution of a linear program

$$\min_k C(k, \hat{k}) + \gamma \textstyle\sum_{\omega \in \bar{\Omega}} \alpha_\omega, \tag{11}$$

where $\gamma \in \mathbb{R}^+$ is a weight on the l_1 regularizer controlling the trade-off between the quality of the approximation and the sparsity of α_ω. This is the method we adopt and extend in this work.

3.3. Alignment using trigonometric polynomials

Tolias *et al.* [38] propose an image representation derived by efficient match kernels and explicit feature maps. We focus on the case that all measurements of set $\mathcal{P}$ are shifted by a constant value Δp; note that measurements p are now scalars. The similarity under such shift forms a trigonometric polynomial

$$\mathcal{S}(\mathcal{P}_{\Delta p}, \mathcal{Q}) = \sum_{\omega \in \Omega} \left(\beta_\omega \cos(\omega \Delta p) + \gamma_\omega \sin(\omega \Delta p)\right), \tag{12}$$

with $\mathcal{P}_{\Delta p} = \{p - \Delta p, p \in \mathcal{P}\}$. Parameters β_ω and γ_ω are given by dot products of relevant sub-vectors of $\mathbf{V}(\mathcal{P})$ and $\mathbf{V}(\mathcal{Q})$. Finally the similarity measure that is invariant under such shifting is given by $\mathcal{S}_1(\mathcal{P}_{\Delta p}, \mathcal{Q}) = \max_{\Delta p} \mathcal{S}(\mathcal{P}_{\Delta p}, \mathcal{Q})$.

We postpone further analysis of polynomials of scores until the image representation is introduced in Section 5.

4. Asymmetric feature maps

In this section, we introduce the concept of asymmetric feature maps. Unlike in classical explicit feature maps, a different feature map $\hat{\Psi}$ is used on the query side and a different one $\hat{\Psi}'$ is used on the database side. We show that with asymmetric feature maps, a number of different kernels can be efficiently evaluated between query and database vectors while keeping the database storage of fixed size. Compare the feature map in equation (8) to the following feature maps for the query and database side respectively

$$\hat{\Psi}_\omega(q) = \left(\alpha_\omega \cos(\omega q),\ \alpha_\omega \sin(\omega q)\right)^\top \tag{13}$$

$$\hat{\Psi}'_\omega(p) = \left(\cos(\omega p),\ \sin(\omega p)\right)^\top. \tag{14}$$

The inner products $\hat{\Psi}(q)^\top \hat{\Psi}'(p) = \Psi(q)^\top \Psi(p)$ are preserved. The kernel function is fully defined by the weights on the query side. No additional storage is required on the database side to evaluate the kernel. The same holds for efficient match kernels, as (1) is a normalized sum of feature maps. To evaluate the cosine similarity (3), only a single scalar per kernel $K^{(i)}$ needs to be stored for each database entry $\mathcal{P}$ – the ℓ_2 norm $\sqrt{\mathbf{V}^{(i)}(\mathcal{P})^\top \mathbf{V}^{(i)}(\mathcal{P})}$, which is computed offline.

[2] If $0 \in \Omega$, then $\alpha_0 \sin(0\lambda) = 0$ for all λ can be dropped from the explicit feature map.

Joint approximation of multiple kernels. In order to evaluate a number of different kernels $K^{(i)}(p,q)$ using the asymmetric feature maps, all respective explicit feature maps $\Psi^{(i)}$ have to be based on the same set of frequencies Ω. A naive approach would be to optimize the set of frequencies for one of the kernels and keep it fixed for other kernels. This approach, however, leads to poor approximation, as shown in Figure 2. We propose an extension to LDFM [9] to jointly approximate a set of kernels $K^{(i)}$ represented by their respective kernel signatures $k^{(i)}$, $i \in \{1 \dots n\}$. The quality of the approximation is measured by the sum of individual qualities (10)

$$C^*_\infty = \sum_{i=1}^{n} C_\infty(k^{(i)}, \hat{k}^{(i)}) = \sum_{i=1}^{n} \max_{\lambda \in Z} |k^{(i)}(\lambda) - \hat{k}^{(i)}(\lambda)|.$$

The optimization is performed by executing a linear program

$$\min_{\alpha^{(i)}_\omega |\ \omega \in \Omega} C^*_\infty + \gamma \sum_{\omega \in \Omega} \max_i \alpha^{(i)}_\omega, \tag{15}$$

where γ is a weight of the sparsity regularizer that controls the number of frequencies used, *i.e.* the dimensionality of the feature map. Following the approach of Chum [9], to ensure the required dimensionality of the feature map, a binary search for γ is performed.

Figure 2 presents the approximation of three different kernels using the same set of frequencies. We compare the approximation using only harmonic frequencies, the naive approximation mentioned above, and our joint approximation. The latter has a significantly better fit.

5. Sketch-Based Retrieval

In this section we present our sketch descriptor employing explicit feature maps and elaborate on the efficient trigonometric polynomial of scores to further approximate it. Our methodology is presented for the symmetric feature maps, while the asymmetric case is equivalent. We finally present efficient ways to perform the initial ranking and re-ranking for sketch-based image retrieval.

5.1. Sketch descriptor

Consider a binary sketch as a set of contour points, that is a set of pixels $\mathcal{P}$ that lie on the contour. A *contour pixel* $p \in \mathcal{P}$ is represented as $p = (p_x, p_y, p_\phi, p_w)$, where p_x and p_y are 2D image coordinates, p_ϕ is the gradient angle (or orientation) of the contour at (p_x, p_y), and p_w is a strength of the gradient. For real images, the contour parameters are obtained form an edge detector. For sketches, $p_w = 1$ is set for all contour pixels.

The similarity between contour pixels is computed using a multiplicative kernel composed of three one-dimensional kernels, spatial kernels over p_x, p_y, and an orientation kernel over p_ϕ. The 1D stationary kernels are denoted $K_x(p_x, q_x) = k_x(\lambda_x)$, $K_y(p_y, q_y) = k_y(\lambda_y)$, and $K_\phi(p_\phi, q_\phi) = k_\phi(\lambda_\phi)$ respectively. The *sketch descriptor* is a weighted sum of contour pixel feature maps[3]

$$\mathbf{V}(\mathcal{P}) = \sum_{p \in \mathcal{P}} p_w \Psi(p_x) \otimes \Psi(p_y) \otimes \Psi(p_\phi). \tag{16}$$

It is easy to show that *sketch similarity* (2) becomes

$$\mathcal{S}(\mathcal{P}, \mathcal{Q}) = \sum_{p \in \mathcal{P}} \sum_{q \in \mathcal{Q}} p_w q_w k_x(\lambda_x) k_y(\lambda_y) k_\phi(\lambda_\phi). \tag{17}$$

The orientation and spatial kernels are implemented by 1D RBF kernels with parameters σ_ϕ and $\sigma_x = \sigma_y$, respectively. The set of frequencies are denoted by Ω_ϕ and $\Omega_x = \Omega_y$, while the dimensionality of the corresponding embeddings is $D_x = 2|\Omega_x| - 1$ and $D_\phi = 2|\Omega_\phi| - 1$, respectively. Note that frequency $\omega = 0$ is always included. The sketch descriptor has dimensionality ${D_x}^2 D_\phi$.

The proposed representation constitutes a holistic representation encoding the global sketch shape. We now define a representation encoding only one of the spatial coordinates along with the orientation. It is equivalent to the projection of contour pixels on the horizontal/vertical image axis. The sketch descriptor derived by projection on the horizontal axis is given by

$$\mathbf{V}_x(\mathcal{P}) = \sum_{p \in \mathcal{P}} p_w \Psi(p_x) \otimes 1 \otimes \Psi(p_\phi), \tag{18}$$

where the $\otimes 1$ can be omitted and is only used to show, that the x-projection is a sub-vector of (16) and hence a special case of the proposed asymmetric feature map. This stems from the presence of the constant component of the feature map for y, corresponding to $0 \in \Omega_y$. An analogous derivation holds for $\mathbf{V}_y(\mathcal{P})$ and vertical projection.

5.2. Position alignment

The sketch descriptor encodes spatial coordinates and orientation of contour pixels. Therefore, alignment of objects is assumed, *i.e.* centered and up-right objects. Such an assumption does not hold in real image collections and introduces significant limitations. We now detail the polynomial of scores (mentioned in Section 3) proposed by Tolias *et al.* [38]. We show that translation invariance is achieved by polynomial of scores, and that its evaluation can be efficiently approximated to speed up the search process.

One dimensional. Consider the x-projected sketch descriptor $\mathbf{V}_x(\mathcal{P})$. Let $\mathcal{P}_{\Delta x}$ be the shifted version sketch $\mathcal{P}$ where all contour pixels are horizontally translated by Δx. Elementary trigonometric identities allow us to show that

$$\begin{aligned} \Psi^c_\omega(x - \Delta x) &= \Psi^c_\omega(x)\cos(\omega \Delta x) + \Psi^s_\omega(x)\sin(\omega \Delta x) \\ \Psi^s_\omega(x - \Delta x) &= \Psi^s_\omega(x)\cos(\omega \Delta x) - \Psi^c_\omega(x)\sin(\omega \Delta x), \end{aligned} \tag{19}$$

[3] We use Ψ to denote both the spatial and orientation feature map and simplify the notation. In fact, $\Psi(p_x)$ and $\Psi(p_y)$ approximate the spatial kernels k_x and k_y, respectively, which are identical, while $\Psi(p_\phi)$ the orientation kernel k_ϕ.

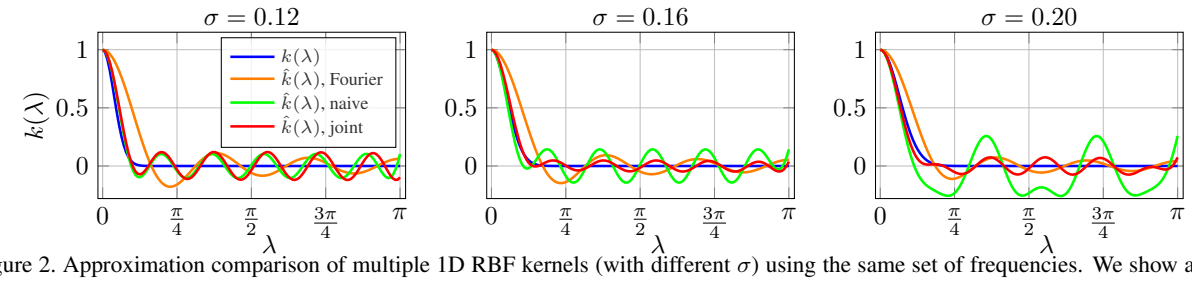


Figure 2. Approximation comparison of multiple 1D RBF kernels (with different σ) using the same set of frequencies. We show approximation using harmonic frequencies only, a naive approach of optimizing the leftmost kernel and using the same frequencies for all, and our joint approximation. Maximum value is normalized to one such that the errors are comparable. $|\Omega| = 7$ for all approximations.

where Ψ^c_ω and Ψ^s_ω denote the first and second dimension of Ψ_ω (8), respectively. Let $\mathbf{V}^c_\omega(\mathcal{P})$ be the sub-vector of $\mathbf{V}(\mathcal{P})$ comprised all elements that contain term $\Psi^c_\omega(x)$, and similarly for $\mathbf{V}^s_\omega(\mathcal{P})$. It turns out that the descriptor of the translated sketch is constructed from that of the original sketch

$$\begin{aligned}\mathbf{V}^c_\omega(\mathcal{P}_{\Delta x}) &= \mathbf{V}^c_\omega(\mathcal{P})\cos(\omega\Delta x) + \mathbf{V}^s_\omega(\mathcal{P})\sin(\omega\Delta x)\\ \mathbf{V}^s_\omega(\mathcal{P}_{\Delta x}) &= \mathbf{V}^s_\omega(\mathcal{P})\cos(\omega\Delta x) - \mathbf{V}^c_\omega(\mathcal{P})\sin(\omega\Delta x).\end{aligned} \quad (20)$$

The sketch similarity between sketches $\mathcal{P}$ and $\mathcal{Q}$ under horizontal translation Δx is a trigonometric polynomial

$$\mathcal{S}(\mathcal{P}_{\Delta x}, \mathcal{Q}) = \sum_{\omega\in\Omega_x} \left(\beta_\omega \cos(\omega\Delta x) + \gamma_\omega \sin(\omega\Delta x)\right), \quad (21)$$

with coefficients β_ω and γ_ω

$$\begin{aligned}\beta_\omega &= \mathbf{V}^c_\omega(\mathcal{P})^\top \mathbf{V}^c_\omega(\mathcal{Q}) + \mathbf{V}^s_\omega(\mathcal{P})^\top \mathbf{V}^s_\omega(\mathcal{Q})\\ \gamma_\omega &= \mathbf{V}^s_\omega(\mathcal{P})^\top \mathbf{V}^c_\omega(\mathcal{Q}) - \mathbf{V}^c_\omega(\mathcal{P})^\top \mathbf{V}^s_\omega(\mathcal{Q}).\end{aligned} \quad (22)$$

The coefficients β_ω and γ_ω of this polynomial are computed by two products of sub-vectors with D_ϕ dimensions. In total there are $N_1 = D_x$ coefficients to be computed. Finally, similarity for any translation with (21) has cost equal to N_1 scalar multiplications. If the candidate translations are fixed, then terms $\cos(\omega\Delta x)$ and $\sin(\omega\Delta x)$ can be precomputed. Normalized similarity comes at no extra cost since the ℓ_2 norm of sketch descriptor remains constant under translations (k_x is a stationary kernel):

$$\mathbf{V}(\mathcal{P}_{\Delta x})^\top \mathbf{V}(\mathcal{P}_{\Delta x}) = \mathbf{V}(\mathcal{P})^\top \mathbf{V}(\mathcal{P}). \quad (23)$$

Similarity that is invariant to horizontal translation is computed by maximizing (21) for all possible translations

$$\mathcal{S}_x(\mathcal{P}_{\Delta x}, \mathcal{Q}) = \max_{\Delta x} \mathcal{S}(\mathcal{P}_{\Delta x}, \mathcal{Q}). \quad (24)$$

Note that this similarity is also invariant to vertical translation as y coordinate is not encoded at all. However, this makes the representation less discriminative. The actual *sketch transformation* aligning the two shapes is given by $\hat{x}_1 = \arg\max_{\Delta x} \mathcal{S}(\mathcal{P}_{\Delta x}, \mathcal{Q})$. Similarity based on the vertical projection is defined in a similar way.

Two dimensional. Consider the full 2D translation $(\Delta x, \Delta y)$. Descriptor $\mathbf{V}(\mathcal{P})$ encoding both spatial coordinates is used. The corresponding second order trigonometric polynomial [38] of scores $\mathcal{S}(\mathcal{P}_{\Delta x,\Delta y}, Q)$ is constructed similarly to the first order one. The details are omitted for the sake of brevity. It allows for an efficient evaluation of similarity for multiple 2D translations in a sliding window manner. The cost to compute one of its coefficients is $4D_\phi$. There are $N_2 = 4(|\Omega_x| - 1)^2 + 4(|\Omega_x| - 1) + 1$ non-zero coefficients in total. The similarity computation for a single 2D translation has cost equal to N_2 scalar multiplications. Translation invariant similarity is given by $\mathcal{S}_{xy}(\mathcal{P}_{\Delta x,\Delta y}, \mathcal{Q}) = \max_{(\Delta x,\Delta y)} \mathcal{S}(\mathcal{P}_{\Delta x,\Delta y}, \mathcal{Q})$, and the transformation aligning the two shapes is given by $(\hat{x}_2, \hat{y}_2) = \arg\max_{(\Delta x,\Delta y)} \mathcal{S}(\mathcal{P}_{\Delta x,\Delta y}, \mathcal{Q})$.

Figure 3. Sketch (left) and the edge map (middle) of a real image (right). We depict the translations maximizing similarity based on 1D projections (magenta) and the full 2D case (green).

In Figures 3 and 4 we present an alignment example between a sketch and a real image. Similarity is computed based on the horizontal and vertical projections, while also for the 2D case. Maximum similarity is met at translations that align the two silhouettes.

5.3. Efficient retrieval and query expansion

Herein, we propose three methods how to avoid exhaustive evaluation of $\mathcal{S}(\mathcal{P}_{\Delta x,\Delta y}, \mathcal{Q})$. First method efficiently selects a shortlist of images on which the score $\bar{\mathcal{S}}_{xy}$ is computed. The other two methods are designed to limit the number of possible translations over which $\mathcal{S}(\mathcal{P}_{\Delta x,\Delta y}, \mathcal{Q})$ is evaluated to obtain a good approximation of $\bar{\mathcal{S}}_{xy}$.

Shortlist by projections. The similarities $\bar{\mathcal{S}}_x$ and $\bar{\mathcal{S}}_y$ computed over the projections (22) provide an estimate of the $\bar{\mathcal{S}}_{xy}$. We propose to use this estimate for initial ranking and to compute the slow similarity $\bar{\mathcal{S}}_{xy}$ only on a shortlist of top S images. Experiments show that initial ranking by $\bar{\mathcal{S}}_x + \bar{\mathcal{S}}_y$ outperforms ranking that uses only one projection. To further speed-up the evaluation for large-scale collections, we propose *discriminative projection first* approach. In this method, one projection is computed over the whole dataset, creating a pre-shortlist of $3S$ images with

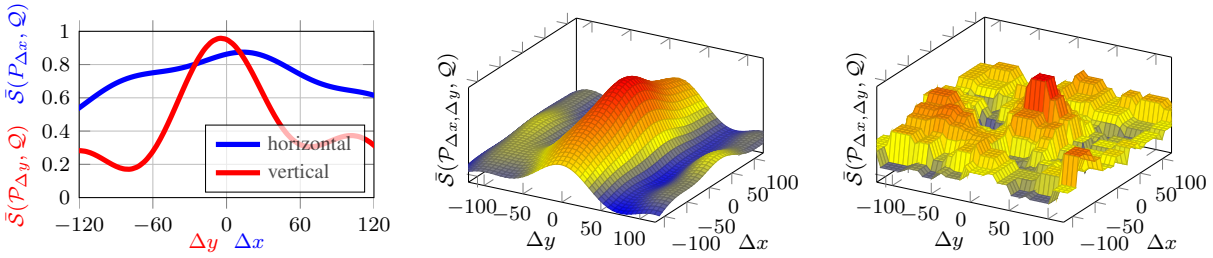


Figure 4. Alignment results for the example in Figure 3. Similarity as a function of translation (in pixels): independent 1D projections (left), the full 2D translation (middle), and the 2D binarized polynomial (right). At zero translation the centers of the sketch and the image are aligned. The detection in magenta color (Figure 3) is based on the similarity shown at the left, while the one in green is based on that shown in the middle.

the highest score. The second projection is only evaluated on this pre-shortlist. Now, the shortlist based on the value of $\bar{\mathcal{S}}_x + \bar{\mathcal{S}}_y$ is a sub-set of the pre-shortlist. The first projection used is query dependent, the one with higher variance in the relevant coordinate in the sketch query is used. Discriminative projection first is denoted as $\bar{\mathcal{S}}_> \xrightarrow{+} \bar{\mathcal{S}}_<$.

Re-ranking by local refinement. The 1D alignment provides, besides the scores, the scale and 1D translations $(\hat{x}_1, \hat{y}_1)$ maximizing the 1D projection scores, which often is a rough approximation of the full 2D alignment (see Figure 3 and 4). In this approach, the full similarity $\mathcal{S}(\mathcal{P}_{\Delta x,\Delta y}, \mathcal{Q})$ is only evaluated evaluated for a small neighborhood of $(\hat{x}_1, \hat{y}_1)$ on a fixed 2D grid. Sketch similarity computed by this method is denoted by $\mathcal{S}_{x/y}$.

Re-ranking by binary polynomial. We efficiently approximate the second order polynomial by a corresponding one that has binary coefficients and variables (*i.e.* $\cos(\omega_x \Delta x)\cos(\omega_y \Delta y)$ is binarized). We simply binarize both by a sign function. The similarity approximation for 2D translation is given by dot product between binary vectors which is faster to compute. Translation maximizing the binary approximation is found, and $\mathcal{S}(\mathcal{P}_{\Delta x,\Delta y}, \mathcal{Q})$ is only computed on a small neighborhood, as in the local refinement. Figure 4 shows an example where the position of the maximum on the 2D map of similarities for the binarized case remains close to that of the real valued one. Experiments show that the binary polynomials provide very good estimate of the translation. We denote this method by $\mathcal{S}_{xy\star}$.

Query expansion. Query Expansion (QE) is a standard approach to improve retrieval results by a new query that exploits the top-ranked results [10, 12, 21]. Unlike the original query, the QE is performed on image descriptors, the sketch descriptors are only used for localization. A global CNN image descriptor is used for QE, in particular off-the-shelf CroW [22] with VGG16 network [34]. The 512D image descriptor extracted per database image is compressed using product quantization [19] into 64 bytes. A basic version of an Average Query Expansion (AQE) [10] is used. CroW descriptors of the top results are averaged and a query is issued.

6. Experiments

We briefly summarize the design choices of the indexing and search procedure of our sketch-based retrieval. Then, we evaluate our method and compare to the state of the art.

Indexing (offline stage). All database images are down-sampled to have the longer side equal to 400 pixels. The edges are detected by off-the-shelf detector of Dollár and Zitnick [13]. The output edge strength is used as p_w, while all edges with strengths lower than 0.2 are completely discarded. A single sketch descriptor per database image is computed with AFM (14). Three kernels are used to search at three scales. Finally, the corresponding ℓ_2 norms for normalizing similarity (3) are computed and stored.

Query (online stage). The sketch query is cropped with a tight bounding box and resized similarly to database images. Two additional scales are given by down-sampling to 80% and 60%. Different query scales need to be matched with different kernels; smaller scale is matched with narrower kernel. The kernels shown in Figure 2 are used accordingly. The orientation kernel has $\sigma_\phi = 0.8$. One query descriptor per kernel is constructed (13). Additionally, each query is also horizontally mirrored.

The translations to be evaluated are fixed in a uniform way. Maximum translation is set to 80 pixels towards both directions and the step is 20. These are used for the maximum query size, while for different scales the maximum translation (step) is increased (decreased) linearly according to the relative query scale. That means, the localization is finer for smaller scales. Similarity is computed per scale independently and maximum similarity is kept.

The descriptor dimensionality is given by the number of frequencies $|\Omega_x|$ and $|\Omega_\phi|$. For instance, a compact setting of $|\Omega_x| = 5$ and $|\Omega_\phi| = 2$ lead to a 243D descriptor, while a high-performance settings of $|\Omega_x| = 6$ and $|\Omega_\phi| = 3$ lead to a 605D descriptor. In all cases, 9 additional scalars per image are stored (normalization of the 2D descriptor, normalizations of the 1D projections, all for 3 different scales).

Method identification. The following notation is used to identify the method, *ranking method* $\rightarrow$ *re-ranking method* (*number of re-ranked images*). Usage of average query expansion using n top images is denoted by QEn.

Method	P@20	Method	P@20
EI [7]	27.9	$\bar{\mathcal{S}}_{xy}$ $(5,2)$	57.9
Riemenschneider [28]	58.0	$\bar{\mathcal{S}}_{xy}$ $(6,3)$	61.4
SYM-FISH [6]	34.0	$\bar{\mathcal{S}}_{xy}$ $(5,2)$ + QE3	77.9
CS+GC [25]	49.3	$\bar{\mathcal{S}}_{xy}$ $(6,3)$ + QE3	**79.3**

Table 1. Performance comparison on the ETHZ extended shape dataset. Average precision at top 20 results is reported. We have not performed query mirroring for these results. The number of frequencies $(|\Omega_x|, |\Omega_\phi|)$ used is reported next to our methods.

Method	mAP	Method	mAP
GF-HOG [18]	12.2	$\bar{\mathcal{S}}_x + \bar{\mathcal{S}}_y \rightarrow \bar{\mathcal{S}}_{xy}$ (1k)	26.7
SHELO [29]	12.3	$\bar{\mathcal{S}}_x + \bar{\mathcal{S}}_y \rightarrow \bar{\mathcal{S}}_{xy}$ (5k)	29.2
LKS [30]	24.5	$\bar{\mathcal{S}}_{xy}$	30.4
GF-HOG [4]	18.2	$\bar{\mathcal{S}}_{xy}$ + QE3	**57.9**

Table 2. Performance comparison via mean Average Precision on the Flickr15k dataset.

6.1. Datasets and evaluation protocol

Constructing large scale ground-truth for sketch-based retrieval systems is not as easy as for traditional retrieval. One reason is the inherent abstraction of sketches. Moreover, positive images should not only comprised images of the same object/category, but also images depicting shapes similar to that of the query. Ground-truth at large scale should be on per query basis and this is not easy to achieve.

We initially evaluate our method on two image collections that are accompanied with ground-truth. These are the ETHZ extended shape dataset [31] and the Flickr15k dataset [18]. They consist of 285 images with 7 queries and 15K images with 330 queries (30 categories), respectively.

We further perform experiments on the large-scale dataset by Parui and Mittal [25] comprised 1.2M images and 175 queries, which has no available annotation. External annotators have manually evaluated the top results.

6.2. Evaluation and comparisons

Performance versus dimensionality. We construct the proposed sketch descriptor using our LDFM-based multiple kernel approximation and using the Fourier-based one. We compare performance for varying number of frequencies and present results in Figure 5 (left). The two methods have roughly the same performance for large number of frequencies where the kernel approximation is relatively good for both cases. The Fourier-based method significantly harms the performance for low number of frequencies due to its bad approximation. The orientation kernel is well approximated with few frequencies due to its wide shape (larger σ). We finally set $|\Omega_x| = 5$ and $|\Omega_\phi| = 2$ for the rest of our experiments, except if otherwise stated. Sketch descriptor $\mathbf{V}(\mathcal{P})$ has 243 dimensions, while $\mathbf{V}_x(\mathcal{P})$ only 27.

Ranking method. We compare ranking of the database with $\bar{\mathcal{S}}_{xy}$ and the projection-based approaches $\bar{\mathcal{S}}_x$ and $\bar{\mathcal{S}}_y$. In the latter case, only the top-ranked images are re-ranked by $\bar{\mathcal{S}}_{xy}$ to evaluate the performance loss. Results are shown in Figure 5 (middle). Ranking with sum of $\bar{\mathcal{S}}_y$ and $\bar{\mathcal{S}}_x$ appears significantly better than their individual use, while re-ranking one third of the database already recovers the performance loss. Speeding-up the ranking by $\bar{\mathcal{S}}_> \xrightarrow{+} \bar{\mathcal{S}}_<$, while in the end we re-rank 1k images, achieves mAP equal to 26.8. The drop is insignificant compared to the 26.9 in Figure 5 when re-ranking 1k images. Always ranking first with x or y projection, instead of our query dependent approach, gives 25.8 and 26.0 respectively.

Method	Dim	Time	DB	P@5	@10	@25	@50
$\bar{\mathcal{S}}_{xy}$ (1.2M)~~APM~~ [38]	(8,3)	55.4	15.3	43.2	40.9	37.2	33.8
$\bar{\mathcal{S}}_{xy}$ (1.2M)~~APM~~ [38]	(5,2)	20.2	3.3	25.8	24.7	22.5	20.2
$\bar{\mathcal{S}}_{xy}$ (1.2M)	(8,3)	55.4	5.1	50.1	46.7	42.0	37.2
$\bar{\mathcal{S}}_{xy}$ (1.2M)	(5,2)	20.2	1.1	45.8	44.1	38.5	35.4
$\bar{\mathcal{S}}_x + \bar{\mathcal{S}}_y \rightarrow \bar{\mathcal{S}}_{xy\star}$ (50k)	(6,3)	3.5	2.8	49.7	47.4	41.3	36.8
$\bar{\mathcal{S}}_> \xrightarrow{+} \bar{\mathcal{S}}_< \rightarrow \bar{\mathcal{S}}_{xy\star}$ (50k)	(6,3)	2.5	2.8	49.6	47.3	41.0	36.6
$\bar{\mathcal{S}}_> \xrightarrow{+} \bar{\mathcal{S}}_< \rightarrow \bar{\mathcal{S}}_{xy\star}$ (50k)†	(6,3)	2.5	0.7	50.3	47.3	41.5	36.7
$\bar{\mathcal{S}}_x + \bar{\mathcal{S}}_y \rightarrow \bar{\mathcal{S}}_{xy\star}$ (50k)	(5,2)	2.5	1.1	45.8	44.2	38.4	35.3
$\bar{\mathcal{S}}_> \xrightarrow{+} \bar{\mathcal{S}}_< \rightarrow \bar{\mathcal{S}}_{xy\star}$ (50k)	(5,2)	1.7	1.1	45.7	44.2	38.3	35.1
$\bar{\mathcal{S}}_> \xrightarrow{+} \bar{\mathcal{S}}_< \rightarrow \bar{\mathcal{S}}_{xy\star}$ (50k)†	(5,2)	1.7	0.3	45.6	43.5	38.0	35.0
$\bar{\mathcal{S}}_> \xrightarrow{+} \bar{\mathcal{S}}_< \rightarrow \bar{\mathcal{S}}_{xy\star}$ (50k)†+QE3	(6,3)	2.7	0.8	55.2	57.4	57.4	57.5
$\bar{\mathcal{S}}_> \xrightarrow{+} \bar{\mathcal{S}}_< \rightarrow \bar{\mathcal{S}}_{xy\star}$ (50k)†+QE10	(6,3)	2.7	0.8	63.0	63.4	64.8	65.2
$\bar{\mathcal{S}}_> \xrightarrow{+} \bar{\mathcal{S}}_< \rightarrow \bar{\mathcal{S}}_{xy\star}$ (50k)†+QE3	(5,2)	1.9	0.4	50.9	52.2	52.5	52.4
$\bar{\mathcal{S}}_> \xrightarrow{+} \bar{\mathcal{S}}_< \rightarrow \bar{\mathcal{S}}_{xy\star}$ (50k)†+QE10	(5,2)	1.9	0.4	56.4	56.8	57.3	57.8

Table 3. Performance, query time (in seconds) and database (DB) memory (in GB) requirements comparison on the 1.2M image dataset [25]. We report precision at n top ranked images (P@n). The number of frequencies $(|\Omega_x|, |\Omega_\phi|)$ is reported, which defines the final dimensionality (Dim = 1125, 605 or 243). ~~APM~~: Asymmetric feature maps are not used. †: Vector components uniformly quantized into 1 byte.

Approximations. We perform re-ranking based on $\bar{\mathcal{S}}_{xy}$ and its two approximations. We use approximation $\bar{\mathcal{S}}_{xy\star}$ to efficiently search over all translations and scales, while we finally refine the translation of maximum similarity. On the other hand, $\bar{\mathcal{S}}_{x/y}$ is used to refine $(\hat{x}_1, \hat{y}_1)$ and acts only on the best scale found by the ranking method. In some cases the ranking method misses the correct scale and this is the main reason for the performance difference between the two. Results are shown in Figure 5 (right).

Comparisons to other methods. Comparison of our method to other methods is reported in Table 1 for the ETHZ extended shape dataset and in Table 2 for the Flickr15k dataset. The scores achieved without the QE are the highest reported on both benchmarks. The QE gives additional significant boost in the performance. On Flickr15k we remarkably outperform the previous state of the art by 24 points of mAP.

Large scale evaluation. We evaluate our method at large scale with the 1.2M dataset [25]. For each query, only top-ranked images are annotated as either negative, positive or similar. Images marked as similar are images of similar shape but different category than the query. Retrieval examples are shown in Figure 6 and performance comparison is presented in Table 3. We measure precision at top-ranked images per query and report average precision on top ranked

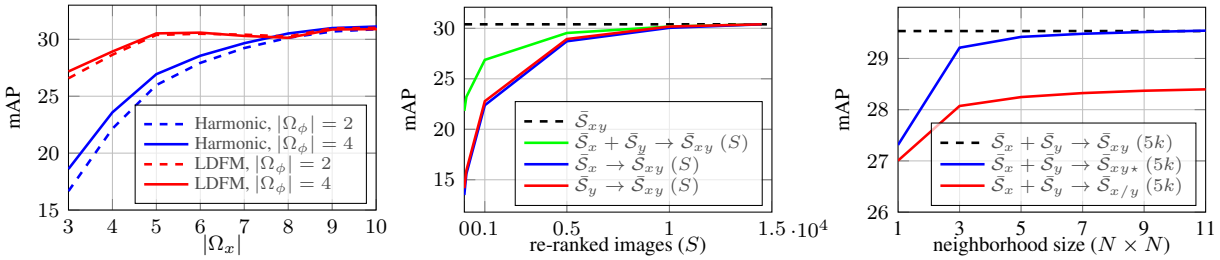


Figure 5. Performance comparison by measuring mean Average Precision (mAP) on the Flickr15k dataset. **Left**: Performance for increasing number of frequencies. Comparison between the Fourier-based approach [39] that uses harmonic frequencies and our joint optimization of the 3 kernel functions. Ranking is performed with $\bar{\mathcal{S}}_{xy}$. **Middle**: Comparison between the proposed methods for ranking the whole dataset. Re-ranking is additionally performed with $\bar{\mathcal{S}}_{xy}$ in all cases. We show mAP versus the number of re-ranked images. $S = 0$ signifies no re-ranking. **Right**: Performance of approximate re-ranking methods for increasing size of local refinement neighborhood. We show mAP versus the neighborhood size, while re-ranking 5k images.

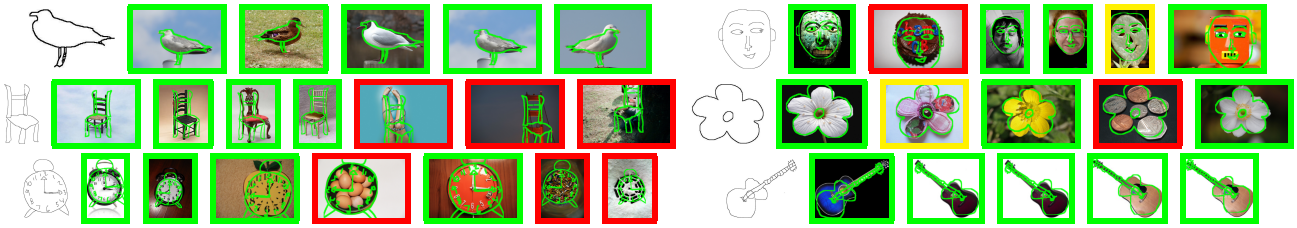


Figure 6. Examples of top-ranked retrieval images on the 1.2M dataset using our method. Localization of the sketch is shown in green color. Image borders denote positive (green), negative (red) and similar (yellow) image.

images over all queries.

We additionally evaluate performance when applying the trigonometric polynomial of Tolias *et al.* [38] to rank all database images. The proposed method by construction requires less memory and is significantly faster. It is also shown to perform better. The memory footprint is significantly decreased due to the asymmetry of our representation and due to good performance achieved with few frequencies. Encoding each vector component with 1 byte instead of single precision does not harm the performance.

Note that the discriminative-projection-first method only slightly decreases the performance, while it decreases the initial ranking time by 40%. Moreover, re-ranking only top 50k images performs with insignificant losses compared to ranking all images with the 2D polynomial. Finally, query expansion significantly improves the results. CNN descriptors are encoded with product quantization [19][4].

Query timings. The execution time was measured on the 1.2M image dataset using a single threaded MATLAB/Mex implementation on a 3.5GHz desktop machine. The results are summarized in Table 3. For $|\Omega_x| = 5$ and $|\Omega_\phi| = 2$, a query takes on average 1.81s for the initial ranking with $\bar{\mathcal{S}}_x + \bar{\mathcal{S}}_y$ and 0.72s for the top 50k re-ranking with $\bar{\mathcal{S}}_{xy\star}$ (with 3x3 neighborhood), giving a total time of 2.5s. Using $\bar{\mathcal{S}}_> \overset{+}{\rightarrow} \bar{\mathcal{S}}_<$, and computing the second projection only on $3 \cdot 50$k top-ranked images, ranking time drops to 1.05s. The values are independent of query complexity. The re-ranking using binary $\bar{\mathcal{S}}_{xy\star}$ is 17% faster compared to full $\bar{\mathcal{S}}_{xy}$. Applying the trigonometric polynomial scoring for ranking all images with the method of Tolias *et al.* [38] takes 20s, one order of magnitude slower than ours, for a low performance setup, while 55s with higher dimensionality and better performance which is still lower than ours.

The performance comparison to the work of Parui and Mittal [25] is not possible on the 1.2M dataset, as they use their own category-level ground truth, which is not publicly available. The comparison in terms of memory footprint (6.5GB is reported[25]) and execution time (1-5 sec per query is reported[25]) is favorable for the proposed method.

7. Conclusions

We have introduced a novel concept of asymmetric (explicit) feature maps. AFM allow to evaluate multiple kernels between a query and database entries with no additional memory requirements. The feature maps are optimally constructed by a joint kernel approximation, which turns out to be crucial for the accuracy. We have introduced a method of efficient approximation of scoring by trigonometric polynomials through 1D projections, which are a special case of asymmetric feature maps.

We have demonstrated the benefits of AFM on sketch-based image retrieval with short codes. We achieve state-of-the-art performance on a number of standard benchmarks. Compared with previous approaches using trigonometric polynomials [38], the proposed method achieves an order of magnitude speed-up, multiple-fold reduction in data storage, while improving the retrieval accuracy at the same time. The performance is further boosted by image-based average query expansion combined with AFM for object outline localization.

[4] After evaluation we discovered that the dataset contains a small amount of training ImageNet images, which can potentially affect with QE by CNN descriptors. Preliminary tests show that it affects insignificantly.

References

[1] Local features: State of the art, open problems and performance evaluation. http://www.iis.ee.ic.ac.uk/ComputerVision/DescrWorkshop/. 2

[2] A. C. Berg, T. L. Berg, and J. Malik. Shape matching and object recognition using low distortion correspondences. In *CVPR*, 2005. 1

[3] L. Bo and C. Sminchisescu. Efficient match kernel between sets of features for visual recognition. In *NIPS*, Dec. 2009. 1, 2

[4] T. Bui and J. Collomosse. Scalable sketch-based image retrieval using color gradient features. In *ICCV*, 2015. 2, 7

[5] A. Bursuc, G. Tolias, and H. Jégou. Kernel local descriptors with implicit rotation matching. In *ICMR*, 2015. 1, 2

[6] X. Cao, H. Zhang, S. Liu, X. Guo, and L. Lin. Sym-fish: A symmetry-aware flip invariant sketch histogram shape descriptor. In *ICCV*. IEEE, 2013. 2, 7

[7] Y. Cao, C. Wang, L. Zhang, and L. Zhang. Edgel index for large-scale sketch-based image search. In *CVPR*. IEEE, 2011. 1, 2, 7

[8] A. Chalechale, G. Naghdy, and A. Mertins. Sketch-based image matching using angular partitioning. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 35(1):28–41, 2005. 1, 2

[9] O. Chum. Low dimensional explicit feature maps. In *ICCV*, 2015. 2, 3, 4

[10] O. Chum, J. Philbin, J. Sivic, M. Isard, and A. Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *ICCV*, 2007. 6

[11] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *ECCV Workshop Statistical Learning in Computer Vision*, May 2004. 2

[12] Q. Danfeng, S. Gammeter, L. Bossard, T. Quack, and L. V. Gool. Hello neighbor: Accurate object retrieval with k-reciprocal nearest neighbors. In *CVPR*, 2011. 6

[13] P. Dollár and C. L. Zitnick. Structured forests for fast edge detection. In *ICCV*, 2013. 6

[14] M. Eitz, J. Hays, and M. Alexa. How do humans sketch objects? *ACM Transactions on Graphics*, 2012. 2

[15] M. Eitz, K. Hildebrand, T. Boubekeur, and M. Alexa. An evaluation of descriptors for large-scale image retrieval from sketched feature lines. *Computers & Graphics*, 34(5):482–498, 2010. 2

[16] M. Eitz, K. Hildebrand, T. Boubekeur, and M. Alexa. Sketch-based image retrieval: Benchmark and bag-of-features descriptors. *Visualization and Computer Graphics, IEEE Transactions on*, 17(11):1624–1636, 2011. 1, 2

[17] V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid. Groups of adjacent contour segments for object detection. *Trans. PAMI*, 30(1):36–51, 2008. 1

[18] R. Hu and J. Collomosse. A performance evaluation of gradient field hog descriptor for sketch based image retrieval. *CVIU*, 117(7):790–806, 2013. 2, 7

[19] H. Jégou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *Trans. PAMI*, 33(1):117–128, Jan. 2011. 6, 8

[20] H. Jégou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *CVPR*, 2010. 2

[21] A. Joly and O. Buisson. Logo retrieval with a contrario visual query expansion. In *ACM Multimedia*, Oct. 2009. 6

[22] Y. Kalantidis, C. Mellina, and S. Osindero. Cross-dimensional weighting for aggregated deep convolutional features. In *ECCVW*, 2016. 6

[23] I. Kokkinos and A. Yuille. Inference and learning with hierarchical shape models. *IJCV*, 93(2):201–225, 2011. 1

[24] C. Ma, X. Yang, C. Zhang, X. Ruan, M.-H. Yang, and O. Coporation. Sketch retrieval via dense stroke features. In *BMVC*, 2013. 2

[25] S. Parui and A. Mittal. Similarity-invariant sketch-based image retrieval in large databases. In *ECCV*, pages 398–414. Springer, 2014. 2, 7, 8

[26] F. Perronnin and C. R. Dance. Fisher kernels on visual vocabularies for image categorization. In *CVPR*, 2007. 2

[27] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *NIPS*, 2007. 1, 3

[28] H. Riemenschneider, M. Donoser, and H. Bischof. Image retrieval by shape-focused sketching of objects. In *Computer Vision Winter Workshop*, 2011. 2, 7

[29] J. M. Saavedra. Sketch based image retrieval using a soft computation of the histogram of edge local orientations (s-helo). In *ICIP*, 2014. 2, 7

[30] J. M. Saavedra, J. M. Barrios, and S. Orand. Sketch based image retrieval using learned keyshapes (lks). In *BMVC*, 2015. 1, 2, 7

[31] K. Schindler and D. Suter. Object detection by global contour shape. *Pattern Recognition*, 41(12):3736–3748, 2008. 7

[32] B. Scholkopf and A. Smola. *Learning with Kernels*. MIT Press, 2002. 3

[33] A. Shrivastava, T. Malisiewicz, A. Gupta, and A. A. Efros. Data-driven visual similarity for cross-domain image matching. *ACM Transactions on Graphics*, 30(6):154, 2011. 2

[34] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2014. 6

[35] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *ICCV*, 2003. 2

[36] X. Sun, C. Wang, C. Xu, and L. Zhang. Indexing billions of images for sketch-based retrieval. In *ACM Multimedia*, 2013. 1, 2

[37] A. Thayananthan, B. Stenger, P. H. Torr, and R. Cipolla. Shape context and chamfer matching in cluttered scenes. In *CVPR*, 2003. 2

[38] G. Tolias, A. Bursuc, T. Furon, and H. Jégou. Rotation and translation covariant match kernels for image retrieval. *CVIU*, 140:9–20, 2015. 1, 2, 3, 4, 5, 7, 8

[39] A. Vedaldi and A. Zisserman. Efficient additive kernels via explicit feature maps. *Trans. PAMI*, 34(3):480–492, Mar. 2012. 1, 2, 3, 8

[40] Q. Yu, Y. Yang, Y.-Z. Song, T. Xiang, and T. M. Hospedales. Sketch-a-net that beats humans. In *BMVC*, 2015. 2

XIII Efficient contour match kernel

Title: Efficient contour match kernel
Authors: G. Tolias, O. Chum
Published at: Image and Vision Computing

Efficient Contour Match Kernel

Giorgos Tolias*, Ondřej Chum

Visual Recognition Group, Faculty of Electrical Engineering, Czech Technical University in Prague

Abstract

We propose a novel concept of asymmetric feature maps (AFM), which allows to evaluate multiple kernels between a query and database entries without increasing the memory requirements. To demonstrate the advantages of the AFM method, we derive an efficient contour match kernel – short vector image representation that, due to asymmetric feature maps, supports efficient scale and translation invariant sketch-based image retrieval. Unlike most of the short-code based retrieval systems, the proposed method provides the query localization in the retrieved image. The efficiency of the search is boosted by approximating a 2D translation search via trigonometric polynomial of scores by 1D projections. The projections are a special case of AFM. An order of magnitude speed-up is achieved compared to traditional trigonometric polynomials. The results are boosted by an image-based average query expansion approach and, without any learning, significantly outperform the state-of-the-art hand-crafted descriptors on standard benchmarks. Our method competes well with recent CNN-based approaches that require large amounts of labeled sketches, images and sketch-image pairs.

Keywords: sketch-based image retrieval, efficient contour matching, kernel descriptors, asymmetric feature maps

1. Introduction

Efficient match kernel [1] is a popular choice in applications evaluating complex similarity measures on large collections of objects, where an object is a set of elements. This includes local feature descriptors [1, 2, 3] and image retrieval with short descriptors [4] .

In efficient match kernel, all elements of the sets are mapped to a finite feature map [5, 6]. An inner product of the feature maps approximates evaluation of a specific kernel, defining similarity of the set elements. We propose an extension to this concept. In the asymmetric feature map, the query uses a different embedding than the database objects. The query embedding defines the kernel that is evaluated between the query and the database entries. Thus, multiple kernels can be evaluated while the memory requirements for the database remains the same (up to a scalar per kernel) as for a single kernel to be evaluated. The embeddings are obtained via joint kernel feature map optimization, which significantly improves the quality of kernel approximation for a fixed dimensionality of the feature map.

The application domain of AFM is wide, in particular many efficient match kernels benefit from AFM. We evaluate the AFM on a sketch-based retrieval application. Sketch-based retrieval has received less attention than image retrieval and still remains challenging. Instead of a real image, the query consists of an abstract binary sketch. This allows the user to quickly outline an object, *e.g.* by a finger on a tablet or smart phone, and search for relevant images (see Figure 1). The progress in this area has more or less followed the footsteps of traditional image retrieval. The first systems employed global descriptors [7]. Then, the Bag-of-Words paradigm with local descriptors and feature quantization [8, 9, 10] was adopted. Geometric constraints are also applied to filter out false positives [11].

Due to the absence of textural cues on the query side, the image representations are shape based. Bridging the representation gap between hand-drawn sketches and real images is one of the challenges making the task difficult. Matching

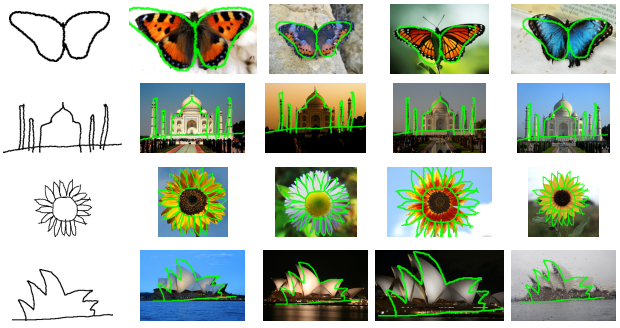


Figure 1: Scale and translation invariant contour matching. Examples of sketch queries which are localized (shown in green) in real images. The images are top-retrieved by the proposed efficient query-by-sketch image retrieval method.

*Corresponding author: giorgos.tolias@cmp.felk.cvut.cz, tel:+420-224-357-666

The authors were supported by the MSMT LL1303 ERC-CZ grant.

based on shape information has been addressed previously. For instance, in object recognition and detection [12, 8, 13], a costly online matching is performed, which prevented the methods to scale to large image collections. Recent methods manage to index million [14] to billion [15] images for sketch-based retrieval, at the cost of sacrificed invariance to geometric transformations.

We cast sketch-based image retrieval as efficient contour matching at large scale. In particular, we attempt to optimize the trade-off between scalability and invariance to geometric transformations. To demonstrate the impact of the AFM, we propose a short vector image representation that allows to index large image collections for sketch-based search. Our scale and translation invariant real-time search manages to process an order of millions of images per one processor thread in less than 2 seconds.

The AFM based method achieves state-of-the-art results on standard benchmarks. The method runs at speed comparable to previously published approaches tailored to sketch-based search. Compared with methods based on efficient match kernel [4], the proposed method achieves one order of magnitude speed-up. Unlike most of the methods using low-dimensional descriptors, the proposed method delivers localization of the object in both scale and space. The scale invariance is achieved by evaluating multiple kernels without the need to store multiple representations for database images. The translation invariance and object localization is provided by an efficient similarity evaluation on a 2D grid of translations.

In contrast to recent approaches that are based on Convolutional Neural Networks (CNN) for this task, our approach requires no learning. CNNs for this task [16, 17, 18] require large annotated datasets for both the domain of sketches and images, but also cross domain labeling. Finally, object localization is not possible with the learned embeddings.

Namely, the four main contributions of this work are as follows. (1) Asymmetric explicit feature maps allowing the use of multiple kernel functions without constructing multiple representations for database items are proposed. (2) A joint kernel approximation approach for multiple kernels is derived, generalizing a recent approach of low dimensional explicit feature maps (LDFM) [19]. (3) The scoring through trigonometric polynomial introduced in [4] is further extended and a significant speed-up of its evaluation is proposed. (4) State-of-the-art sketch-based image retrieval based on the AFM, which is further boosted by query expansion which acts, not on the edge maps as standard sketch matching, but on the original images.

The rest of paper is organized as follows, the main extensions to the conference version [20] of the work are pointed out. Related work is discussed in Section 2 and the necessary background is presented in Section 3. Section 4 describes our contributions on asymmetric explicit feature maps, which now includes detailed description of the joint kernel approximation (Section 4.2) casted as a linear program and its extension to handle continuous frequencies (Section 4.3). Analysis of the sketch retrieval application is given in Section 5, which now additionally provides detailed formulation of the 2D trigonometric polynomials. The retrieval procedure and the experimental evaluation are given in Section 6.

2. Related work

The most similar work to ours in terms of the application of the efficient match kernel is the approach of Tolias *et al.* [4], where the trigonometric polynomial scores were introduced in the context of image retrieval (see Section 3.3 for technical details). Shape properties of local features, such as dominant orientation or position, are jointly encoded with the SIFT descriptor. Despite initially assuming aligned objects, their kernel descriptor comes with an efficient way to compute similarity over multiple image transformations. Compared to their method, asymmetric feature maps introduced in our paper: i) reduce the memory requirements of multi-scale search by roughly a factor of 3, and ii) achieve an order of magnitude speed-up through approximate translation search. The trigonometric polynomials have been also used by Bursuc *et al.* [2] in the context of rotation invariant feature descriptors. This descriptor has recently shown competitive results with CNN based approaches [21], and its extension [3] surpasses CNN local descriptors.

Since we demonstrate the advantages of AFM on sketch based retrieval, we also provide a brief review of relevant literature on this topic, while prior work on explicit feature maps is discussed in Section 3.

Hand-crafted approaches. Object recognition is traditionally tackled with local image patches; a choice shown successful for object categories where local appearance is discriminative. However, several object categories are defined by the object shape. There exist approaches that focus on contour features to cope with this [8]. Costly on-line matching algorithms [12, 13] are effectively applied for object recognition, but this is not equivalently possible for large scale problems, as in the case of sketch-based image retrieval.

The line of research that focuses on sketches includes recognition [22, 23] or retrieval [24] of sketches. This paper addresses sketch-based image retrieval, which tries to match sketch queries to real images from a large collection. Following successful examples of traditional image retrieval, sketch-based methods describe images by either global representation [7, 25] or by local descriptors and the Bag-of-Words model. In the latter case, representative methods employ local descriptors that are traditionally

used on images [9, 10] or proposed particularly for this task [26, 27, 28, 29]. Some examples are HOG descriptors which are adapted for sketch retrieval [28] and were recently extended to capture color [30], symmetry-aware and flip invariant descriptors [29], and descriptors based on local contour fragments [27]. Generic approaches performing learning of discriminative features have been also shown effective for sketch retrieval [31].

Chamfer matching appears to be a good similarity measure for object shapes [32]. Recent attempts focus on Chamfer matching approximations in order to increase scalability. Cao *et al.* [14] binarize the distance transform map and manage to index two million images. However, their approach completely lacks invariance. The same holds for the work of Sun *et al.* [15] who increase the scale of the indexed collection up to one billion. Despite the achievement of scalability, rough approximations of Chamfer matching sacrifice accuracy. Recently, Parui and Mittal [11] proposed a similarity invariant approach able to index up to one million images. Their solution is based on dynamic programming to match chains of contour lines, while the main drawback is the costly off-line indexing. On the contrary to the aforementioned approaches, our approach provides invariance at large scale while indexing reduces to generating short vector representation per image.

CNN-based approaches are typically trained for a fixed set of object categories [33, 18, 17, 16]. This makes the learned representation to be category specific and may limit the performance on new categories. For specific tasks, in order to obtain high accuracy, a different model per category needs to be trained [18, 34, 35]. The outcome of these learning based methods is an embedding of an image or sketch to the same high dimensional space. Such embeddings are not expected to provide any localization ability and simple restrict to global matching.

A common requirement is to obtain labeled data in both domains, while the training involves various different stages. These vary from training on images or sketches to siamese training with image-sketch pairs that are labeled at category or sub-category level [17].

3. Background

We briefly review the necessary background, which includes efficient match kernels [1], explicit feature maps [6] and efficient trigonometric polynomial scores [4].

3.1. Efficient Match Kernels

In many situations, an object is described by a set of measurements $\mathcal{P} = \{p \in \mathbb{R}^d\}$. Employing a mapping $\Psi : \mathbb{R}^d \to \mathbb{R}^D$ to the elements of $\mathcal{P}$, the set representation of efficient match kernels is defined as

$$\mathbf{V}(\mathcal{P}) = \sum_{p \in \mathcal{P}} \Psi(p). \tag{1}$$

Then, a dot product between the set representation yields the similarity between sets

$$\mathcal{S}(\mathcal{P}, \mathcal{Q}) = \mathbf{V}(\mathcal{P})^\top \mathbf{V}(\mathcal{Q}) = \sum_{p \in \mathcal{P}} \sum_{p \in \mathcal{Q}} \Psi(p)^\top \Psi(q). \tag{2}$$

Normalized similarity is computed by cosine similarity [4], *i.e.*, dot product of ℓ_2 normalized vectors,

$$\bar{\mathcal{S}}(\mathcal{P}, \mathcal{Q}) = \frac{\mathbf{V}(\mathcal{P})^\top \mathbf{V}(\mathcal{Q})}{\sqrt{\mathbf{V}(\mathcal{P})^\top \mathbf{V}(\mathcal{P})}\sqrt{\mathbf{V}(\mathcal{Q})^\top \mathbf{V}(\mathcal{Q})}}, \tag{3}$$

while another choice is to normalize by the set cardinality [1]. Herein, the cosine similarity is adopted ensuring self-similarity is normalized to one. A number of image representations, such as BOW [36, 37], Fisher vectors [38], or VLAD [39], can be interpreted as efficient match kernels.

3.2. Explicit feature maps

Let $K(p, q)$ be a one-dimensional (p is now scalar) positive definite stationary kernel [40] $K : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$. The value of a stationary kernel by definition depends only on the difference $\lambda = p - q$,

$$K(p, q) = K(p, p - \lambda) = k(\lambda), \tag{4}$$

where $k(\lambda)$ is a signature of kernel $K(p, q)$. Due to Bochner's theorem, kernel signature k can be written as

$$k(\lambda) = \int_0^\infty \alpha(\omega) \cos(\omega\lambda)\, \mathrm{d}\omega, \tag{5}$$

where $\alpha(\omega) : \mathbb{R}_0^+ \to \mathbb{R}_0^+$. The kernel signature is approximated by sum over a finite set Ω of frequencies

$$\hat{k}(\lambda) \approx \sum_{\omega \in \Omega} \alpha_\omega \cos(\omega\lambda), \tag{6}$$

where $\alpha_\omega \in \mathbb{R}_0^+$. Applying the trigonometric identity

$$\cos(p - q) = \cos(p)\cos(q) + \sin(p)\sin(q) \tag{7}$$

gives rise to feature map (or feature embedding) $\Psi_\omega : \mathbb{R} \to \mathbb{R}^2$ defined as

$$\Psi_\omega(p) = \left(\sqrt{\alpha_\omega}\cos(\omega p),\ \sqrt{\alpha_\omega}\sin(\omega p)\right)^\top. \tag{8}$$

The inner product of two such vectors reconstructs the terms of equation (6) since $\Psi_\omega(p)^\top \Psi_\omega(q) = \alpha_\omega \cos(\omega(p - q))$. Let the feature map $\Psi(p) : \mathbb{R} \to \mathbb{R}^D$ be constructed as a concatenation of $\Psi_\omega(p)$ for all $\omega \in \Omega$. Now, the inner product

$$\Psi(p)^\top \Psi(q) = \hat{k}(p - q) \approx K(p, q) \tag{9}$$

evaluates the approximation of the kernel signature (6) and hence approximates the original kernel K. The choice of the number of frequencies $|\Omega|$ determines the quality of the approximation and the dimensionality of the embedding. The dimensionality is $2|\Omega|$, or $2|\Omega| - 1$ if $0 \in \Omega$[1].

[1] If $0 \in \Omega$, then $\alpha_0 \sin(0\lambda) = 0$ for all λ can be dropped from the explicit feature map.

Feature map construction. We mention in detail (and compare) two approaches to construct the explicit feature maps. We do not consider random feature maps [5], which approximate the integral in (5) using Monte-Carlo methods. Such feature maps provide a poor approximation for low-dimensional feature maps.

Vedaldi and Zisserman [6] propose the following approximation to a kernel signature $k(\lambda)$ on an interval $\lambda \in [-\Lambda, \Lambda]$. First, a periodic function g with period 2Λ is constructed, so that $g(\lambda) = k(\lambda)$ for $\lambda \in [-\Lambda, \Lambda]$. The feature map is then efficiently obtained by approximating periodic g using harmonic frequencies only. This approach has been shown sub-optimal [19]. Further, the periodic function g is not even guaranteed to be positive definite.

A convex optimization approach is proposed by Chum [19, 41]. The input domain of $\hat{k}(\lambda)$ is discretized to finite set $Z \subset [0, \Lambda]$. The quality of the approximation is measured at points in Z as, for example, an ℓ_∞ norm

$$C_\infty(k, \hat{k}) = \max_{\lambda \in Z} |k(\lambda) - \hat{k}(\lambda)|. \tag{10}$$

The set of frequencies $\Omega \subset \bar{\Omega}$ are selected from a pool of frequencies $\bar{\Omega}$, and corresponding weights $\alpha_\omega \geq 0$, $\omega \in \bar{\Omega}$ jointly through a solution of a linear program

$$\min_k C(k, \hat{k}) + \gamma \textstyle\sum_{\omega \in \bar{\Omega}} \alpha_\omega, \tag{11}$$

where $\gamma \in \mathbb{R}^+$ is a weight on the l_1 regularizer controlling the trade-off between the quality of the approximation and the sparsity of α_ω. This is the method we adopt and extend in this work.

3.3. Alignment using trigonometric polynomials

Tolias *et al.* [4] propose an image representation derived by efficient match kernels and explicit feature maps. We focus on the case that all measurements of set $\mathcal{P}$ are shifted by a constant value Δp; note that measurements p are now scalars. The similarity under such shift forms a trigonometric polynomial

$$\mathcal{S}(\mathcal{P}_{\Delta p}, \mathcal{Q}) = \sum_{\omega \in \Omega} \left(\beta_\omega \cos(\omega \Delta p) + \gamma_\omega \sin(\omega \Delta p)\right), \tag{12}$$

with $\mathcal{P}_{\Delta p} = \{p - \Delta p, p \in \mathcal{P}\}$. Parameters β_ω and γ_ω are given by dot products of relevant sub-vectors of $\mathbf{V}(\mathcal{P})$ and $\mathbf{V}(\mathcal{Q})$. Finally the similarity measure that is invariant under such shifting is given by $\mathcal{S}_1(\mathcal{P}_{\Delta p}, \mathcal{Q}) = \max_{\Delta p} \mathcal{S}(\mathcal{P}_{\Delta p}, \mathcal{Q})$.

We postpone further analysis of polynomials of scores until the image representation is introduced in Section 5.

4. Asymmetric feature maps

In this section, we introduce the concept of asymmetric feature maps and describe our extension to LDFM for joint approximation of multiple kernels.

4.1. Asymmetric representation

Unlike in classical explicit feature maps, a different feature map $\hat{\Psi}$ is used on the query side and a different one $\hat{\Psi}'$ is used on the database side. We show that with asymmetric feature maps, a number of different kernels can be efficiently evaluated between query and database vectors while keeping the database storage of fixed size. Compare the feature map in equation (8) to the following feature maps for the query and database side respectively

$$\hat{\Psi}_\omega(q) = \left(\alpha_\omega \cos(\omega q),\ \alpha_\omega \sin(\omega q)\right)^\top = \sqrt{\alpha_\omega}\Psi_\omega \tag{13}$$

$$\hat{\Psi}'_\omega(p) = \left(\cos(\omega p),\ \sin(\omega p)\right)^\top = \Psi_\omega / \sqrt{\alpha_\omega}. \tag{14}$$

The inner products $\hat{\Psi}(q)^\top \hat{\Psi}'(p) = \Psi(q)^\top \Psi(p)$ are preserved. The kernel function is fully defined by the weights on the query side. No additional storage is required on the database side to evaluate the kernel. The same holds for efficient match kernels, as (1) is a normalized sum of feature maps. To evaluate the cosine similarity (3), only a single scalar per kernel $K^{(i)}$ needs to be stored for each database entry $\mathcal{P}$ – the ℓ_2 norm $\sqrt{\mathbf{V}^{(i)}(\mathcal{P})^\top \mathbf{V}^{(i)}(\mathcal{P})}$, which is computed offline.

4.2. Joint approximation of multiple kernels.

In order to evaluate a number of different kernels $K^{(i)}(p, q)$ using the asymmetric feature maps, all respective explicit feature maps $\Psi^{(i)}$ have to be based on the same set of frequencies Ω with non-zero weights $\alpha_\omega^{(i)}$. That is

$$K^{(i)}(p, q) \approx \sum_{\omega \in \Omega} \alpha_\omega^{(i)} \cos\left(\omega(p - q)\right). \tag{15}$$

The problem is to find a set $\Omega \subset \bar{\Omega}$ of frequencies that is common for all kernels $K^{(i)}$, and sets of weights $\alpha_\omega^{(i)}$ that are specific for each kernel $K^{(i)}$, so that all the kernels are approximated well.

A naive approach would be to optimize the set of frequencies for one of the kernels and keep it fixed for other kernels. This approach, however, leads to poor approximation, as shown in Figure 2. We propose an extension to LDFM [19] to jointly approximate a set of kernels $K^{(i)}$ represented by their respective kernel signatures $k^{(i)}$, $i \in \{1 \dots n\}$. The quality of the approximation is measured by the sum of individual qualities (10)

$$C^*_\infty = \sum_{i=1}^{n} C_\infty(k^{(i)}, \hat{k}^{(i)}) = \sum_{i=1}^{n} \max_{\lambda \in Z} |k^{(i)}(\lambda) - \hat{k}^{(i)}(\lambda)|.$$

The optimization is performed by executing a linear program

$$\min_{\alpha_\omega^{(i)} |\ \omega \in \bar{\Omega}} C^*_\infty + \gamma \sum_{\omega \in \bar{\Omega}} \max_i \alpha_\omega^{(i)}, \tag{16}$$

where γ is a weight of the sparsity regularizer that controls the number of frequencies used, *i.e.* the dimensionality of

the feature map. The regularizer in (16) is a relaxation of ℓ_0 norm regularizer $\sum_{\omega\in\bar{\Omega}} D_\omega$, where

$$D_\omega = \begin{cases} 0 & \forall i : \alpha_\omega^{(i)} = 0 \\ 1 & \text{otherwise.} \end{cases}$$

Following the approach of Chum [19], to ensure the required dimensionality of the feature map, a binary search for γ is performed.

Figure 2 presents the approximation of three different kernels using the same set of frequencies. We compare the approximation using only harmonic frequencies, the naive approximation mentioned above, and our joint approximation. The latter has a significantly better fit.

4.3. Continuous frequencies

While selection of the frequencies from a fixed pool of frequencies $\bar{\Omega}$ is a straightforward application of LDFM, the iterative adjustment of the "active" frequencies in Ω, that is additionally performed in LDFM ([19] Section 4.2), requires further derivation.

The $t+1$ iteration updates each frequency $\omega^t \in \Omega^t$ by a small step $d_{\omega^{t+1}}$ to $\omega^{t+1} = \omega^t + d_{\omega^{t+1}}$. Each kernel signature $k^{(i)}$ at iteration $t+1$ (we drop the iteration index $t+1$ of k for the sake of clarity) is then represented as

$$\begin{aligned} k^{(i)}(\lambda) &= \sum_{\omega^{t+1}\in\Omega^{t+1}} \alpha_{\omega^{t+1}}^{(i)} \cos(\omega^{t+1}\lambda) \\ &= \sum_{\omega^t\in\Omega^t} \alpha_{\omega^{t+1}}^{(i)} \cos((\omega^t + d_{\omega^{t+1}})\lambda), \end{aligned} \quad (17)$$

where

$$|d_{\omega^{t+1}}| \le d^{\max}. \quad (18)$$

The first order Taylor expansion of the cosine function in the frequency variable ω^t (not in λ) reads

$$\cos((\omega^t + d_{\omega^{t+1}})\lambda) \approx \cos(\omega^t\lambda) - d_{\omega^{t+1}}\lambda\sin(\omega^t\lambda). \quad (19)$$

Such an approximation is good only in a small neighbourhood of ω^t, which is controlled by the size $d^{\max}$ of the"trust region" (18). By substituting (19) into (17), we obtain

$$\begin{aligned} k^{(i)}(\lambda) &\approx \sum_{\omega^t\in\Omega^t} \alpha_{\omega^{t+1}}^{(i)} \cos(\omega^t\lambda) \\ &\quad - \sum_{\omega^t\in\Omega^t} d_{\omega^{t+1}} \alpha_{\omega^{t+1}}^{(i)} \lambda\sin(\omega^t\lambda). \end{aligned} \quad (20)$$

The unknown variable $d_{\omega^{t+1}}$ now appears in multiple bi-linear terms $d_{\omega^{t+1}}\alpha_{\omega^{t+1}}^{(i)}$ with unknown variables $\alpha_{\omega^{t+1}}^{(i)}$. Unlike in [19], the system cannot be directly linearized by introduction of an auxiliary variables for the bi-linear terms.

For each frequency ω^t, let us choose a pivot kernel function $k^{(p)}$, so that $\alpha_{\omega^t}^{(p)} > 0$. To eliminate multiple dependent bi-linear terms, we fix the ratios of weights

$$\frac{\alpha_{\omega^{t+1}}^{(i)}}{\alpha_{\omega^{t+1}}^{(p)}} = \frac{\alpha_{\omega^t}^{(i)}}{\alpha_{\omega^t}^{(p)}} \quad \text{by imposing} \quad \alpha_{\omega^{t+1}}^{(i)} = \alpha_{\omega^{t+1}}^{(p)} \frac{\alpha_{\omega^t}^{(i)}}{\alpha_{\omega^t}^{(p)}}. \quad (21)$$

Let an auxiliary variable $\beta_{\omega^{t+1}}$ be

$$\beta_{\omega^{t+1}} = d_{\omega^{t+1}}\, \alpha_{\omega^{t+1}}^{(p)} \quad (22)$$

By combining (21) and (22), equation (20) transforms into

$$\begin{aligned} k^{(i)}(\lambda) &\approx \sum_{\omega^t\in\Omega^t} \alpha_{\omega^{t+1}}^{(p)} \frac{\alpha_{\omega^t}^{(i)}}{\alpha_{\omega^t}^{(p)}} \cos(\omega^t\lambda) \\ &\quad - \sum_{\omega^t\in\Omega^t} \beta_{\omega^{t+1}} \frac{\alpha_{\omega^t}^{(i)}}{\alpha_{\omega^t}^{(p)}} \lambda\sin(\omega^t\lambda). \end{aligned} \quad (23)$$

Equation (23) is linear in all unknown variables $\alpha_{\omega^{t+1}}^{(p)}$ and $\beta_{\omega^{t+1}}$. The constraint (18) on variables $d_{\omega^{t+1}}$ transforms to

$$|\beta_{\omega^{t+1}}| \le \alpha_{\omega^{t+1}}^{(p)} d^{\max}. \quad (24)$$

Both (23) and (24) are in a form that can be written as a linear program finding optimal values of $\beta_{\omega^{t+1}}$ and $\alpha_{\omega^{t+1}}^{(p)}$. Compared to the original formulation, $|\Omega|$ variables $\beta_{\omega^{t+1}}$, and $2|\Omega|$ constraints (24) were introduced to the linear program.

After executing the linear program, the frequencies are updated as

$$\omega^{t+1} = \omega^t + \frac{\beta_{\omega^{t+1}}}{\alpha_{\omega^{t+1}}^{(p)}}.$$

The weights $\alpha_{\omega^{t+1}}^{(i)}$ are then computed optimally for each kernel signature $k^{(i)}$ by a linear program with fixed frequencies optimizing $C_\infty(k^{(i)} - k^{(i)})$, which removes the imposed fixed ratio on the weights (21).

5. Sketch-Based Retrieval

In this section we present our sketch descriptor employing explicit feature maps and elaborate on the efficient trigonometric polynomial of scores to further approximate it. Our methodology is presented for the symmetric feature maps, while the asymmetric case is equivalent. We finally present efficient ways to perform the initial ranking and reranking for sketch-based image retrieval.

5.1. Sketch descriptor

Consider a binary sketch as a set of contour points, that is a set of pixels $\mathcal{P}$ that lie on the contour. A *contour pixel* $p \in \mathcal{P}$ is represented as $p = (p_x, p_y, p_\phi, p_w)$, where p_x and p_y are 2D image coordinates, p_ϕ is the gradient angle (or orientation) of the contour at (p_x, p_y), and p_w is a strength

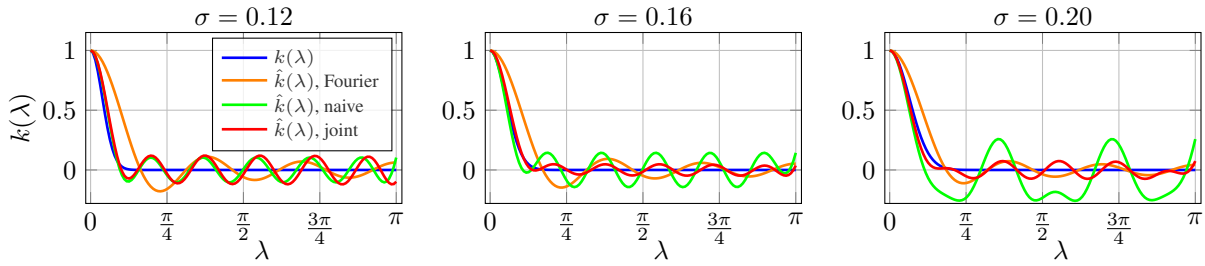


Figure 2: Approximation comparison of multiple 1D RBF kernels (with different σ) using the same set of frequencies. We show approximation using harmonic frequencies only, a naive approach of optimizing the leftmost kernel and using the same frequencies for all, and our joint approximation. Maximum value is normalized to one such that the errors are comparable. $|\Omega| = 7$ for all approximations.

of the gradient. For real images, the contour parameters are obtained form an edge detector. For sketches, $p_w = 1$ is set for all contour pixels.

The similarity between contour pixels is computed using a multiplicative kernel composed of three one-dimensional kernels, spatial kernels over p_x, p_y, and an orientation kernel over p_ϕ. The 1D stationary kernels are denoted $K_x(p_x, q_x) = k_x(\lambda_x)$, $K_y(p_y, q_y) = k_y(\lambda_y)$, and $K_\phi(p_\phi, q_\phi) = k_\phi(\lambda_\phi)$ respectively. The *sketch descriptor* is a weighted sum of contour pixel feature maps[2]

$$\mathbf{V}(\mathcal{P}) = \sum_{p \in \mathcal{P}} p_w \Psi(p_x) \otimes \Psi(p_y) \otimes \Psi(p_\phi). \tag{25}$$

It is easy to show that *sketch similarity* (2) becomes

$$\mathcal{S}(\mathcal{P}, \mathcal{Q}) = \sum_{p \in \mathcal{P}} \sum_{q \in \mathcal{Q}} p_w q_w k_x(\lambda_x) k_y(\lambda_y) k_\phi(\lambda_\phi). \tag{26}$$

The orientation and spatial kernels are implemented by 1D RBF kernels with parameters σ_ϕ and $\sigma_x = \sigma_y$, respectively. The set of frequencies are denoted by Ω_ϕ and $\Omega_x = \Omega_y$, while the dimensionality of the corresponding embeddings is $D_x = 2|\Omega_x| - 1$ and $D_\phi = 2|\Omega_\phi| - 1$, respectively. Note that frequency $\omega = 0$ is always included. The sketch descriptor has dimensionality ${D_x}^2 D_\phi$.

The proposed representation constitutes a holistic representation encoding the global sketch shape. We now define a representation encoding only one of the spatial coordinates along with the orientation. It is equivalent to the projection of contour pixels on the horizontal/vertical image axis. The sketch descriptor derived by projection on the horizontal axis is given by

$$\mathbf{V}_x(\mathcal{P}) = \sum_{p \in \mathcal{P}} p_w \Psi(p_x) \otimes 1 \otimes \Psi(p_\phi), \tag{27}$$

where the $\otimes 1$ can be omitted and is only used to show, that the x-projection is a sub-vector of (25) and hence a special case of the proposed asymmetric feature map. This stems from the presence of the constant component of the feature map for y, corresponding to $0 \in \Omega_y$. An analogous derivation holds for $\mathbf{V}_y(\mathcal{P})$ and vertical projection.

[2]We use Ψ to denote both the spatial and orientation feature map and simplify the notation. In fact, $\Psi(p_x)$ and $\Psi(p_y)$ approximate the spatial kernels k_x and k_y, respectively, which are identical, while $\Psi(p_\phi)$ the orientation kernel k_ϕ.

5.2. Position alignment

The sketch descriptor encodes spatial coordinates and orientation of contour pixels. Therefore, alignment of objects is assumed, *i.e.* centered and up-right objects. Such an assumption does not hold in real image collections and introduces significant limitations. We now detail the polynomial of scores (mentioned in Section 3) proposed by Tolias *et al.* [4]. We show that translation invariance is achieved by polynomial of scores, and that its evaluation can be efficiently approximated to speed up the search process.

One dimensional. Consider the x-projected sketch descriptor $\mathbf{V}_x(\mathcal{P})$. Let $\mathcal{P}_{\Delta x}$ be the shifted version sketch $\mathcal{P}$ where all contour pixels are horizontally translated by Δx. Elementary trigonometric identities allow us to show that

$$\begin{aligned}\Psi^c_\omega(x - \Delta x) &= \Psi^c_\omega(x)\cos(\omega\Delta x) + \Psi^s_\omega(x)\sin(\omega\Delta x)\\ \Psi^s_\omega(x - \Delta x) &= \Psi^s_\omega(x)\cos(\omega\Delta x) - \Psi^c_\omega(x)\sin(\omega\Delta x),\end{aligned} \tag{28}$$

where Ψ^c_ω and Ψ^s_ω denote the first and second dimension of Ψ_ω (8), respectively. Let $\mathbf{V}^c_\omega(\mathcal{P})$ be the sub-vector of $\mathbf{V}(\mathcal{P})$ comprised all elements that contain term $\Psi^c_\omega(x)$, and similarly for $\mathbf{V}^s_\omega(\mathcal{P})$. It turns out that the descriptor of the translated sketch is constructed from that of the original sketch

$$\begin{aligned}\mathbf{V}^c_\omega(\mathcal{P}_{\Delta x}) &= \mathbf{V}^c_\omega(\mathcal{P})\cos(\omega\Delta x) + \mathbf{V}^s_\omega(\mathcal{P})\sin(\omega\Delta x)\\ \mathbf{V}^s_\omega(\mathcal{P}_{\Delta x}) &= \mathbf{V}^s_\omega(\mathcal{P})\cos(\omega\Delta x) - \mathbf{V}^c_\omega(\mathcal{P})\sin(\omega\Delta x).\end{aligned} \tag{29}$$

The sketch similarity between sketches $\mathcal{P}$ and $\mathcal{Q}$ under horizontal translation Δx is a trigonometric polynomial

$$\mathcal{S}(\mathcal{P}_{\Delta x}, \mathcal{Q}) = \sum_{\omega \in \Omega_x} \left(\beta_\omega \cos(\omega\Delta x) + \gamma_\omega \sin(\omega\Delta x)\right), \tag{30}$$

with coefficients β_ω and γ_ω

$$\begin{aligned}\beta_\omega &= \mathbf{V}^c_\omega(\mathcal{P})^\top \mathbf{V}^c_\omega(\mathcal{Q}) + \mathbf{V}^s_\omega(\mathcal{P})^\top \mathbf{V}^s_\omega(\mathcal{Q})\\ \gamma_\omega &= \mathbf{V}^s_\omega(\mathcal{P})^\top \mathbf{V}^c_\omega(\mathcal{Q}) - \mathbf{V}^c_\omega(\mathcal{P})^\top \mathbf{V}^s_\omega(\mathcal{Q}).\end{aligned} \tag{31}$$

The coefficients β_ω and γ_ω of this polynomial are computed by two products of sub-vectors with D_ϕ dimensions. In total there are $N_1 = D_x$ coefficients to be computed. Finally, similarity for any translation with (30) has cost equal to N_1 scalar multiplications. If the candidate translations

are fixed, then terms $\cos(\omega\Delta x)$ and $\sin(\omega\Delta x)$ can be precomputed. Normalized similarity comes at no extra cost since the ℓ_2 norm of sketch descriptor remains constant under translations (k_x is a stationary kernel):

$$\mathbf{V}(\mathcal{P}_{\Delta x})^\top \mathbf{V}(\mathcal{P}_{\Delta x}) = \mathbf{V}(\mathcal{P})^\top \mathbf{V}(\mathcal{P}). \tag{32}$$

Similarity that is invariant to horizontal translation is computed by maximizing (30) for all possible translations

$$\mathcal{S}_x(\mathcal{P}_{\Delta x}, \mathcal{Q}) = \max_{\Delta x} \mathcal{S}(\mathcal{P}_{\Delta x}, \mathcal{Q}). \tag{33}$$

Note that this similarity is also invariant to vertical translation as y coordinate is not encoded at all. However, this makes the representation less discriminative. The actual *sketch transformation* aligning the two shapes is given by $\hat{x}_1 = \arg\max_{\Delta x} \mathcal{S}(\mathcal{P}_{\Delta x}, \mathcal{Q})$. Similarity based on the vertical projection is defined in a similar way.

Two dimensional. Consider the full 2D translation $(\Delta x, \Delta y)$. The corresponding second order trigonometric polynomial [4] of scores $\mathcal{S}(\mathcal{P}_{\Delta x,\Delta y}, Q)$ is constructed similarly to the first order one. It allows alignment for 2D translations and in this section we detail its derivation. Descriptor $\mathbf{V}(\mathcal{P})$ encodes both spatial coordinates x and y and full 2D translation $(\Delta x, \Delta y)$ is considered.

The embedding of a measurement x under shifting equal to Δx, which corresponds to horizontal translation by Δx, is given by

$$\begin{aligned}\Psi^c_{\omega_x}(x-\Delta x) &= \Psi^c_{\omega_x}(x)\cos(\omega_x\Delta x) + \Psi^s_{\omega_x}(x)\sin(\omega_x\Delta x)\\ \Psi^s_{\omega_x}(x-\Delta x) &= \Psi^s_{\omega_x}(x)\cos(\omega_x\Delta x) - \Psi^c_{\omega_x}(x)\sin(\omega_x\Delta x),\end{aligned} \tag{34}$$

while the corresponding embedding of y and vertical translation by Δy is given by

$$\begin{aligned}\Psi^c_{\omega_y}(y-\Delta y) &= \Psi^c_{\omega_y}(y)\cos(\omega_y\Delta y) + \Psi^s_{\omega_y}(y)\sin(\omega_y\Delta y)\\ \Psi^s_{\omega_y}(y-\Delta y) &= \Psi^s_{\omega_y}(y)\cos(\omega_y\Delta y) - \Psi^c_{\omega_y}(y)\sin(\omega_y\Delta y),\end{aligned} \tag{35}$$

where $\Psi^c_{\omega_x}$ and $\Psi^s_{\omega_x}$ denote the first and second dimension of Ψ_{ω_x}, and similarly for y. The embedding encoding both coordinates x, y has dimensionality equal to 4, as a result of a Kronecker product. We denote these 4 dimensions by $\Psi^{cc}_{\omega_x,\omega_y}$, $\Psi^{cs}_{\omega_x,\omega_y}$, $\Psi^{sc}_{\omega_x,\omega_y}$, and $\Psi^{ss}_{\omega_x,\omega_y}$. Under 2D translation equal to $(\Delta x, \Delta y)$ these 4 dimensions of the embedding are given by (36), (37), (38) and (39), respectively.

$$\begin{aligned}\Psi^{cc}_{\omega_x,\omega_y}(x-\Delta x, y-\Delta y) = \;& \Psi^c_{\omega_x}(x)\Psi^c_{\omega_y}(y)\cos(\omega_x\Delta x)\cos(\omega_y\Delta y)\\ &+\Psi^c_{\omega_x}(x)\Psi^s_{\omega_y}(y)\cos(\omega_x\Delta x)\sin(\omega_y\Delta y)\\ &+\Psi^s_{\omega_x}(x)\Psi^c_{\omega_y}(y)\sin(\omega_x\Delta x)\cos(\omega_y\Delta y)\\ &+\Psi^s_{\omega_x}(x)\Psi^s_{\omega_y}(y)\sin(\omega_x\Delta x)\sin(\omega_y\Delta y)\end{aligned} \tag{36}$$

$$\begin{aligned}\Psi^{cs}_{\omega_x,\omega_y}(x-\Delta x, y-\Delta y) = \;& \Psi^c_{\omega_x}(x)\Psi^s_{\omega_y}(y)\cos(\omega_x\Delta x)\cos(\omega_y\Delta y)\\ &-\Psi^c_{\omega_x}(x)\Psi^c_{\omega_y}(y)\cos(\omega_x\Delta x)\sin(\omega_y\Delta y)\\ &+\Psi^s_{\omega_x}(x)\Psi^s_{\omega_y}(y)\sin(\omega_x\Delta x)\cos(\omega_y\Delta y)\\ &-\Psi^s_{\omega_x}(x)\Psi^c_{\omega_y}(y)\sin(\omega_x\Delta x)\sin(\omega_y\Delta y)\end{aligned} \tag{37}$$

$$\begin{aligned}\Psi^{sc}_{\omega_x,\omega_y}(x-\Delta x, y-\Delta y) = \;& \Psi^s_{\omega_x}(x)\Psi^c_{\omega_y}(y)\cos(\omega_x\Delta x)\cos(\omega_y\Delta y)\\ &+\Psi^s_{\omega_x}(x)\Psi^s_{\omega_y}(y)\cos(\omega_x\Delta x)\sin(\omega_y\Delta y)\\ &-\Psi^c_{\omega_x}(x)\Psi^c_{\omega_y}(y)\sin(\omega_x\Delta x)\cos(\omega_y\Delta y)\\ &-\Psi^c_{\omega_x}(x)\Psi^s_{\omega_y}(y)\sin(\omega_x\Delta x)\sin(\omega_y\Delta y)\end{aligned} \tag{38}$$

$$\begin{aligned}\Psi^{ss}_{\omega_x,\omega_y}(x-\Delta x, y-\Delta y) = \;& \Psi^s_{\omega_x}(x)\Psi^s_{\omega_y}(y)\cos(\omega_x\Delta x)\cos(\omega_y\Delta y)\\ &-\Psi^s_{\omega_x}(x)\Psi^c_{\omega_y}(y)\cos(\omega_x\Delta x)\sin(\omega_y\Delta y)\\ &-\Psi^c_{\omega_x}(x)\Psi^s_{\omega_y}(y)\sin(\omega_x\Delta x)\cos(\omega_y\Delta y)\\ &+\Psi^c_{\omega_x}(x)\Psi^c_{\omega_y}(y)\sin(\omega_x\Delta x)\sin(\omega_y\Delta y)\end{aligned} \tag{39}$$

We observe that, similarly to the 1D translation, the embedding under 2D translation is constructed by applying component-wise operations on the original embedding (embedding without any translation). It is also easy to show that the same stands after aggregation of multiple embeddings and construction of descriptor $\mathbf{V}(\mathcal{P}_{\Delta x,\Delta y})$. This is the descriptor of set $\mathcal{P}$ when all its elements undergo 2D translation and comprises 4 parts per frequency pair ω_x, ω_y with the first one given by

$$\begin{aligned}\mathbf{V}^{cc}_{\omega_x,\omega_y}(\mathcal{P}_{\Delta x,\Delta y}) = \;& \mathbf{V}^{cc}_{\omega_x,\omega_y}(\mathcal{P})\cos(\omega_x\Delta x)\cos(\omega_y\Delta y)\\ &+\mathbf{V}^{cs}_{\omega_x,\omega_y}(\mathcal{P})\cos(\omega_x\Delta x)\sin(\omega_y\Delta y)\\ &+\mathbf{V}^{sc}_{\omega_x,\omega_y}(\mathcal{P})\sin(\omega_x\Delta x)\cos(\omega_y\Delta y)\\ &+\mathbf{V}^{ss}_{\omega_x,\omega_y}(\mathcal{P})\sin(\omega_x\Delta x)\sin(\omega_y\Delta y),\end{aligned} \tag{40}$$

where $\mathbf{V}^{cc}_{\omega_x,\omega_y}(\mathcal{P})$ is given by

$$\mathbf{V}^{cc}_{\omega_x,\omega_y}(\mathcal{P}) = \sum_{\mathcal{P}} p_w \Psi^c_{\omega_x}(x)\Psi^c_{\omega_y}(y) \otimes \Psi(\phi) \tag{41}$$

and constitutes the sub-vector associated with frequencies ω_x, ω_y. It corresponds to the first (cc) dimension of the embedding. The dimensionality of this sub-vector is equal to D_ϕ. We obtain the other 3 parts of the descriptor in the same fashion.

Now, sketch similarity under 2D translation forms the trigonometric polynomial

$$\begin{aligned}\mathcal{S}(\mathcal{P}_{\Delta x,\Delta y},Q) = &\sum_{\omega_x\in\Omega_x}\sum_{\omega_y\in\Omega_y}\mu^{cc}_{\omega_x,\omega_y}\cos(\omega_x\Delta x)\cos(\omega_y\Delta y)\\ &+\sum_{\omega_x\in\Omega_x}\sum_{\omega_y\in\Omega_y}\mu^{cs}_{\omega_x,\omega_y}\cos(\omega_x\Delta x)\sin(\omega_y\Delta y)\\ &+\sum_{\omega_x\in\Omega_x}\sum_{\omega_y\in\Omega_y}\mu^{sc}_{\omega_x,\omega_y}\sin(\omega_x\Delta x)\cos(\omega_y\Delta y)\\ &+\sum_{\omega_x\in\Omega_x}\sum_{\omega_y\in\Omega_y}\mu^{ss}_{\omega_x,\omega_y}\sin(\omega_x\Delta x)\sin(\omega_y\Delta y).\end{aligned} \tag{42}$$

The coefficients $\mu^{cc}_{\omega_x,\omega_y}$ are computed as

$$\begin{aligned}\mu^{cc}_{\omega_x,\omega_y} =&\mathbf{V}^{cc}_{\omega_x,\omega_y}(\mathcal{P})^\top\mathbf{V}^{cc}_{\omega_x,\omega_y}(\mathcal{Q})\\ &+\mathbf{V}^{cs}_{\omega_x,\omega_y}(\mathcal{P})^\top\mathbf{V}^{cs}_{\omega_x,\omega_y}(\mathcal{Q})\\ &+\mathbf{V}^{sc}_{\omega_x,\omega_y}(\mathcal{P})^\top\mathbf{V}^{sc}_{\omega_x,\omega_y}(\mathcal{Q})\\ &+\mathbf{V}^{ss}_{\omega_x,\omega_y}(\mathcal{P})^\top\mathbf{V}^{ss}_{\omega_x,\omega_y}(\mathcal{Q}),\end{aligned} \tag{43}$$

while the rest of the coefficients $\mu^{cs}_{\omega_x,\omega_y}$, $\mu^{sc}_{\omega_x,\omega_y}$ and $\mu^{ss}_{\omega_x,\omega_y}$ are similarly computed with the corresponding signs given according to (37),(38), and (39).

The polynomial allows exhaustive template matching (*i.e.* sliding window) in a very efficient way. The cost to compute one of its coefficients is $4D_\phi$. There are $N_2 = 4(|\Omega_x|-1)^2 + 4(|\Omega_x|-1)+1$ non-zero coefficients in total. Give the coefficients, the similarity computation for a single 2D translation has cost equal to N_2 scalar multiplications. Translation invariant similarity is given by $\mathcal{S}_{xy}(\mathcal{P}_{\Delta x,\Delta y},\mathcal{Q}) = \max_{(\Delta x,\Delta y)}\mathcal{S}(\mathcal{P}_{\Delta x,\Delta y},\mathcal{Q})$, and the transformation aligning the two shapes is given by $(\hat{x}_2,\hat{y}_2) = \arg\max_{(\Delta x,\Delta y)}\mathcal{S}(\mathcal{P}_{\Delta x,\Delta y},\mathcal{Q})$.

In Figures 4 and 3 we present an alignment example between a sketch and a real image. Similarity is computed based on the horizontal and vertical projections, while also for the 2D case. Maximum similarity is met at translations that align the two silhouettes. Interestingly, the 2D transformation composed by the independent projections $(\hat{x}_1,\hat{y}_1)$ is not far from that of the full 2D estimation $(\hat{x}_2,\hat{y}_2)$.

5.3. Efficient retrieval and query expansion

Herein, we propose three methods how to avoid exhaustive evaluation of $\mathcal{S}(\mathcal{P}_{\Delta x,\Delta y},\mathcal{Q})$. First method efficiently selects a shortlist of images on which the score $\bar{\mathcal{S}}_{xy}$ is computed. The other two methods are designed to limit the number of possible translations over which $\mathcal{S}(\mathcal{P}_{\Delta x,\Delta y},\mathcal{Q})$ is evaluated to obtain a good approximation of $\bar{\mathcal{S}}_{xy}$.

Shortlist by projections. The similarities $\bar{\mathcal{S}}_x$ and $\bar{\mathcal{S}}_y$ computed over the projections (30) provide an estimate of the $\bar{\mathcal{S}}_{xy}$. We propose to use this estimate for initial ranking and to compute the slow similarity $\bar{\mathcal{S}}_{xy}$ only on a shortlist of top S images. Experiments show that initial ranking by $\bar{\mathcal{S}}_x + \bar{\mathcal{S}}_y$ outperforms ranking that uses only one projection. To further speed-up the evaluation for large-scale collections, we propose *discriminative projection first* approach. In this method, one projection is computed over the whole dataset, creating a pre-shortlist of $3S$ images with the highest score. The second projection is only evaluated on this pre-shortlist. Now, the shortlist based on the value of $\bar{\mathcal{S}}_x + \bar{\mathcal{S}}_y$ is a sub-set of the pre-shortlist. The first projection used is query dependent, the one with higher variance in the relevant coordinate in the sketch query is used. Discriminative projection first is denoted as $\bar{\mathcal{S}}_> \xrightarrow{+} \bar{\mathcal{S}}_<$.

Re-ranking by local refinement. The 1D alignment provides, besides the scores, the scale and 1D translations $(\hat{x}_1,\hat{y}_1)$ maximizing the 1D projection scores, which often is a rough approximation of the full 2D alignment (see Figure 4 and 3). In this approach, the full similarity $\mathcal{S}(\mathcal{P}_{\Delta x,\Delta y},\mathcal{Q})$ is only evaluated evaluated for a small neighborhood of $(\hat{x}_1,\hat{y}_1)$ on a fixed 2D grid. Sketch similarity computed by this method is denoted by $\mathcal{S}_{x/y}$.

Re-ranking by binary polynomial. We efficiently approximate the second order polynomial by a corresponding one that has binary coefficients and variables (*i.e.* $\cos(\omega_x\Delta x)\cos(\omega_y\Delta y)$ is binarized). We simply binarize both by a sign function. The similarity approximation for 2D translation is given by dot product between binary vectors which is faster to compute. Translation maximizing the binary approximation is found, and $\mathcal{S}(\mathcal{P}_{\Delta x,\Delta y},\mathcal{Q})$ is only computed on a small neighborhood, as in the local refinement. Figure 3 shows an example where the position of the maximum on the 2D map of similarities for the binarized case remains close to that of the real valued one. Experiments show that the binary polynomials provide very good estimate of the translation. We denote this method by $\mathcal{S}_{xy\star}$.

Query expansion. Query Expansion (QE) is a standard approach to improve retrieval results by a new query that exploits the top-ranked results [42, 43, 44]. Unlike the original query, the QE is performed on image descriptors, the sketch descriptors are only used for localization. A global CNN image descriptor is used for QE, in particular off-the-shelf CroW [45] with VGG16 network [46]. The 512D image descriptor extracted per database image is compressed using product quantization [47] into 64 bytes. A basic version of an Average Query Expansion (AQE) [42] is used. CroW descriptors of the top results are averaged and a query is issued.

6. Experiments

We briefly summarize the design choices of the indexing and search procedure of our sketch-based retrieval. Then, we evaluate our method and compare to the state of the art.

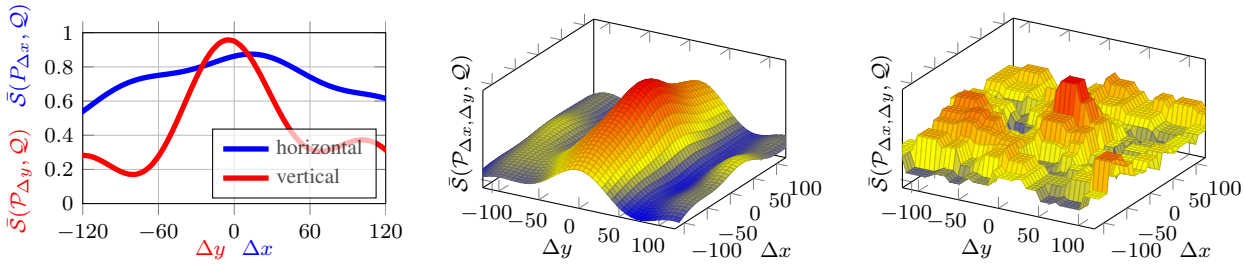


Figure 3: Alignment results for the example in Figure 4. Similarity as a function of translation (in pixels): independent 1D projections (left), the full 2D translation (middle), and the 2D binarized polynomial (right). At zero translation the centers of the sketch and the image are aligned. The detection in magenta color (Figure 4) is based on the similarity shown at the left, while the one in green is based on that shown in the middle.

Indexing (offline stage). All database images are down-sampled to have the longer side equal to 400 pixels. The edges are detected by off-the-shelf detector of Dollár and Zitnick [48]. The output edge strength is used as p_w, while all edges with strengths lower than 0.2 are completely discarded. A single sketch descriptor per database image is computed with AFM (14). Three kernels are used to search at three scales. Finally, the corresponding ℓ_2 norms for normalizing similarity (3) are computed and stored.

Query (online stage). The sketch query is cropped with a tight bounding box and resized similarly to database images. Two additional scales are given by down-sampling to 80% and 60%. Different query scales need to be matched with different kernels; smaller scale is matched with narrower kernel. The kernels shown in Figure 2 are used accordingly. The orientation kernel has $\sigma_\phi = 0.8$. One query descriptor per kernel is constructed (13). Additionally, each query is also horizontally mirrored.

The translations to be evaluated are fixed in a uniform way. Maximum translation is set to 80 pixels towards both directions and the step is 20. These are used for the maximum query size, while for different scales the maximum translation (step) is increased (decreased) linearly according to the relative query scale. That means, the localization is finer for smaller scales. Similarity is computed per scale independently and maximum similarity is kept.

The descriptor dimensionality is given by the number of frequencies $|\Omega_x|$ and $|\Omega_\phi|$. For instance, a compact setting of $|\Omega_x| = 5$ and $|\Omega_\phi| = 2$ leads to a 243D descriptor, while a high-performance settings of $|\Omega_x| = 6$ and $|\Omega_\phi| = 3$ leads to a 605D descriptor. In all cases, 9 additional scalars per image are stored (normalization of the 2D descriptor, normalizations of the 1D projections, all for 3 different scales).

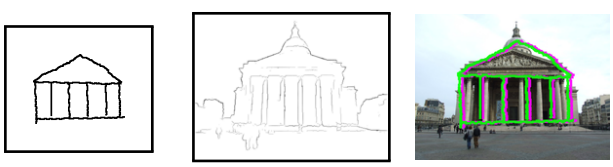


Figure 4: Sketch (left) and the edge map (middle) of a real image (right). We depict the translations maximizing similarity based on 1D projections (magenta) and the full 2D case (green).

Method identification. The following notation is used to identify the method, *ranking method* $\rightarrow$ *re-ranking method* (*number of re-ranked images*). Usage of average query expansion using n top images is denoted by QEn.

6.1. Datasets and evaluation protocol

Constructing large scale ground-truth for sketch-based retrieval systems is not as easy as for traditional retrieval. One reason is the inherent abstraction of sketches. Moreover, positive images should not only comprised images of the same object/category, but also images depicting shapes similar to that of the query. Ground-truth at large scale should be on per query basis and this is not easy to achieve.

We initially evaluate our method on two image collections that are accompanied with ground-truth. These are the ETHZ extended shape dataset [49] and the Flickr15k dataset [28]. They consist of 285 images with 7 queries and 15K images with 330 queries (30 categories), respectively.

We further perform experiments on the large-scale dataset by Parui and Mittal [11] comprised 1.2M images and 175 queries, which has no available annotation. External annotators have manually evaluated the top results.

6.2. Annotation process

In this section we describe the annotation process on the 1.2M image dataset used in our experiments. The dataset consists of 1.2M database images and comes with 175 sketch queries out of which 100 are acquired from the sketch dataset of Eitz *et al.* [22] and 75 are created for sketch retrieval. The query sketches belong to 40 categories defined in the original sketch collection [22]. The employed evaluation measure is precision at K top-ranked images. Precision is evaluated for each query.

We adopt the following protocol and perform evaluation at *particular instance level*. Our principle is closest to that of particular instance retrieval than to categorization. The annotation is conducted on per query basis. Results are annotated according to the particular query and its shape, not according to general categories. Finally, we report average

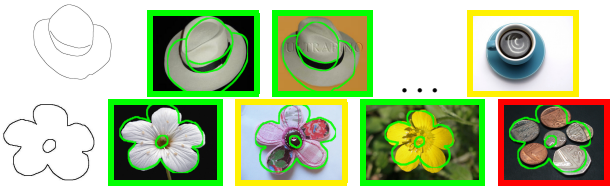


Figure 5: Retrieval example. Top: The sketch of a hat retrieving similar hats but also a coffee cup that is considered as similar. The sketch query is likely to intend the coffee cup too, their shapes match well. Bottom: The sketch of a flower that correctly retrieves flowers and a flower shaped object but also a set of coins whose overall exterior contour is close to a flower shape.

precision over all queries, and not over categories [11], as this does not favor categories with few queries and is more appropriate for future comparisons.

Our instructions to the annotators are summarized as follows. *Positive:* Both the semantics (object category) and the shape/pose of the query and database image are the same. *Negative:* Neither the semantics nor the shape are the same. *Similar:* We consider two different cases of similar images. (1) The object category is different, *i.e.* the object shown in the database image is different than the reported category of the query, but the shape is very similar. If there is some chance that the query sketch is intended to describe an image, then this image is considered similar. Note that sketch queries should be seen in an abstract way. (2) If the semantics are the same, then we relax the shape criterion to cover cases where some details of the object are missing. In this case, we require the shape to be more or less similar. We will share our partial annotation and rankings for future comparisons. Similar images are considered as positive for precision evaluation, however we find it useful to distinguish between positive and similar images for future needs and to show the differences while presenting examples.

We discuss some annotation examples of similar images in order to make our protocol more clear. For instance, the flower query of Figure 5 belongs to the flower category. The second retrieved image is not a real flower, but the shape matches perfectly. It is very likely that the user who drew this query intended to retrieve such an image. However, even though the 4^{th} retrieved image has a good match of shape, it is not likely that this was the user's intention. Drawing a sketch to target the coin images would definitely include the shape of the individual coins. Finally, the hat query shown in Figure 5 can be seen both as a hat and as a coffee cup with a plate. Therefore, the latter is considered as similar.

6.3. Evaluation and comparisons

Performance versus dimensionality. We construct the proposed sketch descriptor using our LDFM-based multiple kernel approximation and using the Fourier-based one. We compare performance for varying number of frequencies and present results in Figure 6 (left). The two methods have roughly the same performance for large number of frequencies where the kernel approximation is relatively good for both cases. The Fourier-based method significantly harms the performance for low number of frequencies due to its bad approximation. The orientation kernel is well approximated with few frequencies due to its wide shape (larger σ). We finally set $|\Omega_x| = 5$ and $|\Omega_\phi| = 2$ for the rest of our experiments, except if otherwise stated. Sketch descriptor $\mathbf{V}(\mathcal{P})$ has 243 dimensions, while $\mathbf{V}_x(\mathcal{P})$ only 27.

Method	P@20	Method	P@20
EI [14]	27.9	$\bar{\mathcal{S}}_{xy}$ $(5,2)$	57.9
Riemenschneider [27]	58.0	$\bar{\mathcal{S}}_{xy}$ $(6,3)$	61.4
SYM-FISH [29]	34.0	$\bar{\mathcal{S}}_{xy}$ $(5,2)$ + QE3	77.9
CS+GC [11]	49.3	$\bar{\mathcal{S}}_{xy}$ $(6,3)$ + QE3	79.3

Table 1: Performance comparison on the ETHZ extended shape dataset. Average precision at top 20 results is reported. We have not performed query mirroring for these results. The number of frequencies $(|\Omega_x|, |\Omega_\phi|)$ used is reported next to our methods.

Method	mAP
GF-HOG [28]	12.2
SHELO [25]	12.3
LKS [10]	24.5
GF-HOG [30]	18.2
$\bar{\mathcal{S}}_x + \bar{\mathcal{S}}_y \rightarrow \bar{\mathcal{S}}_{xy}$ (1k)	26.7
$\bar{\mathcal{S}}_x + \bar{\mathcal{S}}_y \rightarrow \bar{\mathcal{S}}_{xy}$ (5k)	29.2
$\bar{\mathcal{S}}_{xy}$	30.4
$\bar{\mathcal{S}}_{xy}$ + QE3	57.9

Table 2: Performance comparison via mean Average Precision on the Flickr15k dataset.

Ranking method. We compare ranking of the database with the full 2D matching $\bar{\mathcal{S}}_{xy}$ and the 1D projection-based approaches $\bar{\mathcal{S}}_x$ and $\bar{\mathcal{S}}_y$. In the latter case, only the top-ranked images are re-ranked by $\bar{\mathcal{S}}_{xy}$ to evaluate the per-

Method	Shoes		Chairs	
	acc.@1	acc.@10	acc.@1	acc.@10
BoW-HOG+rankSVM [18]	17.4	67.8	28.9	67.0
Dense-HOG+rankSVM [18]	24.4	65.2	52.6	93.8
Sketch-a-Net+rankSVM [18]	20.0	62.6	47.4	82.5
Shoes network [18]	39.1	87.8	–	–
Chairs network [18]	–	–	69.1	97.9
$\bar{\mathcal{S}}_{xy}$ $(5,2)$	29.6	81.7	53.6	87.6
$\bar{\mathcal{S}}_{xy}$ $(6,3)$	32.2	79.1	59.8	89.7

Table 3: Performance comparison via accuracy at rank K (acc.@K) with the state-of-the-art learning based methods on the Shoes/Chairs test datasets. Our method is the only one in this table that is hand-crafted and does not require any supervision.

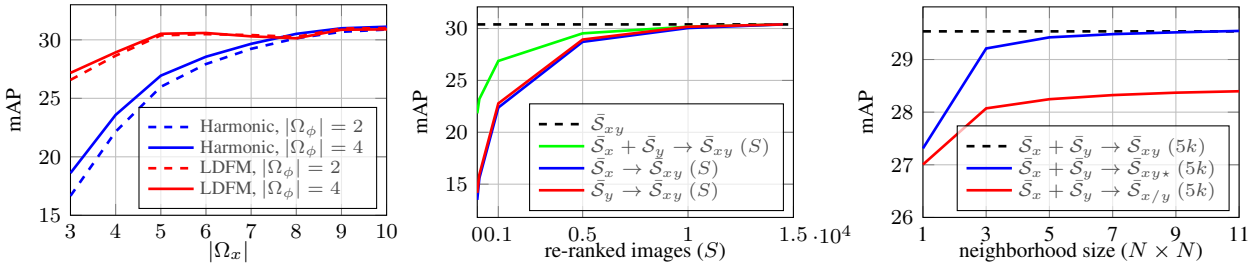


Figure 6: Performance comparison by measuring mean Average Precision (mAP) on the Flickr15k dataset. **Left**: Performance for increasing number of frequencies. Comparison between the Fourier-based approach [6] that uses harmonic frequencies and our joint optimization of the 3 kernel functions. Ranking is performed with $\bar{\mathcal{S}}_{xy}$. **Middle**: Comparison between the proposed methods for ranking the whole dataset. Re-ranking is additionally performed with $\bar{\mathcal{S}}_{xy}$ in all cases. We show mAP versus the number of re-ranked images. $S = 0$ signifies no re-ranking. **Right**: Performance of approximate re-ranking methods for increasing size of local refinement neighborhood. We show mAP versus the neighborhood size, while re-ranking 5k images.

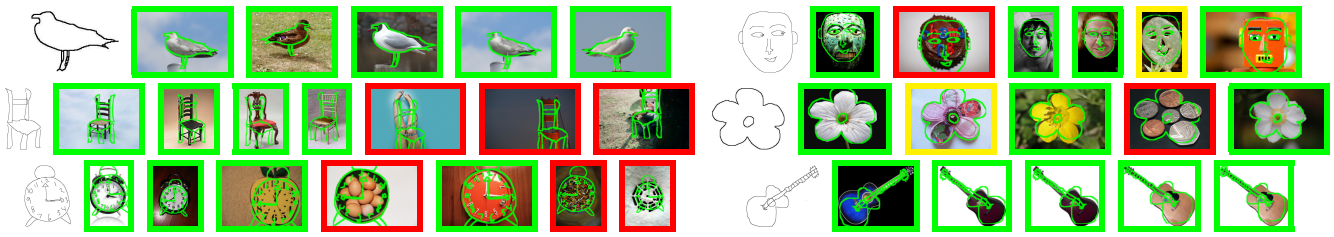


Figure 7: Examples of top-ranked retrieval images on the 1.2M dataset using our method. Localization of the sketch is shown in green color. Image borders denote positive (green), negative (red) and similar (yellow) image.

formance loss. Results are shown in Figure 6 (middle). Ranking with sum of $\bar{\mathcal{S}}_y$ and $\bar{\mathcal{S}}_x$ appears significantly better than their individual use, while re-ranking one third of the database already recovers the performance loss. Speeding-up the ranking by using the discriminative projection first with $\bar{\mathcal{S}}_> \xrightarrow{+} \bar{\mathcal{S}}_<$, while in the end we re-rank 1k images, achieves mAP equal to 26.8. The drop is insignificant compared to the 26.9 in Figure 6 when re-ranking 1k images. Always ranking first with x or y projection, instead of our query dependent approach, gives 25.8 and 26.0 respectively.

Approximations. We perform re-ranking based on $\bar{\mathcal{S}}_{xy}$ and its two approximations. We use the binary approximation $\bar{\mathcal{S}}_{xy\star}$ to efficiently search over all translations and scales, while we finally refine the translation of maximum similarity. On the other hand, the local refinement method $\bar{\mathcal{S}}_{x/y}$ is used to refine $(\hat{x}_1, \hat{y}_1)$ and acts only on the best scale found by the ranking method. In some cases the ranking method misses the correct scale and this is the main reason for the performance difference between the two. Results are shown in Figure 6 (right).

Comparisons with other methods. Comparison of our method with other methods is reported in Table 1 for the ETHZ extended shape dataset and in Table 2 for the Flickr15k dataset. The scores achieved without the QE are the highest reported on both benchmarks when there is no learning involved. The QE gives additional significant boost in the performance. On Flickr15k we remarkably outperform the previous state of the art by 24 points of mAP. Recent CNN-based methods achieve higher performance on the Flickr15k dataset. For instance, the Sketchy network [17] achieves 34.0 mAP with a descriptor of 1k dimensions. It outperforms our descriptor, however, it requires massive amount of training data, both in terms of annotated images and hand-drawn sketches for object categories and particular objects.

Method	Dim	Time	DB	P@5	@10	@25	@50
$\bar{\mathcal{S}}_{xy}$ (1.2M)~~AFM~~ [4]	(8,3)	55.4	15.3	43.2	40.9	37.2	33.8
$\bar{\mathcal{S}}_{xy}$ (1.2M)~~AFM~~ [4]	(5,2)	20.2	3.3	25.8	24.7	22.5	20.2
$\bar{\mathcal{S}}_{xy}$ (1.2M)	(8,3)	55.4	5.1	50.1	46.7	42.0	37.2
$\bar{\mathcal{S}}_{xy}$ (1.2M)	(5,2)	20.2	1.1	45.8	44.1	38.5	35.4
$\bar{\mathcal{S}}_x + \bar{\mathcal{S}}_y \rightarrow \bar{\mathcal{S}}_{xy\star}$ (50k)	(6,3)	3.5	2.8	49.7	47.4	41.3	36.8
$\bar{\mathcal{S}}_> \xrightarrow{+} \bar{\mathcal{S}}_< \rightarrow \bar{\mathcal{S}}_{xy\star}$ (50k)	(6,3)	2.5	2.8	49.6	47.3	41.0	36.6
$\bar{\mathcal{S}}_> \xrightarrow{+} \bar{\mathcal{S}}_< \rightarrow \bar{\mathcal{S}}_{xy\star}$ (50k)†	(6,3)	2.5	0.7	50.3	47.3	41.5	36.7
$\bar{\mathcal{S}}_x + \bar{\mathcal{S}}_y \rightarrow \bar{\mathcal{S}}_{xy\star}$ (50k)	(5,2)	2.5	1.1	45.8	44.2	38.4	35.3
$\bar{\mathcal{S}}_> \xrightarrow{+} \bar{\mathcal{S}}_< \rightarrow \bar{\mathcal{S}}_{xy\star}$ (50k)	(5,2)	1.7	1.1	45.7	44.2	38.3	35.1
$\bar{\mathcal{S}}_> \xrightarrow{+} \bar{\mathcal{S}}_< \rightarrow \bar{\mathcal{S}}_{xy\star}$ (50k)†	(5,2)	1.7	0.3	45.6	43.5	38.0	35.0
$\bar{\mathcal{S}}_> \xrightarrow{+} \bar{\mathcal{S}}_< \rightarrow \bar{\mathcal{S}}_{xy\star}$ (50k)†+QE3	(6,3)	2.7	0.8	55.2	57.4	57.4	57.5
$\bar{\mathcal{S}}_> \xrightarrow{+} \bar{\mathcal{S}}_< \rightarrow \bar{\mathcal{S}}_{xy\star}$ (50k)†+QE10	(6,3)	2.7	0.8	63.0	63.4	64.8	65.2
$\bar{\mathcal{S}}_> \xrightarrow{+} \bar{\mathcal{S}}_< \rightarrow \bar{\mathcal{S}}_{xy\star}$ (50k)†+QE3	(5,2)	1.9	0.4	50.9	52.2	52.5	52.4
$\bar{\mathcal{S}}_> \xrightarrow{+} \bar{\mathcal{S}}_< \rightarrow \bar{\mathcal{S}}_{xy\star}$ (50k)†+QE10	(5,2)	1.9	0.4	56.4	56.8	57.3	57.8

Table 4: Performance, query time (in seconds) and database (DB) memory (in GB) requirements comparison on the 1.2M image dataset [11]. We report precision at n top ranked images (P@n). The number of frequencies $(|\Omega_x|, |\Omega_\phi|)$ is reported, which defines the final dimensionality (Dim = 1125, 605 or 243). ~~AFM~~: Asymmetric feature maps are not used. †: Vector components uniformly quantized into 1 byte.

We additionally evaluate our method on a fine-grained dataset of shoes and chairs [18]. The goal is to retrieve the particular object that corresponds to the sketch query. We present results in Table 3. This task has been always handled with supervision and training data for each particular domain. Our method comes second and right after the network trained for each domain, while we outperform all other supervised approaches without requiring any supervision or training data.

Large scale evaluation. We evaluate our method at large scale with the 1.2M dataset [11]. For each query, only top-ranked images are annotated as either negative, positive or similar. Images marked as similar are images of similar shape but different category than the query. Retrieval examples are shown in Figure 7 and performance comparison is presented in Table 4. We measure precision at top-ranked images per query and report average precision on top ranked images over all queries.

We additionally evaluate performance when applying the trigonometric polynomial of Tolias *et al.* [4] to rank all database images. The proposed method by construction requires less memory and is significantly faster. It is also shown to perform better. The memory footprint is significantly decreased due to the asymmetry of our representation and due to good performance achieved with few frequencies. Encoding each vector component with 1 byte instead of single precision does not harm the performance.

Note that the discriminative-projection-first method only slightly decreases the performance, while it decreases the initial ranking time by 40%. Moreover, re-ranking only top 50k images performs with insignificant losses compared to ranking all images with the 2D polynomial. Finally, query expansion significantly improves the results. CNN descriptors are encoded with product quantization [47][3].

Retrieval examples. We present retrieval examples comparing our sketch descriptor alone and our sketch descriptor when followed by image-based average query expansion. Several query examples are shown in Figure 8, where top retrieved images are presented for both methods. The improvements are evident, even when the top 10 results comprise incorrect images too.

Query timings. The execution time was measured on the 1.2M image dataset using a single threaded MATLAB/Mex implementation on a 3.5GHz desktop machine. The results are summarized in Table 4. For $|\Omega_x| = 5$ and $|\Omega_\phi| = 2$, a query takes on average 1.81s for the initial ranking with $\bar{\mathcal{S}}_x + \bar{\mathcal{S}}_y$ and 0.72s for the top 50k re-ranking with $\bar{\mathcal{S}}_{xy\star}$ (with 3x3 neighborhood), giving a total time of 2.5s. Using $\bar{\mathcal{S}}_> \xrightarrow{+} \bar{\mathcal{S}}_<$, and computing the second projection only on $3 \cdot 50$k top-ranked images, ranking time drops to 1.05s. The values are independent of query complexity. The re-ranking using binary $\bar{\mathcal{S}}_{xy\star}$ is 17% faster compared to full $\bar{\mathcal{S}}_{xy}$. Applying the trigonometric polynomial scoring for ranking all images with the method of Tolias *et al.* [4] takes 20s, one order of magnitude slower than ours, for a low performance setup, while 55s with higher dimensionality and better performance which is still lower than ours.

[3]After evaluation we discovered that the dataset contains a small amount of training ImageNet images, which can potentially affect with QE by CNN descriptors. Preliminary tests show that it affects insignificantly.

The performance comparison to the work of Parui and Mittal [11] is not possible on the 1.2M dataset, as they use their own category-level ground truth, which is not publicly available. The comparison in terms of memory footprint (6.5GB is reported[11]) and execution time (1-5 sec per query is reported[11]) is favorable for the proposed method.

7. Conclusions

We have introduced a novel concept of asymmetric (explicit) feature maps. AFM allow to evaluate multiple kernels between a query and database entries with no additional memory requirements. The feature maps are optimally constructed by a joint kernel approximation, which turns out to be crucial for the accuracy. We have introduced a method of efficient approximation of scoring by trigonometric polynomials through 1D projections, which are a special case of asymmetric feature maps.

We have demonstrated the benefits of AFM on sketch-based image retrieval with short codes. The proposed method outperforms all learning-free methods on a number of standard benchmarks, and it well competes with learning-based approaches, including CNNs, that require large amounts of labeled sketches, images and sketch-image pairs. Compared with previous approaches using trigonometric polynomials [4], the proposed method achieves an order of magnitude speed-up, multiple-fold reduction in data storage, while improving the retrieval accuracy at the same time.

The performance is further boosted by image-based average query expansion combined with AFM for object outline localization.

References

[1] L. Bo, C. Sminchisescu, Efficient match kernel between sets of features for visual recognition, in: NIPS, 2009. 1, 3

[2] A. Bursuc, G. Tolias, H. Jégou, Kernel local descriptors with implicit rotation matching, in: ICMR, 2015. 1, 2

[3] A. Mukundan, G. Tolias, O. Chum, Multiple-kernel local-patch descriptor, in: BMVC, 2017. 1, 2

[4] G. Tolias, A. Bursuc, T. Furon, H. Jégou, Rotation and translation covariant match kernels for image retrieval, CVIU 140 (2015) 9–20. 1, 2, 3, 4, 6, 7, 11, 12

[5] A. Rahimi, B. Recht, Random features for large-scale kernel machines, in: NIPS, 2007. 1, 4

[6] A. Vedaldi, A. Zisserman, Efficient additive kernels via explicit feature maps, Trans. PAMI 34 (3) (2012) 480–492. 1, 3, 4, 11

[7] A. Chalechale, G. Naghdy, A. Mertins, Sketch-based image matching using angular partitioning, Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on 35 (1) (2005) 28–41. 1, 2

[8] V. Ferrari, L. Fevrier, F. Jurie, C. Schmid, Groups of adjacent contour segments for object detection, Trans. PAMI 30 (1) (2008) 36–51. 1, 2

[9] M. Eitz, K. Hildebrand, T. Boubekeur, M. Alexa, Sketch-based image retrieval: Benchmark and bag-of-features descriptors, Visualization and Computer Graphics, IEEE Transactions on 17 (11) (2011) 1624–1636. 1, 3

[10] J. M. Saavedra, J. M. Barrios, S. Orand, Sketch based image retrieval using learned keyshapes (lks), in: BMVC, 2015. 1, 3, 10

[11] S. Parui, A. Mittal, Similarity-invariant sketch-based image retrieval in large databases, in: ECCV, Springer, 2014, pp. 398–414. 1, 3, 9, 10, 11, 12

[12] A. C. Berg, T. L. Berg, J. Malik, Shape matching and object recognition using low distortion correspondences, in: CVPR, 2005. 2

[13] I. Kokkinos, A. Yuille, Inference and learning with hierarchical shape models, IJCV 93 (2) (2011) 201–225. 2

[14] Y. Cao, C. Wang, L. Zhang, L. Zhang, Edgel index for large-scale sketch-based image search, in: CVPR, IEEE, 2011. 2, 3, 10

[15] X. Sun, C. Wang, C. Xu, L. Zhang, Indexing billions of images for sketch-based retrieval, in: ACM Multimedia, 2013. 2, 3

[16] T. Bui, L. Ribeiro, M. Ponti, J. Collomosse, Generalisation and sharing in triplet convnets for sketch based visual search, in: arXiv:1611.05301, 2016. 2, 3

[17] P. Sangkloy, N. Burnell, C. Ham, J. Hays, The sketchy database: learning to retrieve badly drawn bunnies, ACM Trans. on Graphics. 2, 3, 11

[18] Q. Yu, F. Lie, Y.-Z. Song, T. Xian, T. Hospedales, C. C. Loy, Sketch me that shoe, in: CVPR, 2016. 2, 3, 10, 11

[19] O. Chum, Optimizing explicit feature maps on intervals, Image and Vision Computing 66 (2017) 36–47. 2, 4, 5

[20] G. Tolias, O. Chum, Asymmetric feature maps with application to sketch based retrieval, in: CVPR, 2017. 2

[21] Local features: State of the art, open problems and performance evaluation, http://www.iis.ee.ic.ac.uk/ComputerVision/DescrWorkshop/. 2

[22] M. Eitz, J. Hays, M. Alexa, How do humans sketch objects?, ACM Transactions on Graphics. 2, 9

[23] Q. Yu, Y. Yang, Y.-Z. Song, T. Xiang, T. M. Hospedales, Sketch-a-net that beats humans, in: BMVC, 2015. 2

[24] C. Ma, X. Yang, C. Zhang, X. Ruan, M.-H. Yang, O. Coporation, Sketch retrieval via dense stroke features, in: BMVC, 2013. 2

[25] J. M. Saavedra, Sketch based image retrieval using a soft computation of the histogram of edge local orientations (s-helo), in: ICIP, 2014. 2, 10

[26] M. Eitz, K. Hildebrand, T. Boubekeur, M. Alexa, An evaluation of descriptors for large-scale image retrieval from sketched feature lines, Computers & Graphics 34 (5) (2010) 482–498. 3

[27] H. Riemenschneider, M. Donoser, H. Bischof, Image retrieval by shape-focused sketching of objects, in: Computer Vision Winter Workshop, 2011. 3, 10

[28] R. Hu, J. Collomosse, A performance evaluation of gradient field hog descriptor for sketch based image retrieval, CVIU 117 (7) (2013) 790–806. 3, 9, 10

[29] X. Cao, H. Zhang, S. Liu, X. Guo, L. Lin, Sym-fish: A symmetry-aware flip invariant sketch histogram shape descriptor, in: ICCV, IEEE, 2013. 3, 10

[30] T. Bui, J. Collomosse, Scalable sketch-based image retrieval using color gradient features, in: ICCV, 2015. 3, 10

[31] A. Shrivastava, T. Malisiewicz, A. Gupta, A. A. Efros, Data-driven visual similarity for cross-domain image matching, ACM Transactions on Graphics 30 (6) (2011) 154. 3

[32] A. Thayananthan, B. Stenger, P. H. Torr, R. Cipolla, Shape context and chamfer matching in cluttered scenes, in: CVPR, 2003. 3

[33] F. Wang, L. Kang, Y. Li, Sketch-based 3d shape retrieval using convolutional neural networks, in: CVPR, 2015. 3

[34] J. Song, Y.-Z. Song, T. Xiang, T. Hospedales, X. Ruan, Deep multi-task attribute-driven ranking for fine-grained sketch-based image retrieval, in: BMVC, 2016. 3

[35] Y. Li, T. M. Hospedales, Y.-Z. Song, S. Gong, Fine-grained sketch-based image retrieval by matching deformable part models, in: BMVC, 2014. 3

[36] J. Sivic, A. Zisserman, Video Google: A text retrieval approach to object matching in videos, in: ICCV, 2003. 3

[37] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, C. Bray, Visual categorization with bags of keypoints, in: ECCV Workshop Statistical Learning in Computer Vision, 2004. 3

[38] F. Perronnin, C. R. Dance, Fisher kernels on visual vocabularies for image categorization, in: CVPR, 2007. 3

[39] H. Jégou, M. Douze, C. Schmid, P. Pérez, Aggregating local descriptors into a compact image representation, in: CVPR, 2010. 3

[40] B. Scholkopf, A. Smola, Learning with Kernels, MIT Press, 2002. 3

[41] O. Chum, Low dimensional explicit feature maps, in: ICCV, 2015. 4

[42] O. Chum, J. Philbin, J. Sivic, M. Isard, A. Zisserman, Total recall: Automatic query expansion with a generative feature model for object retrieval, in: ICCV, 2007. 8

[43] Q. Danfeng, S. Gammeter, L. Bossard, T. Quack, L. V. Gool, Hello neighbor: Accurate object retrieval with k-reciprocal nearest neighbors, in: CVPR, 2011. 8

[44] A. Joly, O. Buisson, Logo retrieval with a contrario visual query expansion, in: ACM Multimedia, 2009. 8

[45] Y. Kalantidis, C. Mellina, S. Osindero, Cross-dimensional weighting for aggregated deep convolutional features, in: ECCVW, 2016. 8

[46] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, in: ICLR, 2014. 8

[47] H. Jégou, M. Douze, C. Schmid, Product quantization for nearest neighbor search, Trans. PAMI 33 (1) (2011) 117–128. URL http://lear.inrialpes.fr/pubs/2011/JDS11 8, 12

[48] P. Dollár, C. L. Zitnick, Structured forests for fast edge detection, in: ICCV, 2013. 9

[49] K. Schindler, D. Suter, Object detection by global contour shape, Pattern Recognition 41 (12) (2008) 3736–3748. 9

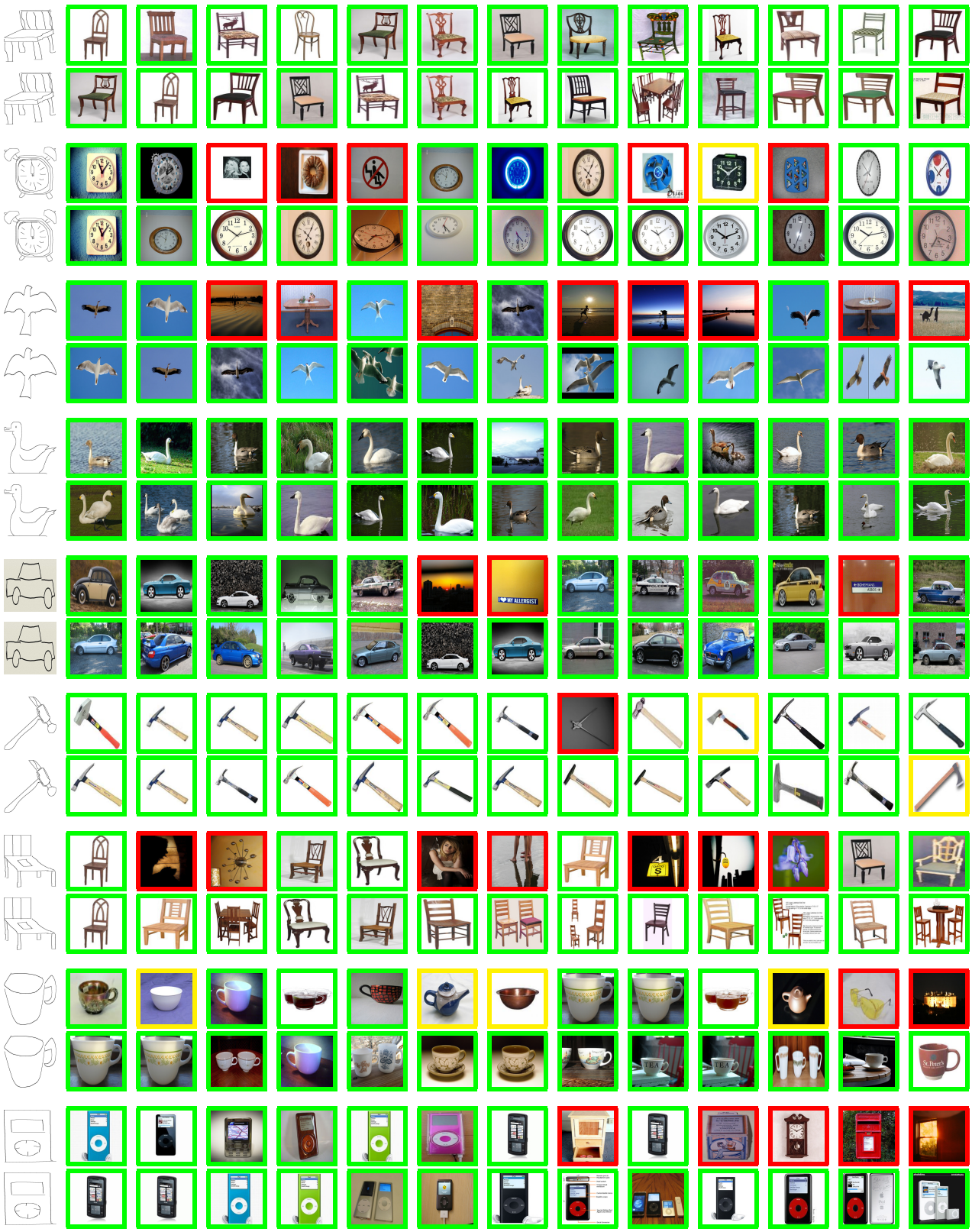


Figure 8: Random selection of sketch queries and the corresponding top ranked images using our method. We present results for our sketch descriptors (top row) and our sketch descriptor followed by AQE10.

XIV Deep shape matching

Title: Deep shape matching
Authors: F. Radenovic, G. Tolias, O. Chum
Published at: ECCV 2014

Deep Shape Matching

Filip Radenović Giorgos Tolias Ondřej Chum

Visual Recognition Group, FEE, CTU in Prague
{filip.radenovic, giorgos.tolias, chum}@cmp.felk.cvut.cz

Abstract. We cast shape matching as metric learning with convolutional networks. We break the end-to-end process of image representation into two parts. Firstly, well established efficient methods are chosen to turn the images into edge maps. Secondly, the network is trained with edge maps of landmark images, which are automatically obtained by a structure-from-motion pipeline. The learned representation is evaluated on a range of different tasks, providing improvements on challenging cases of domain generalization, generic sketch-based image retrieval or its fine-grained counterpart. In contrast to other methods that learn a different model per task, object category, or domain, we use the same network throughout all our experiments, achieving state-of-the-art results in multiple benchmarks.

Keywords: shape matching · cross-modal recognition and retrieval

1 Introduction

Deep neural networks have recently become very popular for computer-vision problems, mainly due to their good performance and generalization. These networks have been first used for image classification by Krizhevsky *et al.* [3], then their application spread to other related problems. A standard architecture of a classification network starts with convolutional layers followed by fully connected layers. Convolutional neural networks (CNNs) became a popular choice

Fig. 1. Three examples where **shape** is the only relevant information: sketch, artwork, extreme illumination conditions. Top retrieved images from the Oxford Buildings dataset [1]: CNN with an RGB input [2] (left), and our shape matching network (right). Query images are shown with black border.

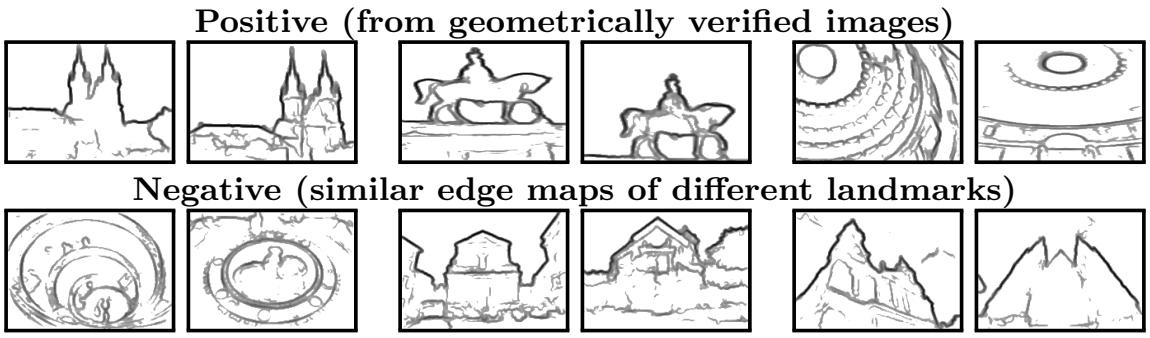


Fig. 2. Edge maps extracted from matching and non-matching image pairs that serve as training data for our network.

of learning image embeddings, *e.g.* in efficient image matching – image retrieval. It has been observed that the convolutional part of the classification network captures well *colours*, *textures* and *structures* within the receptive field.

In a number of problems, the colour and/or the texture is not available or misleading. Three examples are shown in Figure 1. For sketches or outlines, there is no colour or texture available at all. For artwork, the colour and texture is present, but often can be unrealistic to stimulate certain impression rather than exactly capture the reality. Finally, in the case of extreme illumination changes, such as a day-time versus night images, the colours may be significantly distorted and the textures weakened. On the other hand, the image discontinuities in colour or texture, as detected by modern edge detectors, and especially their shapes, carry the information about the content, independent of, or insensitive to, the illumination changes, artistic drawing and outlining.

This work is targeting at shape matching, in particular the goal is to extract a descriptor that captures the shape depicted in the input. The shape descriptors are extracted from image edge maps by a CNN. Sketches, black and white line drawings or cartoons are simply considered as a special type of an edge map.

The network is trained without any human supervision or image, sketch or shape annotation. Starting from a pre-trained classification network stripped off the fully connected layers, the CNN is fine-tuned using a simple contrastive loss function. Matching and non-matching training pairs are extracted from automatically generated 3D models of landmarks [2]. Edge maps detected on these images provide training data for the network. Examples of positive and negative pairs of edge maps are shown in Figure 2.

We show the importance of shape matching on two problems: 1) modality invariant representation, *i.e.* classification for domain generalization, and 2) cross modality matching of sketches to images.

In the domain generalization, some of the domains are available, but some are completely unseen during the training phase. We evaluate on domain generalization by performing object recognition. We extract the learned descriptors and train a simple classifier on the seen domains, which is later used to classify images of the unseen domain. We show, that for some combinations of seen-unseen domains, such as artwork and photograph, descriptors using colour and texture are useful. However, for some combinations, such as photograph and line

drawing, the shape information is crucial. Combining both types of descriptors outperforms the state-of-the-art approach in all settings.

In the cross modality matching task, it is commonly assumed that annotated training data is available for both modalities. Even for this task, we apply the domain generalization approach, using the descriptors learned on edge maps of building images. We evaluate the cross modality matching on sketch based image retrieval datasets. Modern sketch-based image retrieval take the path of object recognition from human sketches. Rather than performing shape matching, the networks are trained to recognize simplified human drawings. Such an approach requires very large number of annotated images and drawn sketches for each category of interest. Even though the proposed network is *not trained* to recognize human-drawn object sketches, our experiments show that it performs well on standard benchmarks.

2 Related work

Shape matching is shown useful in several computer vision tasks such as object recognition [4], object detection [5], 3D shape matching [6,7] and cross-modal retrieval [8,9]. In this section we review prior work related to sketch-based image retrieval, a particular flavor of cross-modal retrieval, where we apply the proposed representation. Finally, we discuss domain generalization approaches since our method is directly applicable on this problem handling it simply by learning shape matching.

Sketch-based image retrieval has been, until recently, handled with hand-crafted descriptors [10,11,12,13,14,15,16,17,18,19]. Deep learning methods have been applied to the task of sketch-based retrieval [20,21,22,23,24,25,26] much later than to the related task of image retrieval. We attribute the delay to the fact that the training data acquisition for sketch-based retrieval is much more tedious compared to image-based retrieval because it not only includes labeling the images, but also sketches must be drawn in large quantities. Methods with no learning typically carry no assumptions on the depicted categories, while the learning based methods often include category recognition into training. The proposed method aims at generic sketch-based retrieval, not limited to a fixed set of categories; it is, actually, not even limited to objects.

Learning-free methods have followed the same initial steps as in the traditional image search. These include the construction of either global [27,12,16] or local [28,29,11,30,14] image and/or sketch representations. Local representations are also using vector quantization to create a Bag-of-Words model [31]. Further cases are symmetry-aware and flip invariant descriptors [30], descriptors that are based on local contours [29] or line segments [14], and kernel descriptors [19]. Transformation invariance is often sacrificed for the sake of scalability [8,9]. In contrast, the method proposed in this paper is fully translation invariant, and scalable, because it reduces to a nearest-neighbor search in a descriptor space.

Learning-based methods require annotated data in both domains, typically for a fixed set of object categories, making the methods [6,20,21,22,23,24,25,26,32] to be category specific. End-to-end learning methods are applied to both category

level [25,26] and to fine-grained, *i.e.* sub-category level retrieval [22,23,24,32], while sometimes a different model per category has to be learned [33,22,34,32]. A sequence of different learning and fine-tuning stages is applied [22,23,24,25,26], involving massive manual annotation effort. For example, the Sketchy dataset [23] required collectively 3,921 hours of sketching. On the contrary, our proposed fine-tuning does not require any manual annotation.

Domain generalization is handled in a variety of ways, ranging from learning domain invariant features [35,36] to learning domain invariant classifiers [37,38] or both [39,40]. Several methods focus on one-way shift between two domains, such as sketch-based retrieval described earlier or learning on real photos and testing on art [41,42]. An interesting benchmark is released in the work of Li *et al.* [39], where four domains of increasing visual abstraction are used, namely *photos*, *art*, *cartoon*, and *sketches* (PACS). Prior domain generalization methods [35,36,37] are shown effective on PACS, while simply training a CNN on all the available (seen) domains is a very good baseline [39]. We tackle this problem from the representation point of view and focus on the underlined shapes. Our shape descriptor is extracted and the labels are used only to train a linear classifier. In this fashion, we are able to train on a single domain and test on all the rest, while common domain generalization approaches require different domains present in the training set.

3 Method

In this section we describe the proposed approach. The process of fine-tuning the CNN is described in Section 3.1, while the final representation and the way it is used for retrieval and classification is detailed in Section 3.2.

We break the end-to-end process of image description into two parts. In the first part, the images are turned into edge maps. In particular, throughout all our experiments we used the edge detector of Dollár and Zitnick [43] due to its great trade-off between efficiency and accuracy, and the tendency not to consider textured regions as edges. Our earlier experiments on sketch-based image retrieval with a CNN-based edge detector [44] did not show any significant changes in the performance. An image is represented as an edge map, which is a 2D array containing the edge strength in each image pixel. The edge strength is in the range of $[0, 1]$, where 0 represents background. Sketches, in the case of sketch-to-image retrieval, are represented as a special case of an edge map, where the edge strength is either 0 for the background or 1 for a contour.

The second part is a fully convolutional network extracting a global image descriptor. The two part approach allows, in a simple manner, to unify all modalities at the level of edge maps. Jointly training these two parts, *e.g.* in the case of a CNN-based edge detector [44], can deliver an image descriptor too. However, this descriptor may not be based on shapes. It is unlikely that such an optimization would end in a state where the representation between the two parts actually corresponds to edges. Enforcing this with additional training data in the form of edge maps and a loss on the output of the first part is exactly what we are avoiding and improving in this work.

3.1 Training

We use a network architecture previously proposed for image classification [45], in particular, we use all convolutional layers and the activations of the very last one, *i.e.*, the network is stripped of the fully-connected layers. The CNN is initialized by the parameters learned on a large scale annotated image dataset, such as ImageNet [46]. This is a fairly standard approach adopted in a number of problems, including image search [47,2,48]. The network is then fine-tuned with pairs of image edge maps.

The network. The image classification network expects an RGB input image, while the edge maps are only two dimensional. We sum the first convolution filters over RGB. Unlike in RGB input, no mean pixel subtraction is performed to the input data. To obtain a compact, shift invariant descriptor, a global max-pooling [49] layer is appended after the last convolutional layer. This approach is also known as Maximum Activations of Convolutions (MAC) vector [50]. After the MAC layer, the vectors are ℓ_2 normalized.

Edge filtering. A typical output of edge detectors is a strength of an edge in every pixel. We introduce an edge filtering layer to address two frequent issues with edge responses. First, the background often contains close-to-zero responses, which typically introduce noise into the representation. This issue is commonly handled by thresholding the response function. Second, the strength of the edges provides ordering, *i.e.* higher edge response implies that the edge is more likely to be present, however its value typically does not have practical interpretation. Prior to the first convolution layer, a continuous and differentiable function is pre-pended. This layer is trained together with the rest of the network to transform the edge detector output with soft thresholding by a sigmoid and power transformation. Denote the edge strength by $w \in [0, 1]$. Edge filtering is performed as

$$f(w) = \frac{w^p}{1 + e^{\beta(\tau - w)}}, \tag{1}$$

where p controls the contrast between strong and weak edges, τ is the threshold parameter, and β is the scale of the sigmoid choosing between hard thresholding and a softer alternative. The final function (1) with learned parameters is plotted

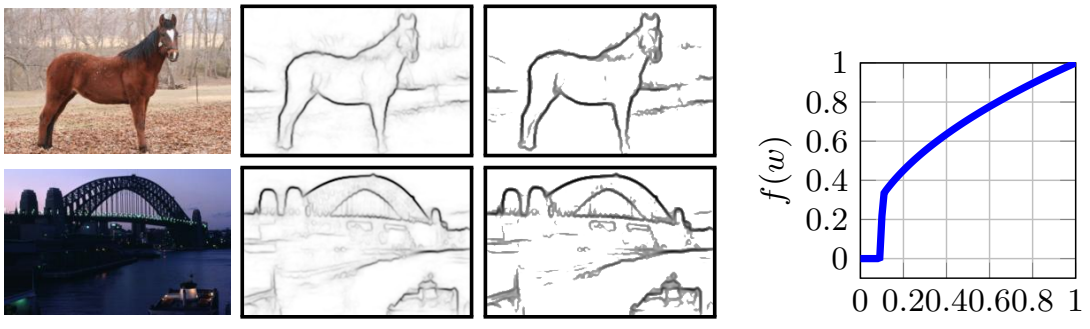


Fig. 3. Sample images, the output of the edge detector, the filtered edge map, and the edge-filtering function.

in Figure 3 (right). The figure also visually demonstrates the effect of application of the filtering. The weak edges are removed on the background and the result appearance is closer to a rough sketch, while the uncertainty in edges is still preserved.

Fine tuning. The CNN is trained with Stochastic Gradient Descent in a Siamese fashion with contrastive loss [51]. The positive training pairs are edge maps of matching images (similarity of the edge maps is not considered), while the negative pairs are similar edge maps (according to the current state of the network) of non-matching images.

Given a pair of vectors $\mathbf{x}$ and $\mathbf{y}$, the loss is defined as their squared Euclidean distance $||\mathbf{x}-\mathbf{y}||^2$ for positive examples, and as $\max\{(m-||\mathbf{x}-\mathbf{y}||)^2, 0\}$ for negative examples. Hard-negative mining is performed several times per epoch which has been shown to be essential [2,48].

Training data. The training images for fine tuning the network are collected in a fully automatic way. In particular, we use the publicly available dataset used in Radenovic *et al.* [2] and follow the same methodology, briefly reviewed in the following. A large unordered image collection is passed through a 3D reconstruction system based on local features and Bag-of-Words retrieval [52,53]. The outcome consists of a set of 3D models which mostly depict outdoor landmarks and urban scenes. For each landmark, a maximum of 30 six-tuples of images are being selected. The six-tuple consists of: one image as the training query, then one matching image to the training query, and five similar non-matching images. This gives arise to one positive and five negative pairs. The geometry of the 3D models, including camera positions, allows to mine matching images, *i.e.* those that share adequate visual overlap. Negative-pair mining is facilitated by the 3D models, too: negative images are chosen only if they belong to a different model.

Data augmentation. A standard data-augmentation, *i.e.* random horizontal flipping (mirroring) procedure is applied to introduce further variance in the training data and to avoid over-fitting. The training query and the positive example are jointly mirrored with 50% probability. Negative examples are sought after eventual flipping. We propose an additional augmentation technique for the selected training queries. Their edge map responses are thresholded with a random threshold uniformly chosen from $[0, 0.2]$ and the result is binarized. Matching images (in positive examples) are left unchanged; negative images are selected after the transformation. This augmentation process is applied with a probability of 50%. It offers a level of shape abstraction and mimics the asymmetry of sketch-to-edge map matching. The randomized threshold can be also seen as an approximation of the stroke removal in [22].

3.2 Representation, Search and Classification

We use the trained network to extract image and sketch descriptors capturing the underlying shapes, which are then used to perform cross-modal image retrieval, in particular sketch-based, and object recognition via transfer learning, in particular domain generalization.

Representation. The input to the descriptor extraction process is always resized to a maximum dimensionality of 227×227 pixels. A multi-scale representation is performed by processing at 5 fixed scales, *i.e.*, re-scaling the original input by a factor of $\frac{1}{2}$, $1/\sqrt{2}$, 1, $\sqrt{2}$, 2, and, with the additional mirroring, 10 final instances are produced. *Images* undergo edge detection and the resulting edge map [43] is fed to the CNN[1]. *Sketches* come in the form of strokes, thin line drawings, or brush drawings, depending on the input device or the dataset. To unify the sketch input, a simple morphological filter is applied to a binary sketch image. Specifically, a morphological thinning followed by dilation is performed. After the pre-processing, the sketch is treated as an edge map. As a consequence of the rescaling and mirroring, an image/sketch is mapped to 10 high dimensional vectors. We refer to these ℓ_2 normalized vectors as EdgeMAC descriptors. They are subsequently sum-aggregated or indexed separately, depending on the evaluation benchmark, see Section 4 for more details.

Search. An image collection is indexed by simply extracting and storing the corresponding EdgeMAC descriptors for each image. Search is performed by nearest-neighbors search of the query descriptor in the database. This makes retrieval compatible with approximate methods [54,55] that can speed up search and offer memory savings.

Classification. We extract EdgeMAC descriptors from labeled images and train a multi-class linear classifier [56] to perform object recognition. This is especially useful for transfer learning when the training domain is different from the target/testing one. In this case, no labeled images of the training domain are available during the training of our network and no labeled images of the target domain are available during classifier training.

3.3 Implementation details

In this section we discuss implementation details. The training dataset used to train our network is presented. We train a single network, which is then used for different tasks. Training sets provided for specific tasks are not exploited.

Training data. We use the same training set as in the work of Radenovic *et al.* [2][2] which comprises landmarks and urban scenes. There are around 8k tuples. Due to the overlap of landmarks contained in the training set and one of the test sets involved in our evaluation, we manually excluded these landmarks from our training data. We end up with with 5,969 tuples for training and 1,696 for validation. Hard negatives are re-mined 3 times per epoch [2] from a pool of around 22k images.

Training implementation. We use the MatConvNet toolbox [57] to implement the learning. We initialize the convolutional layers by VGG16 [45] (results in 512D EdgeMAC descriptor) trained on ImageNet and sum the filters of the first

[1] We perform zero padding by 30 pixels to avoid border effects.
[2] Training data available at cmp.felk.cvut.cz/cnnimageretrieval

layer over the feature maps dimension to accommodate for the 2D edge map input instead of the 3D image. The edge-filtering layer is initialized with values $p = 0.5$, $\tau = 0.1$ and β is fixed and equal to 500 so that it always approximates hard thresholding. Additionally, the output of the egde-filtering layer is linearly scaled from $[0, 1]$ to $[0, 10]$. Initial learning rate is $l_0 = 0.001$ with an exponential learning rate decay $l_0 \exp(-0.1j)$ over epoch j; momentum is 0.9; weight decay is 0.0005; contrastive loss margin is 0.7; and batch size is equal to 20 training tuples. All training images are resized so that the maximum extent is 200 pixels, while keeping the original aspect ratio.

Training time. Training is performed for at most 20 epochs and the best network is chosen based on the performance on validation tuples. The whole training takes about 10 hours on a single GeForce GTX TITAN X (Maxwell) GPU with 12GB of memory.

4 Experiments

We evaluate EdgeMAC descriptor on domain generalization and sketch-based image retrieval. We train a single network and apply it on both tasks proving the generic nature of the representation.

4.1 Domain Generalization through Shape Matching

We extract EdgeMAC descriptors from labeled images, sum-aggregate descriptors of rescaled and mirrored instances and ℓ_2 normalize to produce one descriptor per image, and train a linear classifier [56] to perform object recognition. We evaluate on domain generalization to validate the effectiveness of our representation on shape matching.

PACS dataset was recently introduced by Li *et al.* [39]. It consists of images coming from 4 domains with varying level of abstraction, namely, *art* (painting), *cartoon*, *photo*, and *sketch*. Images are labeled according to 7 categories, namely, dog, elephant, giraffe, guitar, horse, house, and person. Each time, one domain is considered unseen, otherwise called target or test domain, while the image of the other 3 are used for training. Finally, multi-class accuracy is evaluated on the unseen domain. In our work, we additionally perform classifier training using a single domain and then test on the rest. We find this scenario to be realistic, especially in the case of training on photos and testing on the rest. The domain of realistic photos is the richest in terms of annotated data, while others such as sketches and cartoons are very sparsely annotated.

Baselines. We are interested in translation invariant representations and consider the two following baselines. First, MAC [50] descriptors extracted using a network that is pre-trained on ImageNet. Second, MAC descriptors extracted by a network that is fine-tuned for image retrieval in a siamese manner [2]. These two baselines have the same descriptor extraction complexity as ours. They are extracted on RGB images, while ours on edge maps. Note, that we treat all

Table 1. Multi-class accuracy on PACS dataset for 4 different descriptors. The combined descriptor (pre-trained + ours) is constructed via concatenation. A: Art, C: Cartoon, P: Photo, S: Sketch, 3: all 3 other domains.

	Pre-trained (RGB)				Siamese [2] (RGB)				Ours (edge map)				Pre-trained+Ours			
Test →	A	C	P	S	A	C	P	S	A	C	P	S	A	C	P	S
Train A	N/A	59.2	95.0	33.1	N/A	59.5	86.3	42.9	N/A	55.9	61.2	65.6	N/A	61.6	94.9	38.4
Train C	71.7	N/A	86.8	37.0	61.0	N/A	77.0	51.6	45.2	N/A	57.3	74.8	69.3	N/A	85.0	55.3
Train P	72.5	33.3	N/A	24.8	66.0	38.0	N/A	31.9	45.4	42.3	N/A	46.3	73.3	34.0	N/A	27.61
Train S	31.9	49.5	42.5	N/A	38.7	49.3	44.4	N/A	34.8	63.0	43.3	N/A	33.7	59.3	43.4	N/A
Train 3	78.0	68.0	94.4	47.1	71.5	64.3	85.1	56.0	53.8	67.9	64.5	74.7	80.0	68.7	93.7	62.7
Mean 3	71.9				69.2				65.2				76.2			

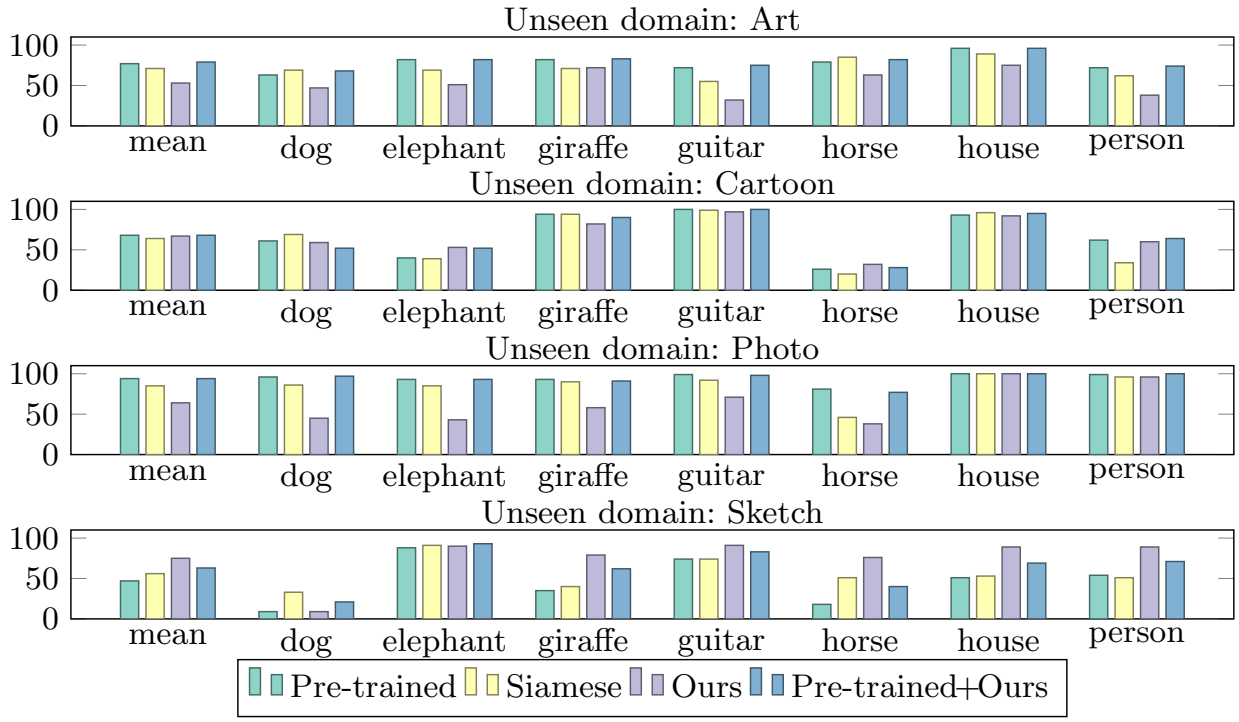


Fig. 4. Classification accuracy on PACS dataset with different descriptors. Testing is performed on 1 unseen domain each time, while training is performed on the other 3.

domains as images with our approach and extract edge maps, *i.e.* we do not perform any special treatment on sketches as in the case of sketch retrieval.

Performance comparison. We evaluate our descriptor, the two baselines, and the concatenated version of ours and the descriptor of the pre-trained baseline network, and report results in Table 1. Our representation significantly improves sketch recognition while training on a single or all seen domains. Similar improvements are observed for cartoon recognition when training on photos or sketches, while when training on artwork the color information appears to be beneficial. We consider the case of training only on photos and testing on other domains to be the most interesting and realistic one. In this scenario, we provide improvements, compared to the baselines, for sketch recognition (15% and 22%) and cartoon recognition (4% and 9%). Finally, the combined descriptor reveals the complementarity of the representations in several cases, such as artwork and cartoon recognition while training on all seen domains, or training on single domain when artwork is involved, *e.g.*. train on P (or A) and test on A (or C). The

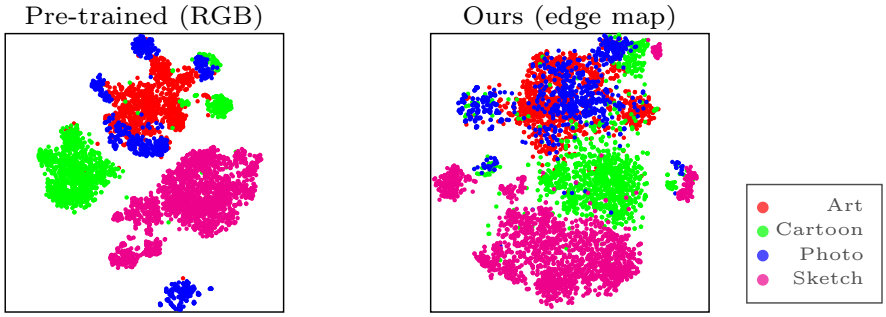


Fig. 5. Visualization of PACS images with t-SNE (more overlap is better).

best reported score on PACS is 69.2 [39] by fine-tuning AlexNet on PACS. The achieved score by our descriptor with fine-tuned VGG (PACS not used during network training) is 76.2, which is significantly higher. The same experiment with AlexNet achieves 70.9. Performance is reported per category in Figure 4. Our descriptor achieves significant improvements on most categories for sketch recognition, while the combined is a safe choice in several cases. Interestingly, our experiments reveal that the siamese baseline slightly improves shape matching, despite being trained on RGB images.

Visualization with t-SNE. We use t-SNE [58] to reduce the dimensionality of descriptors to 2 and visualize the result for the pre-trained baseline and our descriptor in Figure 5. The different modalities are brought closer with our descriptor. Observe how separated is the sketch modality with the pre-trained network that receives an RGB image for input.

4.2 Sketch-based Image Retrieval

We extract EdgeMAC descriptors to index an image collection, treat sketch queries as described in Section 3.2 and perform sketch-based image retrieval via simple nearest neighbor search.

Test datasets and evaluation protocols. The method is evaluated on two standard sketch-based image retrieval benchmarks.

Flickr15k [11] consists of 15k database images and 330 sketch queries that are related to 33 categories. Categories include particular object instances (Brussels Cathedral, Colosseum, Arc de Triomphe, *etc.*), generic objects (airplane, bicycle, bird, *etc.*), and shapes (circle shape, star shape, heart, balloon, *etc.*). The performance is measured via mean Average Precision (mAP) [1]. We sum-aggregate descriptors of rescaled and mirrored instances and ℓ_2 normalize to produce one descriptor per image. Search is performed by a cosine similarity nearest-neighbor search.

Shoes/Chairs/Handbags [22,32] datasets contain images from one category only, *i.e.* shoe/chair/handbag category respectively. It consists of pairs of a photo and a corresponding hand-drawn detailed sketch of this photo. There are 419, 297, and 568 sketch–photo pairs of shoes, chairs, and handbags, respectively. Out of these, 304, 200, and 400 pairs are selected for training, and 115, 97, and 168 for testing shoes, chairs, and handbags, respectively. The performance

Table 2. Performance evaluation of the different components of our method on Flickr15k dataset. Network: off-the-shelf (O), fine-tuned (F).

Component	Network							
	O	O	F	F	F	F	F	F
Train/Test: Edge filtering		■	■	■	■	■	■	■
Train: Query binarization				■	■	■	■	■
Test: Mirroring					■		■	■
Test: Multi-scale						■	■	■
Test: Diffusion								■
mAP	25.9	27.9	38.4	41.9	43.4	45.6	46.1	68.9

is measured via the matching accuracy at the top K retrieved images, denoted by acc.@K. The underlying task is quite different compared to Flickr15k. The photograph used to generate the sketch is to be retrieved, while all other images are considered false positives. The search protocol used in [22] is as follows: Descriptors are extracted from 5 image crops (corners and center) and their horizontally mirrored counterparts. This holds for database images and the sketch query. During search, these 10 descriptors are compared one-to-one and their similarity is averaged. For fair comparison, we adopt this protocol and do not use a single descriptor per image/sketch for this benchmark. However, instead of image crops, we extract descriptors at 5 image scales and their horizontally mirrored counterparts, as these are defined in Section 3.2.

Impact of different components. Table 2 shows the impact of different components on the final performance of the proposed method as measured on Flickr15k dataset. Direct application of the off-the-shelf CNN on edge maps already outperforms most prior hand-crafted methods (see Table 3). Adding the edge-filtering layer to the off-the-shelf network improves the precision. The initial parameters for filtering are used. Fine-tuning brings significant jump to 38.4 mAP, which is already the state-of-the-art on this dataset. Random training-query binarization and multi-scale with mirroring representation further improve the mAP score to 46.1. Finally, we boost the recall of our sketch-based retrieval by global diffusion, as recently proposed by Iscen *et al.* [59]. We construct the neighborhood graph by combining kNN-graphs built on two different similarities [60,61]: edge map similarity with EdgeMAC and image similarity with CroW descriptors [62]. This increases the performance to 68.9.

Performance evolution during learning. We report the performance of the fine-tuned network at different stages (epochs) of training. The same network is evaluated for all datasets as we train a single network for all tasks. The performance is shown in Figure 6 for both benchmarks. On all datasets, the fine-tuning significantly improves the performance already from the first few epochs.

As a sanity check, we also perform a non-standard sketch-to-sketch evaluation. On the Flickr15k dataset, each of the 330 sketches is used to query the other 329 sketches (the query sketch is removed from the evaluation), which attempts to retrieve sketches of the same category. The evolution of the performance shows similar behavior as the sketch-to-image search, *i.e.*, the learning on edge maps improves the performance on sketch-to-sketch retrieval, see Figure 6.

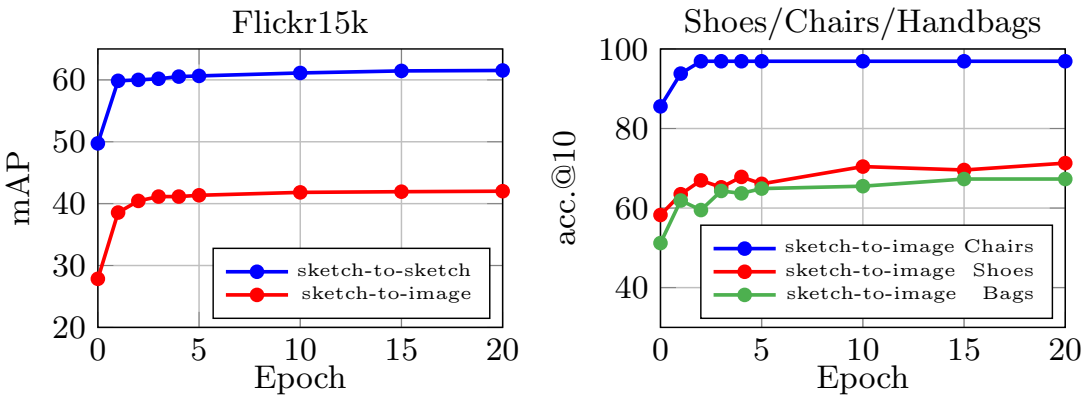


Fig. 6. Performance evaluation of the fine-tuned network over training epochs for the single-scale representation. All shown datasets and their evaluation protocols are described in Section 4.2.

Comparison with the state of the art. We extensively compare our method to the state-of-the-art performing methods on both benchmarks. Whenever code and trained models are publicly available, we additionally evaluate them on test sets they were not originally applied on. In cases that the provided code is used for evaluation on Flickr15k we center and align the sketches appropriately in order to achieve high scores, while our method is translation invariant so there is no such need. First we give a short overview of the best performing and most relevant methods to ours. Finally, a comparison via quantitative results is given.

Shoes/Chairs/Handbags networks [22,32] are trained from scratch based on the Sketch-a-Net architecture [63]. This is achieved by the following steps [22][3]: (i) Training with classification loss for 1k categories from ImageNet-1K data with edge maps input. (ii) Training with classification loss for 250 categories of TU-Berlin [64] sketch data. (iii) Training a triplet network with shared weights and ranking loss on TU-Berlin sketches and ImageNet images. (iv) Finally, training separate networks for fine-grain instance-level ranking using the Shoes/Chairs/ /Handbags training datasets. This approach is later improved [32] by adding an attention module with a coarse-fine fusion (CFF) into the architecture, and by extending the triplet loss with a higher order learnable energy function (HOLEF). Such a training involves various datasets, with annotation at different levels, and a variety of task-engineered loss functions. Note that the two models available online achieve higher performance than the ones reported in [22], due to parameter retuning. We compare our results to their best performing models.

Sketchy network [23] consists of two asymmetric sketch and image branches, both initialized with GoogLeNet. The training involves the following steps[4]: (i) Training for classification on TU-Berlin sketch dataset. (ii) Separate training of the sketch branch with classification loss on 125 categories of Sketchy dataset and training of the image branch with classification loss on the same categories with additional 1000 Flickr photos per category. (iii) Training both branches in a triplet network with ranking loss on the Sketchy sketch–photo pairs. The last part involves approximately 100k positive and a billion negative pairs.

[3] Networks/code available at github.com/seuliufeng/DeepSBIR
[4] Network/code available at github.com/janesjanes/sketchy

Table 3. Performance comparison via mean Average Precision (mAP) with the state-of-the-art sketch-based image retrieval on the Flickr15k dataset. Best result is highlighted in **red**, second best in **bold**. Query expansion methods are shown below the horizontal line and are highlighted separately. Our evaluation of the methods that do not originally report results on Flickr15 is marked with †.

Hand-crafted methods

Method	Dim	mAP
GF-HOG [11]	n/a	12.2
S-HELO [12]	1296	12.4
HLR+S+C+R [14]	n/a	17.1
GF-HOG extended [15]	n/a	18.2
PerceptualEdge [16]	3780	18.4
LKS [17]	1350	24.5
AFM [19]	243	30.4
AFM+QE [19]	755	**57.9**

CNN-based methods

Method	Dim	mAP
Sketch-a-Net+EdgeBox [20]	5120	27.0
Shoes network [22]†	256	29.9
Chairs network [22]†	256	29.8
Sketchy network [23]†	1024	34.0
Quadruplet network [24]	1024	32.2
Triplet no-share network [26]	128	**36.2**
⋆ EdgeMAC	512	**46.1**
Sketch-a-Net+EdgeBox+GraphQE [20]	n/a	32.3
⋆ EdgeMAC+Diffusion	n/a	**68.9**

Quadruplet network [24] tackles the problem in a similar way as Sketchy network, however, they use ResNet-18 [65] architecture with shared weights for both sketch and image branches. The training involves the following steps: (i) Training with classification loss on Sketchy dataset. (ii) Training a network with triplet loss on Sketchy dataset, while mining three different types of triplets.

Triplet no-share network [26] consists of asymmetric sketch and image branches initialized by Sketch-a-Net and AlexNet [3], respectively. The training involves: (i) Separate training of the sketch branch with classification loss on TU-Berlin and training of the image branch with classification loss on ImageNet. (ii) Training a triplet network with ranking loss on TU-Berlin sketches augmented with 25k corresponding photos harvested from the Internet. (iii) Training a triplet network with ranking loss on the Sketchy dataset.

Performance comparison. We compare our network with other methods on both benchmarks. Methods that have not reported scores on a particular datasets are evaluated by ourselves while using the publicly available networks.

Results on the Flickr15k dataset are presented in Table 3, where our method significantly outperforms both hand-crafted descriptors and CNN-based that are learned on a variety of training data. This holds for both plain search with the descriptors, and for methods using re-ranking techniques, such as query expansion [66] and diffusion [59].

Results on the fine-grained Shoes/Chairs/Handbags benchmark are shown in Table 4. In this experiment, we also report the performance after applying descriptor whitening which is learned in a supervised way [2] by using the descriptors of the training images of this benchmark. A single whitening transformation is learned for all three datasets. Such a process takes only a few seconds once descriptors are given. It is orders of magnitude faster than using the training set to perform network fine-tuning. We achieve the top performance in 2 out of 3 categories and the second best in the other one. The approach of [22] and [32] train a separate network per category (3 in total), which is clearly not scalable to many objects. In contrast our approach uses a single generic network. An additional drawback is revealed when we evaluate the publicly available Shoes

Table 4. Performance comparison via accuracy at rank K (acc.@K) with the state-of-the-art sketch-based image retrieval on the Shoes/Chairs test datasets. Best result is highlighted in **red**, second best in **bold**. Note that [22] and [32] train a separate network per object category. †We evaluate the publicly available networks, because the performance is higher than the one originally reported in [22].

Method	Dim	Shoes		Chairs		Handbags	
		acc.@1	acc.@10	acc.@1	acc.@10	acc.@1	acc.@10
BoW-HOG + rankSVM [22]	500	17.4	67.8	28.9	67.0	2.4	10.7
Dense-HOG + rankSVM [22]	200K	24.4	65.2	52.6	93.8	15.5	40.5
Sketch-a-Net + rankSVM [22]	512	20.0	62.6	47.4	82.5	9.5	44.1
CCA-3V-HOG + PCA [18]	n/a	15.8	63.2	53.2	90.3	–	–
Shoes net [22]†	256	52.2	**92.2**	65.0	92.8	23.2	59.5
Chairs net [22]†	256	30.4	75.7	72.2	**99.0**	26.2	58.3
Handbags net [32]	256	–	–	–	–	39.9	82.1
Shoes net + CFF + HOLEF [32]	512	**61.7**	**94.8**	–	–	–	–
Chairs net + CFF + HOLEF [32]	512	–	–	**81.4**	95.9	–	–
Handbags net + CFF + HOLEF [32]	512	–	–	–	–	**49.4**	**82.7**
⋆ EdgeMAC	512	40.0	76.5	**85.6**	95.9	35.1	70.8
⋆ EdgeMAC + whitening	512	**54.8**	**92.2**	**85.6**	**97.9**	**51.2**	**85.7**

and Chairs networks on the category they were not trained on. We observe a significant drop in performance, see Table 4.

The number of parameters. Our reported results use the VGG16 network stripped off the fully connected layers (FC), leaving ∼15M parameters. The number of parameters of Sketch-A-Net [63] is ∼8.5M parameters, while when used for SBIR in two different branches (Shoes, Chairs, Handbags [22]) there is ∼17M parameters. Triplet no-share network [26] uses two branches (Sketch-a-Net with additional FC layer and AlexNet [3]) leading to ∼115M, and Sketchy [23] uses 2× GoogLeNet leading to ∼26M parameters. Our network has the smallest number of parameters from the competing methods.

5 Conclusions

We have introduced an approach to learn shape matching by training a CNN with edge maps of matching images. The training stage does not require any manual annotation, achieved by following the footsteps of image retrieval [2], where image pairs are automatically mined from large scale 3D reconstruction.

The generic applicability of the representation is proven by validating on a variety of cases. It achieves state-of-the-art results on standard benchmarks for sketch-based image retrieval, while we have further demonstrated the applicability beyond sketch-based image retrieval. Promising results were achieved for queries with different modality (artwork) and significant change of illumination (day-night retrieval). The descriptor is shown beneficial for object recognition via transfer learning, especially to classify images of unseen domains, such as cartoons and sketches, where the amount of annotated data is limited. Remarkably, the same network is applied in all the different tasks. Training data, trained models, and code are publicly available at cmp.felk.cvut.cz/cnnimageretrieval.

Acknowledgements: This work was supported by the MSMT LL1303 ERC-CZ grant and the OP VVV funded project CZ.02.1.01/0.0/0.0/16_019/0000765 “RCI”.

References

1. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching. In: CVPR. (2007) 1, 10
2. Radenović, F., Tolias, G., Chum, O.: CNN image retrieval learns from BoW: Unsupervised fine-tuning with hard examples. In: ECCV. (2016) 1, 2, 5, 6, 7, 8, 9, 13, 14
3. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS. (2012) 1, 13, 14
4. Belongie, S., Malik, J., Puzicha, J.: Shape matching and object recognition using shape contexts. PAMI (2002) 3
5. Ferrari, V., Fevrier, L., Jurie, F., Schmid, C.: Groups of adjacent contour segments for object detection. PAMI (2008) 3
6. Wang, F., Kang, L., Li, Y.: Sketch-based 3d shape retrieval using convolutional neural networks. In: CVPR. (2015) 3
7. Tabia, H., Laga, H.: Covariance-based descriptors for efficient 3d shape matching, retrieval, and classification. IEEE Trans. on Multimedia (2015) 3
8. Cao, Y., Wang, C., Zhang, L., Zhang, L.: Edgel index for large-scale sketch-based image search. In: CVPR. (2011) 3
9. Sun, X., Wang, C., Xu, C., Zhang, L.: Indexing billions of images for sketch-based retrieval. In: ACM Multimedia. (2013) 3
10. Eitz, M., Richter, R., Boubekeur, T., Hildebrand, K., Alexa, M.: Sketch-based shape retrieval. ACM Trans. on Graphics (2012) 3
11. Hu, R., Collomosse, J.: A performance evaluation of gradient field hog descriptor for sketch based image retrieval. CVIU (2013) 3, 10, 13
12. Saavedra, J.M.: Sketch based image retrieval using a soft computation of the histogram of edge local orientations (S-HELO). In: ICIP. (2014) 3, 13
13. Parui, S., Mittal, A.: Similarity-invariant sketch-based image retrieval in large databases. In: ECCV. (2014) 3
14. Wang, S., Zhang, J., Han, T.X., Miao, Z.: Sketch-based image retrieval through hypothesis-driven object boundary selection with hlr descriptor. IEEE Trans. on Multimedia (2015) 3, 13
15. Bui, T., Collomosse, J.: Scalable sketch-based image retrieval using color gradient features. In: ICCV. (2015) 3, 13
16. Qi, Y., Song, Y.Z., Xiang, T., Zhang, H., Hospedales, T., Li, Y., Guo, J.: Making better use of edges via perceptual grouping. In: CVPR. (2015) 3, 13
17. Saavedra, J.M., Barrios, J.M., Orand, S.: Sketch based image retrieval using learned keyshapes (LKS). In: BMVC. (2015) 3, 13
18. Xu, P., Yin, Q., Huang, Y., Song, Y.Z., Ma, Z., Wang, L., Xiang, T., Kleijn, W.B., Guo, J.: Cross-modal subspace learning for fine-grained sketch-based image retrieval. Neurocomputing (2017) 3, 14
19. Tolias, G., Chum, O.: Asymmetric feature maps with application to sketch based retrieval. In: CVPR. (2017) 3, 13
20. Bhattacharjee, S.D., Yuan, J., Hong, W., Ruan, X.: Query adaptive instance search using object sketches. In: ACM Multimedia. (2016) 3, 13
21. Qi, Y., Song, Y.Z., Zhang, H., Liu, J.: Sketch-based image retrieval via siamese convolutional neural network. In: ICIP. (2016) 3
22. Yu, Q., Lie, F., Song, Y.Z., Xian, T., Hospedales, T., Loy, C.C.: Sketch me that shoe. In: CVPR. (2016) 3, 4, 6, 10, 11, 12, 13, 14

23. Sangkloy, P., Burnell, N., Ham, C., Hays, J.: The sketchy database: learning to retrieve badly drawn bunnies. ACM Trans. on Graphics (2016) 3, 4, 12, 13, 14
24. Seddati, O., Dupont, S., Mahmoudi, S.: Quadruplet networks for sketch-based image retrieval. In: ICMR. (2017) 3, 4, 13
25. Liu, L., Shen, F., Shen, Y., Liu, X., Shao, L.: Deep sketch hashing: Fast free-hand sketch-based image retrieval. In: CVPR. (2017) 3, 4
26. Bui, T., Ribeiro, L., Ponti, M., Collomosse, J.: Generalisation and sharing in triplet convnets for sketch based visual search. In: arXiv:1611.05301. (2016) 3, 4, 13, 14
27. Chalechale, A., Naghdy, G., Mertins, A.: Sketch-based image matching using angular partitioning. IEEE Trans. on Systems, Man, and Cybernetics (2005) 3
28. Eitz, M., Hildebrand, K., Boubekeur, T., Alexa, M.: An evaluation of descriptors for large-scale image retrieval from sketched feature lines. Computers & Graphics (2010) 3
29. Riemenschneider, H., Donoser, M., Bischof, H.: Image retrieval by shape-focused sketching of objects. In: CVWW. (2011) 3
30. Cao, X., Zhang, H., Liu, S., Guo, X., Lin, L.: Sym-fish: A symmetry-aware flip invariant sketch histogram shape descriptor. In: ICCV. (2013) 3
31. Ma, C., Yang, X., Zhang, C., Ruan, X., Yang, M.H., Coporation, O.: Sketch retrieval via dense stroke features. In: BMVC. (2013) 3
32. Song, J., Yu, Q., Song, Y.Z., Xiang, T., Hospedales, T.: Deep spatial-semantic attention for fine-grained sketch-based image retrieval. In: ICCV. (2017) 3, 4, 10, 12, 13, 14
33. Li, Y., Hospedales, T.M., Song, Y.Z., Gong, S.: Fine-grained sketch-based image retrieval by matching deformable part models. In: BMVC. (2014) 4
34. Song, J., Song, Y.Z., Xiang, T., Hospedales, T., Ruan, X.: Deep multi-task attribute-driven ranking for fine-grained sketch-based image retrieval. In: BMVC. (2016) 4
35. Ghifary, M., Bastiaan Kleijn, W., Zhang, M., Balduzzi, D.: Domain generalization for object recognition with multi-task autoencoders. In: ICCV. (2015) 4
36. Muandet, K., Balduzzi, D., Schölkopf, B.: Domain generalization via invariant feature representation. In: ICML. (2013) 4
37. Xu, Z., Li, W., Niu, L., Xu, D.: Exploiting low-rank structure from latent domains for domain generalization. In: ECCV. (2014) 4
38. Khosla, A., Zhou, T., Malisiewicz, T., Efros, A.A., Torralba, A.: Undoing the damage of dataset bias. In: ECCV. (2012) 4
39. Li, D., Yang, Y., Song, Y.Z., Hospedales, T.M.: Deeper, broader and artier domain generalization. In: ICCV. (2017) 4, 8, 10
40. Bousmalis, K., Trigeorgis, G., Silberman, N., Krishnan, D., Erhan, D.: Domain separation networks. In: NIPS. (2016) 4
41. Crowley, E., Zisserman, A.: The state of the art: Object retrieval in paintings using discriminative regions. In: BMVC. (2014) 4
42. Crowley, E.J., Zisserman, A.: The art of detection. In: ECCV. (2016) 4
43. Dollár, P., Zitnick, C.L.: Structured forests for fast edge detection. In: ICCV. (2013) 4, 7
44. Kokkinos, I.: Pushing the boundaries of boundary detection using deep learning. In: ICLR. (2016) 4
45. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR. (2014) 5, 7
46. Dong, W., Socher, R., Li-Jia, L., Li, K., Fei-Fei, L.: ImageNet: A large-scale hierarchical image database. In: CVPR. (2009) 5

47. Arandjelović, R., Gronat, P., Torii, A., Pajdla, T., Sivic, J.: NetVLAD: CNN architecture for weakly supervised place recognition. In: CVPR. (2016) 5
48. Gordo, A., Almazan, J., Revaud, J., Larlus, D.: Deep image retrieval: Learning global representations for image search. In: ECCV. (2016) 5, 6
49. Razavian, A.S., Sullivan, J., Carlsson, S., Maki, A.: Visual instance retrieval with deep convolutional networks. ITE Trans. on MTA (2016) 5
50. Tolias, G., Sicre, R., Jégou, H.: Particular object retrieval with integral max-pooling of cnn activations. In: ICLR. (2016) 5, 8
51. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: CVPR. (2005) 6
52. Schönberger, J.L., Radenović, F., Chum, O., Frahm, J.M.: From single image query to detailed 3D reconstruction. In: CVPR. (2015) 6
53. Radenović, F., Schönberger, J.L., Ji, D., Frahm, J.M., Chum, O., Matas, J.: From dusk till dawn: Modeling in the dark. In: CVPR. (2016) 6
54. Muja, M., Lowe, D.G.: Fast approximate nearest neighbors with automatic algorithm configuration. In: VISAPP. (2009) 7
55. Johnson, J., Douze, M., Jégou, H.: Billion-scale similarity search with GPUs. In: arXiv:1702.08734. (2017) 7
56. Perronnin, F., Akata, Z., Harchaoui, Z., Schmid, C.: Towards good practice in large-scale learning for image classification. In: CVPR. (2012) 7, 8
57. Vedaldi, A., Lenc, K.: MatConvNet: Convolutional neural networks for matlab. In: ACM Multimedia. (2015) 7
58. Maaten, L.v.d., Hinton, G.: Visualizing data using t-sne. JMLR (2008) 10
59. Iscen, A., Tolias, G., Avrithis, Y., Furon, T., Chum, O.: Efficient diffusion on region manifolds: Recovering small objects with compact CNN representations. In: CVPR. (2017) 11, 13
60. Bai, S., Sun, S., Bai, X., Zhang, Z., Tian, Q.: Smooth neighborhood structure mining on multiple affinity graphs with applications to context-sensitive similarity. In: ECCV. (2016) 11
61. Zhang, S., Yang, M., Cour, T., Yu, K., Metaxas, D.N.: Query specific fusion for image retrieval. In: ECCV. (2012) 11
62. Kalantidis, Y., Mellina, C., Osindero, S.: Cross-dimensional weighting for aggregated deep convolutional features. In: ECCVW. (2016) 11
63. Yu, Q., Yang, Y., Song, Y.Z., Xiang, T., Hospedales, T.M.: Sketch-a-Net that beats humans. In: BMVC. (2015) 12, 14
64. Eitz, M., Hays, J., Alexa, M.: How do humans sketch objects? ACM Trans. on Graphics (2012) 12
65. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. (2016) 13
66. Chum, O., Philbin, J., Sivic, J., Isard, M., Zisserman, A.: Total recall: Automatic query expansion with a generative feature model for object retrieval. In: ICCV. (2007) 13

XV Panorama to panorama matching for location recognition

Title: Panorama to panorama matching for location recognition
Authors: A. Iscen, G. Tolias, Y. Avrithis, T. Furon, O. Chum
Published at: ICMR 2017

Panorama to panorama matching for location recognition

Ahmet Iscen[1] Giorgos Tolias[2] Yannis Avrithis[1] Teddy Furon[1] Ondřej Chum[2]
[1] Inria, Rennes, France
[2] Visual Recognition Group, Faculty of Electrical Engineering, CTU in Prague, Czech Republic

ABSTRACT

Location recognition is commonly treated as visual instance retrieval on "street view" imagery. The dataset items and queries are panoramic views, *i.e.* groups of images taken at a single location. This work introduces a novel panorama-to-panorama matching process, either by aggregating features of individual images in a group or by explicitly constructing a larger panorama. In either case, multiple views are used as queries. We reach near perfect location recognition on a standard benchmark with only four query views.

CCS CONCEPTS

•Information systems → Image search; •Computing methodologies → Visual content-based indexing and retrieval;

KEYWORDS

image retrieval, location recognition

ACM Reference format:
A. Iscen, G. Tolias, Y. Avrithis, T. Furon and O. Chum. 2017. Panorama to panorama matching for location recognition. In *Proceedings of ICMR '17, Bucharest, Romania, June 06-09, 2017,* 5 pages.
DOI: http://dx.doi.org/10.1145/3078971.3079033

1 INTRODUCTION

Location recognition has been treated as a visual instance retrieval task for many years [1, 3, 11, 23, 25, 33]. Additional, task-specific approaches include ground truth locations to find informative features [25], regression for a more precise localization [18, 31], or representation of the dataset as a graph [7]. A dense collection of multiple views allows 3D representations are possible, *e.g.* structured from motion [12], searching 2D features in 3D models [19, 24], or simultaneous visual localization and mapping [8]. However, this does not apply to sparse "street-view" imagery [30, 32], where dataset items and queries are groups of images taken at a single location, in a panorama-like layout.

Several approaches on visual instance retrieval propose to jointly represent a set of images. These sets of images can appear at the query or at the database side. In the former case, these images are different views of the same object or scene [2, 27] and finally performance is improved. This joint representation, which commonly is an average query vector constructed via aggregation, is presumably more robust than each individual query vector. On the other hand, when aggregating images on the database side it is better to group them together by similarity [14]; images are assigned to sets, and a joint representation is created per set.

This work revisits location recognition by aggregating images both on query and database sides. Our method resembles implicit construction of a panorama, *i.e.* images are combined in the feature space and not in the image space, but we also experiment with an explicit construction. Contrary to the general case of visual instance retrieval, it is easy to obtain multiple query images, e.g. capturing them with a smartphone or with multiple cameras in the case of autonomous driving. On the database side, location provides a natural way of grouping images together. Thus, contrary to generic retrieval, the images to be aggregated on the query and database sides, may not be similar to each other; they rather depict whatever is visible around a particular location.

We significantly outperform the state of the art without any form of supervision other than the natural, location-based grouping of images, and without any costly offline process like 3D reconstruction. Indeed we are reaching near perfect location recognition on the Pittsburgh dataset [32] even when we use as few as four views on the query side.

2 BACKGROUND

This section describes the related work on Convolutional Neural Network (CNN) based descriptors for image retrieval and on image set joint representations. Our approach applies these methods on the dataset and query images.

2.1 CNN Descriptors for Retrieval

CNN-based global descriptors are becoming popular in image retrieval, especially for instance-level search. Existing works [4, 17, 22, 29] employ "off-the-shelf" networks, originally trained on ImageNet, to extract descriptors via various pooling strategies. This offers invariance to geometric transformation and robustness to background clutter. Other approaches [5, 9, 20] fine-tune such networks to obtain descriptor representations specifically adapted for instance search.

NetVLAD [1] is a recent work that trains a VLAD layer on top of convolutional layers in an end-to-end manner. It is tuned for the location recognition task. The training images are obtained from panoramas, fed to a triplet loss to make it more compatible with image retrieval. As a result, their representation outperforms existing works in standard location recognition benchmarks.

2.2 Representing Sets of Vectors

Two common scenarios aggregate a set of vectors into a single vector representation for image retrieval. The first case involves aggregation of a large number of local descriptors, either to reduce

ICMR '17, Bucharest, Romania

978-1-4503-4701-3/17/06…$15.00
DOI: http://dx.doi.org/10.1145/3078971.3079033

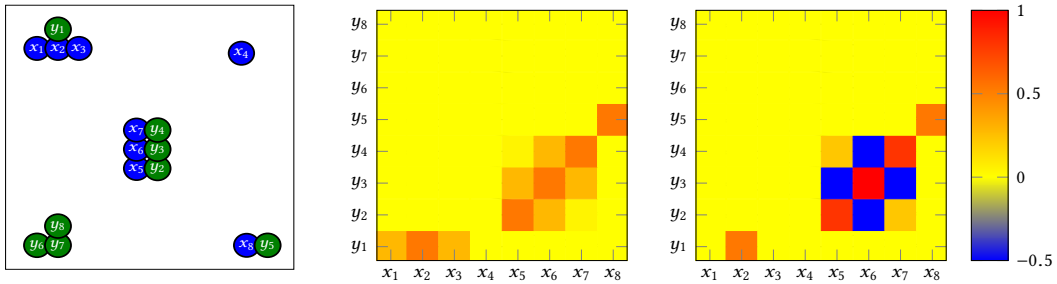


Figure 1: Left: Toy example of two vector sets X, Y on the 2D plane are shown on the left. Middle: Pairwise similarity between all vectors, cross-matching with sum-vectors, *i.e.* $\mathrm{X}^\top\mathrm{Y}$ (3). Only for visualization purposes, and since we are dealing with unnormalized 2D vectors, the similarity between vectors x, y is defined as $e^{\|\mathrm{x}-\mathrm{y}\|^2}$. Right: weighted pairwise-similarity between all vectors, cross-matching with pinv-vectors, *i.e.* $\mathrm{G}_\mathrm{X}^{-1}\mathrm{X}^\top\mathrm{Y}\mathrm{G}_\mathrm{Y}^{-1}$ (4).

the number of descriptors [26, 28], or to create a global descriptor [10, 16]. In the other case, which is exploited in this work, a set of global image descriptors is aggregated into a single vector representation to construct a joint representation for a set of images [14].

In particular, we follow the two *memory vector* construction strategies proposed by Iscen et al. [14]. The first method simply computes the sum of all vectors in a set. Given a set of vectors represented as the columns of a $d \times n$ matrix $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]$ with $\mathbf{x}_i \in R^d$, the *sum* memory vector is defined as

$$m(\mathbf{X}) = \mathbf{X}\mathbf{1}_n. \tag{1}$$

Assuming linearly independent columns ($n < d$), the second method is based on the Moore-Penrose pseudo-inverse $\mathbf{X}^+$ [21], given by

$$m^+(\mathbf{X}) = (\mathbf{X}^+)^\top \mathbf{1}_n = \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{1}_n. \tag{2}$$

It is theoretically optimized for high dimensional spaces and performs better in practice. This paper refers to the sum memory vector (1) as *sum-vector*, and to the pseudo-inverse memory vector (2) as *pinv-vector*.

Aggregating Dataset Images. The main purpose of aggregating dataset images is to reduce the computation cost of similarity search at query time [14]. Dataset vectors are assigned to sets in an offline process, and each set is represented by a single (memory) vector. At query time, the similarity between the query vector and each memory vector is computed, and memory vectors are ranked accordingly. Then the query is only compared to the database vectors belonging to the top ranked sets. This strategy eliminates the exhaustive computation of the similarities query vs. dataset vectors. Existing works use random assignments to create the sets, or weakly-supervised assignment based on k-means or kd-tree [13, 14].

Aggregating Query Images. Aggregation of query images has been also studied for instance-level object retrieval. Multiple images depicting the query object allow to better handle the problems of occlusion, view-point change, scale change and other variations. Arandjelovic et al. [2] investigate various scenarios, such as average or max pooling on query vectors and creating SVM models. Recently, Sicre and Jégou [27] have shown that aggregating query vectors with pinv-vector improves the search quality.

Aggregation of dataset images offers speed and memory improvements at the cost of performance loss. On the other hand, aggregation of query images is only applicable in the particular case of multiple available query images and offers performance improvements at no extra cost. Our approach adopts aggregation on both sides for the first time while enjoying speed, memory and performance improvements.

3 PANORAMA TO PANORAMA MATCHING

This section describes our contribution for location recognition. We assume that for each possible location we are given a set of images covering a full 360 degree view while consecutive images have an overlap (see Figure 2). We propose two ways to construct a *panoramic representation* of each location: an implicit way by vector aggregation and an explicit way by image stitching into a panorama and extraction of a single descriptor.

3.1 Implicit Panorama Construction

We form a panoramic representation by aggregating the descriptors of images from the same location. In this way, we implicitly construct a panorama in the descriptor space. In order to achieve this, we employ two approaches for creating memory vectors, *i.e.* sum-vector (1) and pinv-vector (2).

In contrast to previous works that aggregate the image vectors only on the dataset side [14] or only on the query side [27], we rather do it for both. This requires that the query is also defined by a set of images which offer a 360 degree view. A realistic scenario of this context is autonomous driving and auto-localization where the query is defined by such a set of images.

Assume that n images in a dataset location are represented by $d \times n$ matrix $\mathbf{X}$ and that k images in the query location by $d \times k$ matrix $\mathbf{Y}$. Analyzing the similarity between the two sum-vectors is straightforward. Panorama similarity is given by the inner product

$$s(\mathbf{X}, \mathbf{Y}) = m(\mathbf{X})^\top m(\mathbf{Y}) = \mathbf{1}_n^\top \mathbf{X}^\top \mathbf{Y}\mathbf{1}_k. \tag{3}$$

Similarly, panorama similarity for pinv-vectors is given by

$$s^+(\mathbf{X}, \mathbf{Y}) = m^+(\mathbf{X})^\top m^+(\mathbf{Y}) = \mathbf{1}_n^\top \mathbf{G}_\mathbf{X}^{-1}\mathbf{X}^\top\mathbf{Y}\mathbf{G}_\mathbf{Y}^{-1}\mathbf{1}_k, \tag{4}$$

where $\mathbf{G}_\mathbf{X} = \mathbf{X}^\top\mathbf{X}$ is the Gram matrix for $\mathbf{X}$. Compared to (3), the sum after cross-matching is weighted now, and the weights are given by $\mathbf{G}_\mathbf{X}^{-1}$ and $\mathbf{G}_\mathbf{Y}^{-1}$. This is interpreted as "democratizing" the result of cross-matching; the contribution of vectors that are similar within the same set are down-weighted, just as in handling the burstiness phenomenon for local descriptors [16]. We visualize this with a toy example in Figure 1. Unweighted cross-matching is

Figure 2: Example of all images assigned to a single location (first two rows) and the corresponding panorama (last row) covering a full 360 degree view, constructed by automatic stitching.

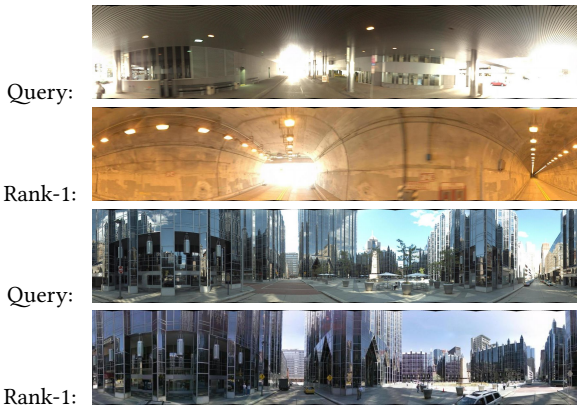


Figure 3: Two examples of failures with *pan2pan/net*. We show the query and the top ranked image from the dataset.

dominated by "bursty" vectors in the same cluster. Democratization down-weights these contributions.

3.2 Explicit Panorama Construction

Our second approach explicitly creates a panoramic image. The descriptors are then extracted from the panorama. Given that images of a location are overlapping, we construct a panoramic image using an existing stitching method. In particular, we use the work of Brown and Lowe [6], which aligns, stitches, and blends images automatically based on their local SIFT descriptors and inlier correspondences. Figure 2 shows a stitched panoramic image. Once stitching is complete, we extract a single global descriptor from the panorama image, capturing the entire scene.

4 EXPERIMENTS

In this section, we describe our experimental setup, and compare our method to a number of baselines using the state-of-the-art NetVLAD network in a popular location recognition benchmark.

4.1 Experimental Setup

The methods are evaluated on the Pittsburgh dataset [32] referred to as Pitt250k. It contains 250k database and 24k query images from Google Street View. It is split into training, validation, and test sets [1]. We evaluate our approach on the test set, which consists of 83,952 dataset images and 8,280 query images. Each image is associated with a GPS location and 24 images are associated with the same GPS location. Therefore, each panoramic representation aggregates 24 images. There is a total of 345 query locations and 3,498 dataset locations. We use NetVLAD for our descriptor representation in all experiments. While the original representation is $d = 4{,}096$ dimensional, we also experiment with reducing dimensionality to $d = 256$ by PCA.

The standard evaluation metric is Recall@N. It is defined to equal 1 if at least one of the top N retrieved dataset images is within 25 meters from the spatial location of the query. Average is reported over all queries. We follow this protocol for the baseline and other cases where the query images are used individually.

Aggregating on the query side implies that there is a single query per location: the number of queries decreases from 8,280 to 345. We report the average recall@N from these 345 panorama queries. Section 4.3 also experiments with a larger number of random queries, each capturing only a fraction of the panoramic view. In this case, recall@N is averaged over those random queries. Aggregating on the dataset side does not affect the standard evaluation.

4.2 Panorama Matching

We refer to our proposed method as panorama to panorama or *pan2pan* matching, in particular *pan2pan/sum* and *pan2pan/pinv* when aggregating descriptors with sum-vector and pinv-vector respectively; and as *pan2pan/net* when using a NetVLAD descriptor from an explicit panorama. We compare against the following baselines: image to image matching (*im2im*) as in the work by Arandjelovic et al. [1], image to panorama matching (*im2pan*) corresponding to dataset-side aggregation as in the work by Iscen et al. [14], and panorama to image matching (*pan2im*) corresponding to query-side aggregation as in the work by Sicre and Jégou [27].

Figure 4 compares all methods for different descriptor dimensions. Clearly, panorama to panorama matching outperforms all other methods. The improvement is consistent for all N and significant for low N: pan2pan/net obtains 98% recall@1! There are only 7 failure queries. Two of them are shown in Figure 3. One is a challenging query depicting an indoor parking lot and the other

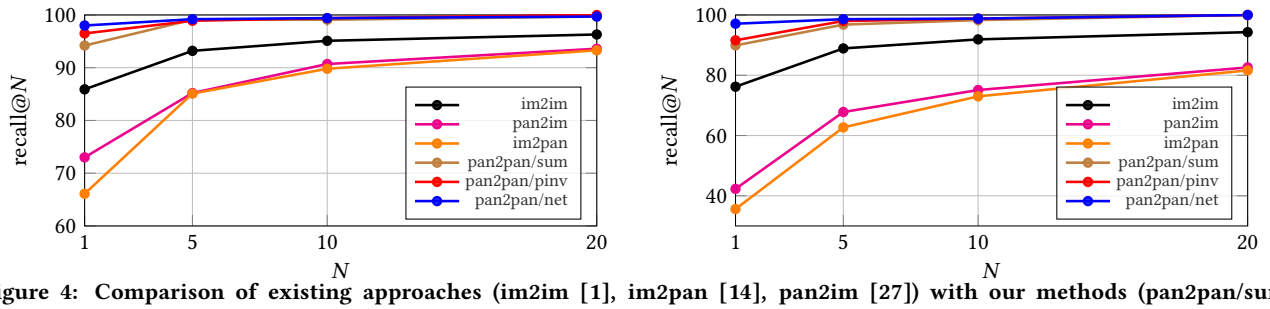


Figure 4: Comparison of existing approaches (im2im [1], im2pan [14], pan2im [27]) with our methods (pan2pan/sum, pan2pan/pinv and pan2pan/net) for the full 4096D (left) and for reduced dimensionality to 256D (right).

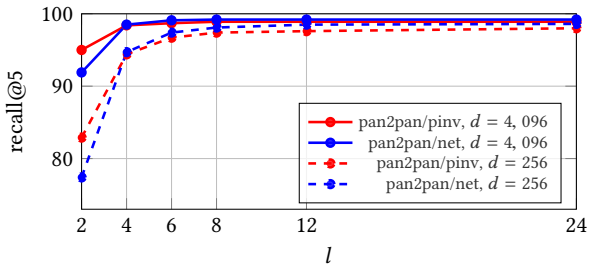


Figure 5: Recall@5 on Pitt250k, sampling l images from each query panorama and using NetVLAD descriptors of two different dimensionalities d. We report average measurements over 10 random experiments and compare our methods pan2pan/pinv and pan2pan/net.

actually retrieves the same building, which is incorrectly marked in the dataset's ground truth.

The recall is not only improved, but the search is also more efficient both speed-wise ($24^2\times$ faster) and memory-wise (24× less memory). Instead of comparing a given query image against 83k vectors, we only make 3.5k comparisons. Additional operations are introduced when aggregating the set of query images, but this cost is fixed and small compared to the savings from the dataset side.

Comparing to results in prior work, im2pan behaves as in the work of Iscen et al. [14] when compared to the baseline im2im. That is, memory compression and speed up at the cost of reduced performance. However, pan2im does not appear to be effective in our case, in contrast to the work of Sicre and Jégou [27]. On the contrary, pan2pan significantly improves the performance while enjoying both memory compression and search efficiency.

4.3 Sparse Panorama Matching

Aggregating on the dataset side is performed off-line. However, the user is required to capture images and to construct a full panorama (24 images in our case) at query time. Even though this is not a daunting task given the advances of smartphones and tablets, we additionally investigate a scenario where the user only captures a partial panoramic view.

In particular, we randomly sample a subset of l images from the query location and consider them as the query image set. Explicit panorama construction is no longer possible because the sampled images may not overlap and so we cannot stitch them. In this case, we feed sampled images through the convolutional layers only, and stack together all activations before pooling them through the NetVLAD layer (pan2pan/net for sparse panoramas).

Figure 5 shows the results. Our methods have near-perfect performance even for a small number of sampled images. When the user only takes four random photos, we are able to locate them up to 99% recall@5. Another interesting observation is that pan2pan/pinv outperforms pan2pan/net for $l = 2$, which is expected due to the nature of pinv-vec construction. It is theoretically shown to perform well even if all the vectors in the set are random, as shown in the original paper [14].

4.4 Comparison to Diffusion-based Retrieval

This work casts location recognition as a retrieval task. Query expansion techniques significantly improve retrieval performance. We compare to the state-of-the-art retrieval method by Iscen et al. [15], a kind of query expansion based on graph diffusion. In this method, an image is represented by individual region descriptors and at query time all query regions are processed.We compare to this method by considering that regions and images in [15] correspond to images and panoramas respectively in our scenario.

Our pan2pan/pinv and pan2pan/net approaches achieve 96.5% and 98% recall@1 respectively, while the approach [15] gives 91.9%. Even though query expansion improves the baseline, it does not help as much as our methods. This can be expected because [15] is based on many instances of the same object, which is not the case for location recognition on street view imagery.

5 CONCLUSIONS

Our method is unsupervised and conceptually very simple, yet very effective. Besides the performance gain, we make significant savings in space by aggregating descriptors of individual images over each group. The need for multiple query views is not very demanding because only four views are enough—an entire query panorama is definitely not needed.

Although our aggregation methods have been used for instance retrieval in the past, we are the first to successfully aggregate on both dataset and query-side for location recognition (which in fact has failed for instance retrieval [26]). An interesting finding is that although the NetVLAD descriptor has been explicitly optimized to aggregate CNN activations on the location recognition task, in some cases it is preferable to aggregate individual views into a pinv-vector rather than extracting a single NetVLAD descriptor from an explicit panorama.

Acknowledgments. The authors were supported by the MSMT LL1303 ERC-CZ grant. The Tesla K40 used for this research was donated by the NVIDIA Corporation.

REFERENCES

[1] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. In *CVPR*, 2016.

[2] R. Arandjelovic and A. Zisserman. Multiple queries for large scale specific object retrieval. In *BMVC*, 2012.

[3] R. Arandjelovic and A. Zisserman. Dislocation: Scalable descriptor distinctiveness for location recognition. In *ACCV*, 2014.

[4] A. Babenko and V. Lempitsky. Aggregating deep convolutional features for image retrieval. In *ICCV*, 2015.

[5] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky. Neural codes for image retrieval. In *ECCV*, 2014.

[6] M. Brown and D. G. Lowe. Automatic panoramic image stitching using invariant features. *IJCV*, 74:59–73, 2007.

[7] S. Cao and N. Snavely. Graph-based discriminative learning for location recognition. In *CVPR*, 2013.

[8] M. Cummins and P. Newman. Fab-map: Appearance-based place recognition and mapping using a learned visual vocabulary model. In *ICML*, 2010.

[9] A. Gordo, J. Almazan, J. Revaud, and D. Larlus. Deep image retrieval: Learning global representations for image search. *ECCV*, 2016.

[10] P.-H. Gosselin, N. Murray, H. Jégou, and F. Perronnin. Revisiting the fisher vector for fine-grained classification. *Pattern recognition letters*, 49, 2014.

[11] J. Hays and A. A. Efros. Im2gps: estimating geographic information from a single image. In *CVPR*, 2008.

[12] A. Irschara, C. Zach, J.-M. Frahm, and H. Bischof. From structure-from-motion point clouds to fast location recognition. In *CVPR*, 2009.

[13] A. Iscen, L. Amsaleg, and T. Furon. Scaling group testing similarity search. In *ICMR*, 2016.

[14] A. Iscen, T. Furon, V. Gripon, M. Rabbat, and H. Jégou. Memory vectors for similarity search in high-dimensional spaces. *IEEE Trans. Big Data*, 2017.

[15] A. Iscen, G. Tolias, Y. Avrithis, T. Furon, and O. Chum. Efficient diffusion on region manifolds: Recovering small objects with compact cnn representations. In *CVPR*, 2017.

[16] H. Jégou and A. Zisserman. Triangulation embedding and democratic kernels for image search. In *CVPR*, June 2014.

[17] Y. Kalantidis, C. Mellina, and S. Osindero. Cross-dimensional weighting for aggregated deep convolutional features. In *ECCVW*, 2016.

[18] A. Kendall, M. Grimes, and R. Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *ICCV*, 2015.

[19] Y. Li, N. Snavely, D. Huttenlocher, and P. Fua. Worldwide pose estimation using 3d point clouds. In *ECCV*, 2012.

[20] F. Radenović, G. Tolias, and O. Chum. CNN image retrieval learns from BoW: Unsupervised fine-tuning with hard examples. *ECCV*, 2016.

[21] C. R. Rao and S. K. Mitra. Generalized inverse of a matrix and its applications. In *Sixth Berkeley Symposium on Mathematical Statistics and Probability*, 1972.

[22] A. S. Razavian, J. Sullivan, S. Carlsson, and A. Maki. Visual instance retrieval with deep convolutional networks. *ITE Transactions on Media Technology and Applications*, 4, 2016.

[23] T. Sattler, M. Havlena, K. Schindler, and M. Pollefeys. Large-scale location recognition and the geometric burstiness problem. In *CVPR*, 2016.

[24] T. Sattler, B. Leibe, and L. Kobbelt. Improving image-based localization by active correspondence search. In *ECCV*, 2012.

[25] G. Schindler, M. Brown, and R. Szeliski. City-scale location recognition. In *CVPR*, 2007.

[26] M. Shi, Y. Avrithis, and H. Jégou. Early burst detection for memory-efficient image retrieval. In *CVPR*, 2015.

[27] R. Sicre and H. Jégou. Memory vectors for particular object retrieval with multiple queries. In *ICMR*, 2015.

[28] G. Tolias, Y. Avrithis, and H. Jégou. To aggregate or not to aggregate: Selective match kernels for image search. In *ICCV*, December 2013.

[29] G. Tolias, R. Sicre, and H. Jégou. Particular object retrieval with integral max-pooling of cnn activations. *ICLR*, 2016.

[30] A. Torii, R. Arandjelovic, J. Sivic, M. Okutomi, and T. Pajdla. 24/7 place recognition by view synthesis. In *CVPR*, 2015.

[31] A. Torii, J. Sivic, and T. Pajdla. Visual localization by linear combination of image descriptors. In *ICCVW*, 2011.

[32] A. Torii, J. Sivic, T. Pajdla, and M. Okutomi. Visual place recognition with repetitive structures. In *CVPR*, 2013.

[33] W. Zhang and J. Kosecka. Image based localization in urban environments. In *3DPVT*, 2006.

XVI Targeted Mismatch Adversarial Attack: Query with a Flower to Retrieve the Tower

Title: Targeted Mismatch Adversarial Attack: Query with a Flower to Retrieve the Tower
Authors: G. Tolias, F. Radenovic, O. Chum
Published at: ICCV 2019

Targeted Mismatch Adversarial Attack:
Query with a Flower to Retrieve the Tower

Giorgos Tolias Filip Radenovic Ondřej Chum
Visual Recognition Group, Faculty of Electrical Engineering, Czech Technical University in Prague

Abstract

Access to online visual search engines implies sharing of private user content – the query images. We introduce the concept of targeted mismatch attack for deep learning based retrieval systems to generate an adversarial image to conceal the query image. The generated image looks nothing like the user intended query, but leads to identical or very similar retrieval results. Transferring attacks to fully unseen networks is challenging. We show successful attacks to partially unknown systems, by designing various loss functions for the adversarial image construction. These include loss functions, for example, for unknown global pooling operation or unknown input resolution by the retrieval system. We evaluate the attacks on standard retrieval benchmarks and compare the results retrieved with the original and adversarial image.

1. Introduction

Information about users is a valuable article. Websites, service providers, and even operating systems collect and store user data. The collected data have various forms, *e.g*. visited websites, interactions between users in social networks, hardware fingerprints, keyboard typing or mouse movement patterns, *etc*. Internet search engines record what the users search for, as well as the responses, *i.e*. clicks, to the returned results.

Recent development in computer vision allowed efficient and precise large scale image search engines to be launched, such as Google Image Search. Nevertheless, similarly to text search engines, queries – the images – are stored and further analyzed by the provider[1]. In this work, we protect the user image (*target*) by constructing a novel image. The constructed image is visually dissimilar to the target, however, when used as a query, identical results are retrieved as with the target image. Large-scale search methods require short-code image representation, both for storage minimization and for search efficiency, which are usually extracted with Convolutional Neural Networks (CNN). We formulate the problem as an adversarial attack on CNNs.

[1]Google Search Help: "The pictures you upload in your search may be stored by Google for 7 days. They won't be a part of your search history, and we'll only use them during that time to make our products and services better."

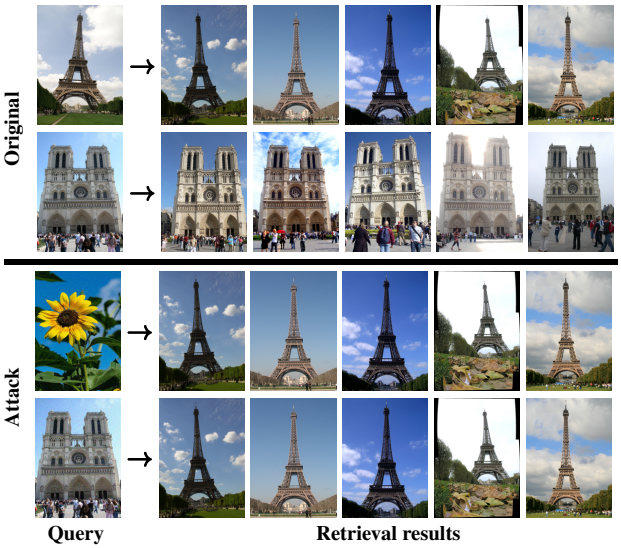


Figure 1. Top two rows show retrieval results to the user query image (target). Bottom two rows show the results of our attack where a carrier image (flower, Notre Dame) is perturbed to have identical descriptor to that of the target in the first row. Identical results are obtained without disclosing the target.

Adversarial attacks, as introduced by Szegedy *et al*. [35], study *imperceptible* non-random image perturbations to mislead a neural network. The first attacks were introduced and tested on image classification. In that context, adversarial attacks are divided into two categories, namely *non-targeted* and *targeted*. The goal of non-targeted attacks is to change the prediction of a test image to an arbitrary class [25, 24], while targeted attacks attempt to make a specific change of the network prediction, *i.e*. to misclassify the test image to a predefined target class [35, 7, 10].

Similarly to image classification, adversarial attacks have been proposed in the domain of image retrieval too. An non-targeted attack attempts to generate an image that for a human observer carries the same visual information, while for the neural network it appears dissimilar to other images of the same object [19, 20, 37]. This way, a user protects personal images and does not allow them to be in-

dexed for content-based search, even when the images are publicly available. In this paper, we address targeted attacks aiming to retrieve images that are related to a hidden target query without explicitly revealing the image (see Figure 1). Example applications include users checking whether their copyrighted image, or personal photo with sensitive content, *etc*. is indexed, *i.e*. used by anyone else, without providing the query image itself. Such case of privacy protection is an example of "legal" motivation. A concept that bears resemblance to ours exists in the speech recognition, but in a malicious context. Carlini *et al*. [6] generate *hidden voice commands* that are imperceivable to human listeners but are interpreted as commands by devices. We investigate adversarial attacks beyond the white-box scenario, in which all the parameters and design choices of the retrieval system are known. Specifically, we analyze the cases of unknown indexing image resolution and unknown global pooling used in the network.

2. Related work

Adversarial attacks on image classification were introduced by Szegedy *et al*. [35]. Follow up approaches are categorized to *white-box* attacks [35, 12] if there is complete knowledge of the model or to *black-box* [28, 29] otherwise. Adversarial images are generated by various methods in the literature, such as optimization-based approaches using box-constrained L-BFGS optimizer [35], gradient descent with change of variable [7]. A fast gradient sign method [12] and variants [18, 10] are designed to be *fast* rather than optimal, while DeepFool [25] analytically derives an optimal solution method by assuming that neural networks are totally linear. All these approaches solve an optimization problem given a test image and its associated class in the case of non-targeted attacks or a test image and a target class in the case of targeted attacks. A universal non-targeted approach is proposed by Moosavi *et al*. [24], where an image-agnostic Universal Adversarial Perturbation (UAP) is computed and applied to unseen images to cause network misclassification.

Adversarial attacks on image retrieval are studied by recent work [19, 20, 37] in a non-targeted scenario for CNN-based approaches. Liu *et al*. [20] and Zheng *et al*. [37] adopt the optimization-based approach [35], while Li *et al*. [19] adopt the UAP [24]. Similar attacks on classical retrieval systems that are based on SIFT local descriptors [21] have been addressed in an earlier line of work by Do *et al*. [9, 8]. To the best of our knowledge, no existing work focuses on targeted adversarial attacks for image retrieval. Targeted attacks for nearest neighbors in high dimensional spaces are studied by Amsaleg *et al*. [2], where they directly perturb the high dimensional vectors and show that the high local intrinsic dimensionality results in high vulnerability.

3. Background

We provide the background for non-targeted and targeted adversarial attacks in the domain of image classification, then detail the basic components of CNN-based image retrieval approaches, and finally discuss non-targeted attacks for image retrieval. All variants presented in this section assume white-box access to the network classifier for classification or the feature extractor network for retrieval.

3.1. Image classification attacks

We denote the initial RGB image, called the *carrier image*, by tensor $\mathbf{x}_c \in [0,1]^{W\times H\times 3}$, and its associated label by $y_c \in \{1 \dots K\}$. A CNN trained for K-way classification, denoted by function $f : \mathbb{R}^{W\times H\times 3} \rightarrow \mathbb{R}^K$, produces vector $f(\mathbf{x}_c)$ comprising class confidence values. Adversarial attack methods for classification typically study the case of images with correct class prediction, *i.e*. $\arg\max_i f(\mathbf{x}_c)_i$ is equal to y_c, where $f(\mathbf{x}_c)_i$ is the i-th dimension of vector $f(\mathbf{x}_c)$. An adversary aims at generating *adversarial image* $\mathbf{x}_a$ that is visually similar to the carrier image but is classified incorrectly by f. The goal of the attack can vary [1] and corresponds to different loss functions optimizing $\mathbf{x} \in [0,1]^{W\times H\times 3}$.

Non-targeted misclassification is achieved by reducing the confidence for class y_c, while increasing for all other classes. It is achieved by minimizing loss function

$$L_{\text{nc}}(\mathbf{x}_c, y_c; \mathbf{x}) = -\ell_{\text{ce}}(f(\mathbf{x}), y_c) + \lambda \, ||\mathbf{x} - \mathbf{x}_c||^2. \quad (1)$$

Function $\ell_{\text{ce}}(f(\mathbf{x}), y_c)$ is the cross-entropy loss which is maximized to achieve misclassification. In this way, misclassification is performed to any wrong class. Term $||\mathbf{x} - \mathbf{x}_c||^2$ is called *carrier distortion* or simply *distortion* and is the squared l_2 norm of the perturbation vector $\mathbf{r} = \mathbf{x} - \mathbf{x}_c$. Other norms, such as l_∞, are also applicable [7].

Targeted misclassification has the goal of generating an adversarial image that gets classified into target class y_t. It is achieved by minimizing loss function

$$L_{\text{tc}}(\mathbf{x}_c, y_t; \mathbf{x}) = \ell_{\text{ce}}(f(\mathbf{x}), y_t) + \lambda \, ||\mathbf{x} - \mathbf{x}_c||^2. \quad (2)$$

In contrast to (1), cross-entropy loss is minimized w.r.t. the target class instead of maximized w.r.t. the carrier class.

Optimization of (1) or (2) generates the adversarial images given by

$$\mathbf{x}_a = \arg\min_{\mathbf{x}} L_{\text{nc}}(\mathbf{x}_c, y_c; \mathbf{x}), \quad (3)$$

or

$$\mathbf{x}_a = \arg\min_{\mathbf{x}} L_{\text{tc}}(\mathbf{x}_c, y_t; \mathbf{x}), \quad (4)$$

respectively. In the literature [35, 7], various optimizers such as Adam [16], or L-BFGS [5] are used. The box constraints, *i.e*. $\mathbf{x} \in [0,1]^{W\times H\times 3}$, are ensured by projected

gradient descent, clipped gradient descent, change of variables [7], or optimization algorithms that support box constraints such as L-BFGS. It is a common practice to perform line search for weight $\lambda > 0$ and keep the attack of minimum distortion. The optimization is initialized by the carrier image.

3.2. Image retrieval components

This work focuses on attacks on CNN-based image retrieval with global image descriptors. An image is mapped to a high dimensional descriptor by a CNN with a global pooling layer. The descriptor is consequently normalized to have unit l_2 norm. Then, retrieval from a large dataset w.r.t. a *query image* reduces to nearest neighbor search via inner product evaluation between the query descriptor and dataset descriptors. The model for descriptor extraction consists of the following components or parameters.

Image resolution: The input image $\mathbf{x}$ is re-sampled to image $\mathbf{x}^s$ to have the largest dimension equal to s.

Feature extraction: Image $\mathbf{x}^s$ is fed as an input to a Fully Convolutional Network (FCN), denoted by function $g : \mathbb{R}^{W\times H\times 3} \to \mathbb{R}^{w\times h\times d}$, which maps $\mathbf{x}^s$ to tensor $g(\mathbf{x}^s)$. When the image is fed at its original resolution we denote it by $g(\mathbf{x})$.

Pooling: A global pooling operation $h : \mathbb{R}^{w\times h\times d} \to \mathbb{R}^d$ maps the input tensor $g(\mathbf{x}^s)$ to descriptor $(h \circ g)(\mathbf{x}^s)$. We assume that l_2 normalization is included in this process, so that the output descriptor has unit l_2 norm. We consider various options for pooling, namely, max pooling (MAC) [32, 36], sum pooling (SPoC) [4], generalized mean pooling (GeM) [31], regional max pooling (R-MAC) [36], and spatially and channel-wise weighted sum pooling (CroW) [15]. The framework can be extended to multiple other variants [27, 23, 3].

Whitening: Descriptor post-processing is performed by function $w : \mathbb{R}^d \to \mathbb{R}^d$, which includes centering, whitening and l_2 re-normalization [31]. Finally, input image $\mathbf{x}^s$ is mapped to descriptor $(w \circ h \circ g)(\mathbf{x}^s)$.

For brevity we denote $\mathbf{g}_\mathbf{x} = g(\mathbf{x})$, $\mathbf{h}_\mathbf{x} = (h \circ g)(\mathbf{x})$, and $\mathbf{w}_\mathbf{x} = (w \circ h \circ g)(\mathbf{x})$. In the following, we consider an extraction model during the adversarial image optimization and another one during the testing of the retrieval/matching performance. In order to differentiate between the two cases we refer to the components of the former as *attack-model*, *attack-resolution*, *attack-FCN*, *attack-pooling* and *attack-whitening* and the latter as *test-model*, *test-resolution*, *test-FCN*, *test-pooling* and *test-whitening*.

3.3. Image retrieval attacks

Adversarial attacks for image retrieval are so far limited to the non-targeted case.

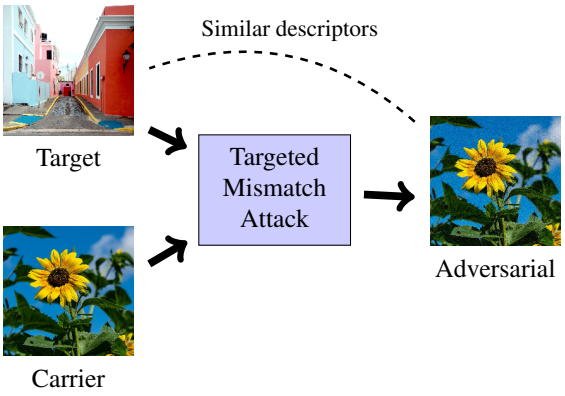


Figure 2. In targeted mismatch attacks an adversarial image is generated given a carrier and a target image. The adversarial image should match the descriptor of the target image but be visually dissimilar to the target; visual dissimilarity to the target is achieved via visual similarity to the carrier. The attack is formed by a retrieval query using the adversarial image, where the goal is to obtain identical results as with the target query while keeping the target image private.

Non-targeted mismatch aims at generating an adversarial image with small perturbation compared to the carrier image and descriptor that is dissimilar to that of the carrier. This is formulated by loss function

$$\begin{aligned} L_{\text{nr}}(\mathbf{x}_c;\mathbf{x}) &= \ell_{\text{nr}}(\mathbf{x},\mathbf{x}_c) &&+\lambda\,||\mathbf{x}-\mathbf{x}_c||^2 \\ &= \mathbf{h}_\mathbf{x}^\top \mathbf{h}_{\mathbf{x}_c} &&+\lambda\,||\mathbf{x}-\mathbf{x}_c||^2. \end{aligned} \tag{5}$$

The adversarial image is given by minimizer

$$\mathbf{x}_a = \arg\min_\mathbf{x} L_{\text{nr}}(\mathbf{x}_c;\mathbf{x}). \tag{6}$$

In this way, the adversary modifies images into their non-indexable counterpart. The exact formulation in (5) has not been addressed; the closest is the work of Li *et al.* [19] where they are seeking of a UAP by maximizing l_1 descriptor distance instead of minimizing cosine similarity.

4. Method

We formulate the problem of targeted mismatch attack and then propose various loss functions to address it and to construct concealed query images.

4.1. Problem formulation

The adversary tries to generate an adversarial image with the goal of using it as a (concealed) query for image retrieval instead of a *target image*. The goal is to obtain the same retrieval results without disclosing any information about the target image itself.

We assume a target image $\mathbf{x}_t \in \mathbb{R}^{W\times H\times 3}$ and a carrier image $\mathbf{x}_c$ with the same resolution (see Figure 2). The goal

of the adversary is to generate an adversarial image $\mathbf{x}_a$ that has high *descriptor similarity* but very low *visual similarity* to the target. Visual (human) dissimilarity is not straightforward to model; we model visual similarity w.r.t. another image, *i.e*. the carrier, instead. We refer to this problem as *targeted mismatch attack* and the corresponding loss function is given by

$$L_{\text{tr}}(\mathbf{x}_c, \mathbf{x}_t; \mathbf{x}) = \ell_{\text{tr}}(\mathbf{x}, \mathbf{x}_t) + \lambda \, ||\mathbf{x} - \mathbf{x}_c||^2. \quad (7)$$

In Section 4.2 we propose different instantiations of the *performance loss* ℓ_{tr} according to the known and unknown components of the test-model.

4.2. Targeted mismatch attacks

In all the following, we assume a white-box access to the FCN, while the whitening is assumed unknown and is totally ignored during the optimization of the adversarial image; its impact on the attack is evaluated by adding it to the test-model. In general, if all the parameters of the test-model are known, the task is to generate an adversarial image that reproduces the descriptor of the target image. Then, nearest neighbor search will retrieve identical results as if querying with the target image. Choosing a different performance loss introduces invariance or robustness to some parameters of the attacked retrieval system, when these parameters are unknown. We list different performance loss functions used to minimize (7).

Global descriptor. Loss function

$$\ell_{\text{desc}}(\mathbf{x}, \mathbf{x}_t) = 1 - \mathbf{h}_{\mathbf{x}}^\top \mathbf{h}_{\mathbf{x}_t}. \quad (8)$$

is suitable when all parameters of the retrieval system are known, including the pooling, and when the image is processed by the neural network at its original resolution. Pooling function h is MAC, SPoC, or GeM in our experiments.

Activation tensor. In this scenario, the output of the FCN should be the same for the adversarial and target image, at the original resolution. This is achieved by minimizing the mean squared difference of the two activation tensors

$$\ell_{\text{tens}}(\mathbf{x}, \mathbf{x}_t) = \frac{||\mathbf{g}_{\mathbf{x}} - \mathbf{g}_{\mathbf{x}_t}||^2}{w \cdot h \cdot d}. \quad (9)$$

Identical tensors guarantee identical descriptors computed on top of these tensors, including those where spatial information is taken into account. This covers all global or regional pooling operations, and even deep local features, *e.g*. DELF [26]. However, our experiments show that preserving the activation tensor may result in transferring the target's visual content on the adversarial image (see Figure 7). Further, the visual appearance of the target image can be partially recovered by inverting [22] the activation tensor of the adversarial image.

Activation histogram. Preserving channel-wise first order statistics of the activation tensor, at the original resolution, is a weaker constraint than preserving the exact activation tensor. It guarantees identical descriptors for all global pooling operations that ignore spatial information. Activation histogram loss function is defined as

$$\ell_{\text{hist}}(\mathbf{x}, \mathbf{x}_t) = \frac{1}{d} \sum_{i=1}^{d} ||u(\mathbf{g}_{\mathbf{x}}, \mathbf{b})_i - u(\mathbf{g}_{\mathbf{x}_t}, \mathbf{b})_i||, \quad (10)$$

where $u(\mathbf{g}_{\mathbf{x}}, \mathbf{b})_i$ is the histogram of activations from the i-th channel of $\mathbf{g}_{\mathbf{x}}$ and $\mathbf{b}$ is the vector of histogram bin centers. Histograms are created with soft assignment by an RBF kernel[2]. Compared with the tensor case, the histogram optimization does not preserve the spatial distribution, is significantly faster, and does not suffer from undesirable disclosure artifacts.

Different image resolution. We require an adversarial image at the original resolution of the target ($W \times H$), which when down-sampled to resolution s, it retrieves similar results as the target image down-sampled to the same resolution. This is achieved by loss function

$$L^s_{\text{tr}}(\mathbf{x}, \mathbf{x}_t; \mathbf{x}) = \ell_{\text{tr}}(\mathbf{x}^s, \mathbf{x}^s_t) + \lambda \, ||\mathbf{x} - \mathbf{x}_c||^2, \quad (11)$$

where ℓ_{tr} can be any of the descriptor, tensor, or histogram based performance loss functions. Note that (11) is different from (7), the performance loss is computed from resampled images, while the distortion loss is still on the original images.

A common down-sampling method used in CNNs is bilinear interpolation. We have observed that different implementations of such a layer result in different descriptors. The difference is caused by the presence of high-frequencies in the high-resolution image. The adversarial perturbation tends to be high-frequency, therefore different down-sampling results may significantly alternate the result of attack. In order to reduce the sensitivity to down-sampling, we introduce high-frequency removal by Gaussian blurring in the optimization. Instead of (11), the following loss is used

$$L^{\hat{s}}_{\text{tr}}(\mathbf{x}, \mathbf{x}_t; \mathbf{x}) = \ell_{\text{tr}}(\mathbf{x}^{\hat{s}}, \mathbf{x}^{\hat{s}}_t) + \lambda \, ||\mathbf{x} - \mathbf{x}_c||^2, \quad (12)$$

where $\mathbf{x}^{\hat{s}}$ is image $\mathbf{x}$ blurred with Gaussian kernel with σ_b and then down-sampled. Our experiments show, that blurring plays an important role when the attack-resolution s does not exactly match the test-resolution s', *i.e*. $s' = s + \Delta$.

Ensembles. We perform the adversarial optimization for a combination of the aforementioned loss functions by minimizing their sum. Some examples follow.

[2] We use $e^{\frac{(x-b)^2}{2\sigma^2}}$, where $\sigma = 0.1$, x is a scalar activation normalized by the maximum activation value of the target, and b is the bin center. We uniformly sample bin centers in [0,1] with step equal to 0.05.

- The test-pooling operation is unknown but there is a set $\mathcal{P}$ of possible pooling operations. Minimization of (7) is performed for performance loss

$$\ell_{\mathcal{P}}(\mathbf{x}, \mathbf{x}_t) = \frac{\sum_{p \in \mathcal{P}} \ell_p(\mathbf{x}, \mathbf{x}_t)}{|\mathcal{P}|}. \quad (13)$$

- The test-resolution is unknown. Joint optimization for a set $\mathcal{S}$ of resolutions is performed with

$$L^{S}_{\text{tr}}(\mathbf{x}, \mathbf{x}_t; \mathbf{x}) = \frac{\sum_{s \in \mathcal{S}} \ell_{\text{tr}}(\mathbf{x}^s, \mathbf{x}^s_t)}{|S|} + \lambda \, ||\mathbf{x} - \mathbf{x}_c||^2. \quad (14)$$

Any performance loss ℓ_{tr} is used, with or without blurring.

4.3. Optimization

The optimization is performed with Adam and projected gradient descent is used to apply the box constraints, *i.e.* $\mathbf{x} \in [0,1]^{W \times H \times 3}$. The adversarial image is initialized by the carrier image, while after every update its values are clipped to be in $[0, 1]$. The adversarial image is given by

$$\mathbf{x}_a = \arg\min_{\mathbf{x}} L_{\text{tr}}(\mathbf{x}_c, \mathbf{x}_t; \mathbf{x}), \quad (15)$$

where L_{tr} can be L_{desc} (with "desc" equal to MAC, SPoC, or GeM), $L_{\mathcal{P}}$, L_{hist}, or L_{tens} according to the variant, while the variants with multiple scales are denoted *e.g.* by $L^{\mathcal{S}}_{\text{hist}}$ without blur or $L^{\hat{\mathcal{S}}}_{\text{hist}}$ with blur.

5. Experiments

Given a test architecture, we validate the success of the targeted mismatch attack in two ways. First, by measuring the cosine similarity between descriptors of the adversarial image $\mathbf{x}_a$ and the target $\mathbf{x}_t$ (should be as high as possible), and second, by using $\mathbf{x}_a$ as an image retrieval query and compare its performance with that of the target query (should be as close as possible)[3].

5.1. Datasets and evaluation protocol

We perform experiments on four standard image retrieval benchmarks, namely Holidays [14], Copydays [11], $\mathcal{R}$Oxford [30], and $\mathcal{R}$Paris [30]. They all consist of a set of query images and a set of database images, while the ground-truth denotes which are the relevant dataset images per query. We choose to perform attacks only with the first 50 queries for Holidays and Copydays to form adversarial attack benchmarks of reasonable size, while for $\mathcal{R}$Oxford and $\mathcal{R}$Paris we keep all 70 of them and use the *Medium* evaluation setup. All queries are used as targets to form an attack and retrieval performance is measured with mean Average Precision (mAP). Unless otherwise stated we use the "flower" of Figure 1 as the carrier; it is cropped to match the aspect ratio of the target. All images are re-sampled to have the largest dimension equal to 1024, this is the original image resolution. $\mathcal{R}$Oxford and $\mathcal{R}$Paris are treated differently than the other two due to the cropped image queries; the cropped image region that defines the query is used as a target and the relative scale change between queries and database images should be preserved not to affect the ground truth. When the image resolution for descriptor extraction is different than the original one, we down-sample the cropped image with the same scaling factor that the uncropped one should have been down-sampled with.

[3]Public implementation: https://github.com/gtolias/tma

5.2. Implementation details and experimental setup

We set the learning rate equal to 0.01 in all our experiments and perform 100 iterations for L_{desc} and L_{hist}, while 1000 iterations for L_{tens}. If there is no convergence, we decrease the learning rate by a factor of 5 and increase the number of iterations by a factor 2 and re-start. We normalize the distortion term with the dimensionality of $\mathbf{x}$; this is skipped in the loss function of Sections 3 and 4 for brevity. Moreover, in order to handle the different range of activations for different FCNs, we normalize activation tensors with the maximum target activation before computing the mean squared error in (9). Image blurring at resolution s in (12) is performed by a Gaussian kernel with $\sigma_b = 0.3 \max(W, H)/s$. The exponent of GeM pooling is always set to 3.

Setting $\lambda = 0$ provides a trivial solution to (7), *i.e.* $\mathbf{x}_a = \mathbf{x}_t$. However, we observe that initialization by $\mathbf{x}_c$ converges to local minima that are significantly closer to $\mathbf{x}_c$ than $\mathbf{x}_t$ even for the case of $\lambda = 0$. In this way, we satisfy the non-disclosure constraint, *i.e.* the adversarial image is visually dissimilar to the target, and do not sacrifice the performance loss. The image distortion w.r.t. to the carrier image does not sacrifice the goal of concealing the target and preserving user privacy. Therefore, in our experiments we mostly focus on cases with $\lambda = 0$, but also validate cases with $\lambda > 0$ to show the impact of the distortion term or in order to promote the non-disclosure constraint for the case of L_{tens}.

We experiment with different loss functions for targeted mismatch attacks. We define $\mathcal{S}_0$, $\mathcal{S}_1$, $\mathcal{S}_2$, and $\mathcal{S}_3$ sets of attack-resolutions[4]. We denote AlexNet [17], ResNet18 [13], and VGG16 [34] by $\mathcal{A}$, $\mathcal{R}$, and $\mathcal{V}$, respectively. We use networks that are pre-trained on ImageNet [33] and keep only their fully convolutional part. The AlexNet and ResNet18 ensemble is denoted by $\mathcal{E}$; mean loss over two networks is minimized. We report the triplet attack-model, loss function and value of λ to denote the kind of adversarial optimization, for example $(\mathcal{A}, L^{\mathcal{S}_1}_{\text{hist}}, 0)$. For testing, we report the triplet test-model, test-pooling and test-resolution, for example $[\mathcal{A}, \text{GeM}, \mathcal{S}_0]$.

[4]$\mathcal{S}_0 = \{1024\}$, $\mathcal{S}_1 = \mathcal{S}_0 \cup \{300, 400, 500, 600, 700, 800, 900\}$, $\mathcal{S}_2 = \mathcal{S}_1 \cup \{350, 450, 550, 650, 750, 850, 950\}$, $\mathcal{S}_3 = \mathcal{S}_0 \cup \{262, 289, 319, 351, 387, 427, 470, 518, 571, 630, 694, 765, 843, 929\}$

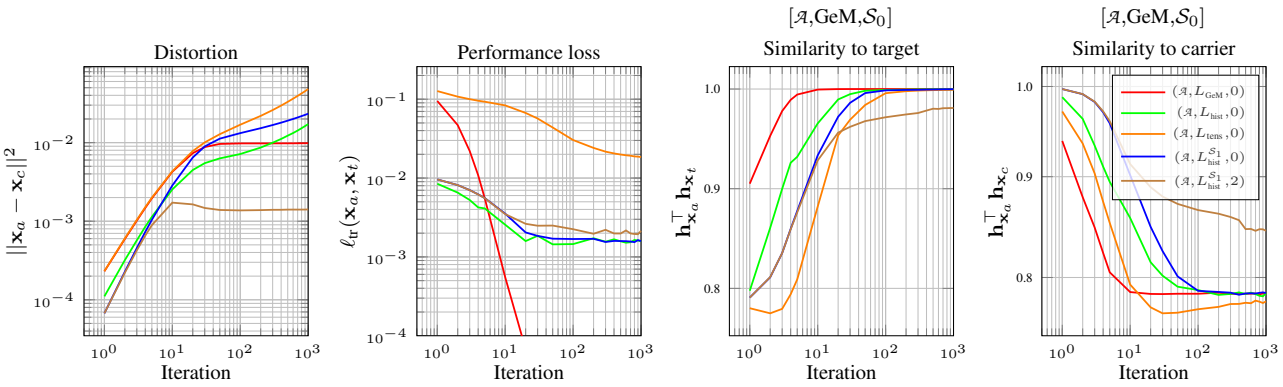


Figure 3. Adversarial images are generated with different loss functions and various measurements are reported as they evolve with the number of iterations. The presented measurements are: (a) the distortion w.r.t. the carrier image, (b) the performance loss from (7), (c) descriptor similarity of the adversarial image to the target for test case [$\mathcal{A}$,GeM,$\mathcal{S}_0$] and (d) descriptor similarity of the adversarial image to the carrier for test case [$\mathcal{A}$,GeM,$\mathcal{S}_0$]. The target and carrier images are the ones shown in Figure 7.

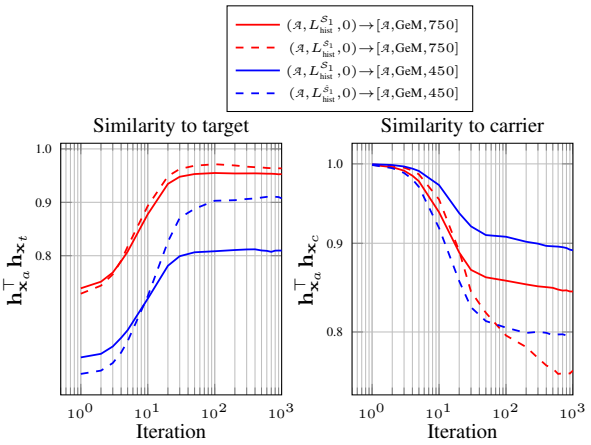


Figure 4. Descriptor similarity between the adversarial image and the target or the carrier as it evolves with the number of iterations. Comparison for test-resolutions that are not in the attack-resolutions for cases without (solid) and with (dashed) blurring. The adversarial optimization (left) and the test model (right) are denoted with $\rightarrow$. The target and carrier images are from Figure 7.

h \ L_{tr}	Original	L_{GeM}	$L_{\mathcal{P}}$	L_{hist}	L_{tens}
	mAP	mAP difference to original			
GeM	41.3	−0.0	−0.0	−0.2	−0.1
MAC	37.0	−0.5	−0.0	−0.8	−0.0
SPoC	32.9	−4.4	−0.1	−0.1	−0.7
R-MAC	44.1	−1.2	−0.5	−0.7	−0.0
CroW	38.2	−1.3	−0.4	−0.2	−0.0
	$\mathbf{x}_t^\top \mathbf{x}_a$				
GeM	1.000	1.000	1.000	0.997	0.998
MAC	1.000	0.972	1.000	0.985	0.996
SPoC	1.000	0.909	1.000	0.999	0.996
R-MAC	1.000	0.972	0.978	0.979	0.997
CroW	1.000	0.968	0.994	0.995	0.998

Table 1. Performance evaluation for attacks based on AlexNet and various loss functions optimized at the original image resolution $\mathcal{S}_0$. Testing is performed on [$\mathcal{A}$,desc,$\mathcal{S}_0$] for multiple types of descriptor/pooling. Mean average Precision on $\mathcal{R}$Paris and mean descriptor similarity between the adversarial image and the target across all queries is reported. *Original* corresponds to queries without attack.

5.3. Results

For each adversarial image we perform the following measurements. We compute its similarity to the target and to the carrier by cosine similarity of the corresponding descriptors, we measure the carrier distortion and, lastly, we perform an attack by using it as a query and measure the average precision which is compared to that of the target.

Optimization iterations. We perform the optimization for different loss functions and report the measurements over iteration in Figure 3. Optimizing global descriptor or histogram converges much faster than the tensor case and results in significantly lower distortion. This justifies our choice of using a lower number of iterations for the two approaches. Increasing the value of λ keeps the distortion lower but sacrifices the performance loss, as expected.

In Figure 4 we show how the similarity to the target and the carrier evolves for test-resolution that is not included in the set of attack-resolutions. Processing the images with image blurring offers significant improvements, especially for the smaller resolutions.

Robustness to unknown test-pooling. In Table 1 we present the evaluation comparison for different loss functions and test-pooling. The case of same attack-resolution and test-resolution is examined first. If the test-pooling is directly optimized (L_{GeM} or $L_{\mathcal{P}}$ case), then perfect performance is achieved. The histogram and tensor based approaches both perform well for a variety of test-descriptors.

Robustness to unknown test-resolution. Cases with different attack-resolution and test-resolution are evaluated and results are presented in Figure 5. Resolutions that

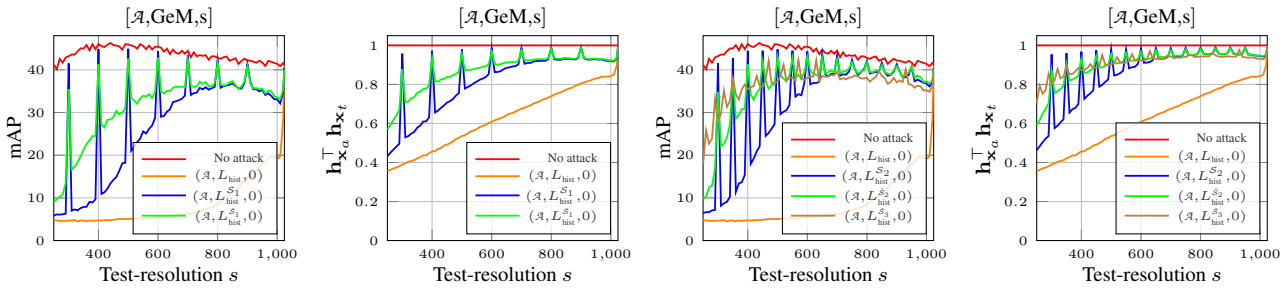


Figure 5. Performance evaluation for attack based on AlexNet and a set of attack-resolutions. Mean average Precision on $\mathcal{R}$Paris and mean descriptor similarity between the adversarial image and the target across all queries is shown for increasing test-resolution. Comparison using different sets of attack-resolutions and comparison for optimization without ($\mathcal{S}$) and with ($\hat{\mathcal{S}}$) image blurring.

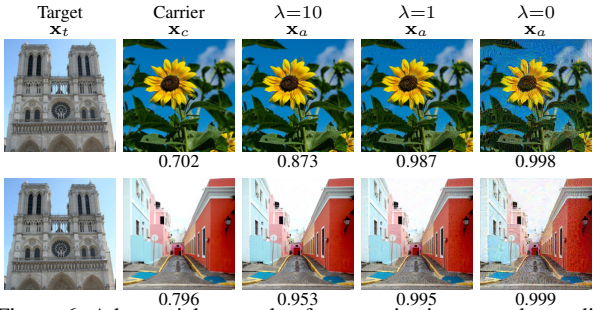


Figure 6. Adversarial examples for a carrier image and two different targets while optimizing $(\mathcal{A},L^{\hat{\mathcal{S}}_2}_{\text{hist}},\lambda)$ for various values of λ. Descriptor similarity is reported for $[\mathcal{A},\text{GeM},\mathcal{S}_0]$.

are not part of the attack-resolutions suffer from significant drop in performance when blurring is not performed, while blurring improves it. We observe how the retrieval performance and descriptor similarity between adversarial image and target are correlated. Moreover, the optimization for multiple resolutions is clearly better than that for single resolution, while logarithmic sampling of attack-resolutions ($\mathcal{S}_3$) significantly improves the performance for very small test-resolution but harms it for larger ones.

Impact of the distortion term. We evaluate $[\mathcal{A},\text{GeM},\mathcal{S}_0]$ on queries of $\mathcal{R}$Paris for $(\mathcal{A},L^{\hat{\mathcal{S}}_2}_{\text{hist}},\lambda)$ and λ equal to 0, 0.1, 1, 10. The average similarity between the adversarial image and the target is 0.990, 0.987, 0.956, and 0.767, respectively, while the average distortion is 0.0177, 0.0083, 0.0026, and 0.0008, respectively. Examples of adversarial images are shown in Figure 6.

Impact of the whitening in the test-model. We now consider the case that the test-model includes descriptor whitening. The whitening is unknown during the time of the adversarial optimization. We evaluate the performance on $\mathcal{R}$Paris while learning whitening with PCA on $\mathcal{R}$Oxford. Testing without whitening and $[\mathcal{A},\text{GeM},\mathcal{S}_0]$ or $[\mathcal{A},\text{GeM},768]$ achieves 41.3, and 40.2 mAP, respectively. After applying whitening the respective performances increase to 47.5 and 48.0 mAP. Attacks with $(\mathcal{A},L^{\hat{\mathcal{S}}_2}_{\text{hist}},0)$ achieve 40.2, and 39.4 mAP when tested in the aforementioned cases without whitening. Attacks with $(\mathcal{A},L^{\hat{\mathcal{S}}_2}_{\text{hist}},0)$ achieve 47.3, and 42.9 mAP when tested in the aforementioned cases with whitening. Whitening introduces additional challenges, but the attacks seem effective with slightly reduced performance.

Concealing/revealing the target. We generate adversarial images for different loss functions and show examples in Figure 7. The corresponding tensors show that spatial information is only preserved in the tensor-based loss function. The tensor-based approach requires the distortion term to avoid revealing visual structures of the target (adversarial images in 6-th and 7-th column). We now pose the question "can the FCN activations of the adversarial image reveal the content of the target?". To answer, we invert tensor $\mathbf{g}_{\mathbf{x}_a}$ at multiple resolutions using the method of Mahendran and Vedaldi [22]. The tensor-based approach indeed reveals the target's content in the reconstruction, while no other approach does. This highlights the benefits of the proposed histogram-based optimization. Note that the reconstructed image resembles the target less if the resolutions used in the reconstruction are not the same as the attack-resolutions (rightmost column).

Timings. We report the average optimization time per target image on Holidays dataset and on a single GPU (Tesla P100) for some indicative cases. Optimizing $(\mathcal{A},L_{\text{GeM}},0)$, $(\mathcal{A},L^{\hat{\mathcal{S}}_1}_{\text{GeM}},0)$, $(\mathcal{A},L^{\hat{\mathcal{S}}_1}_{\text{hist}},0)$, $(\mathcal{A},L^{\hat{\mathcal{S}}_2}_{\text{hist}},0)$, and $(\mathcal{A},L^{\hat{\mathcal{S}}_1}_{\text{tens}},0)$ takes 1.9, 7.5, 12.3, 22.9, and 68.4 seconds, respectively. Using ResNet18 $(\mathcal{R},L_{\text{GeM}},0)$ and $(\mathcal{R},L^{\hat{\mathcal{S}}_2}_{\text{hist}},0)$ take 3.9 and 40.6 seconds, respectively.

Multiple attacks. We show results of multiple attacks in Table 2. We present the original retrieval performance together with the difference in the performance caused by the attack. It summarizes the robustness of the histogram and tensor based optimization to unknown pooling operations. It emphasizes the challenges of unknown test-resolution and the impact of the blurring; this outcome can be useful in various different attack models. The very last row suggests that transferring attacks to different FCNs (optimizing on $\mathcal{E}$, which includes $\mathcal{A}$ and $\mathcal{R}$, and testing on $\mathcal{V}$) is hard to achieve; it is harder than for classification [35].

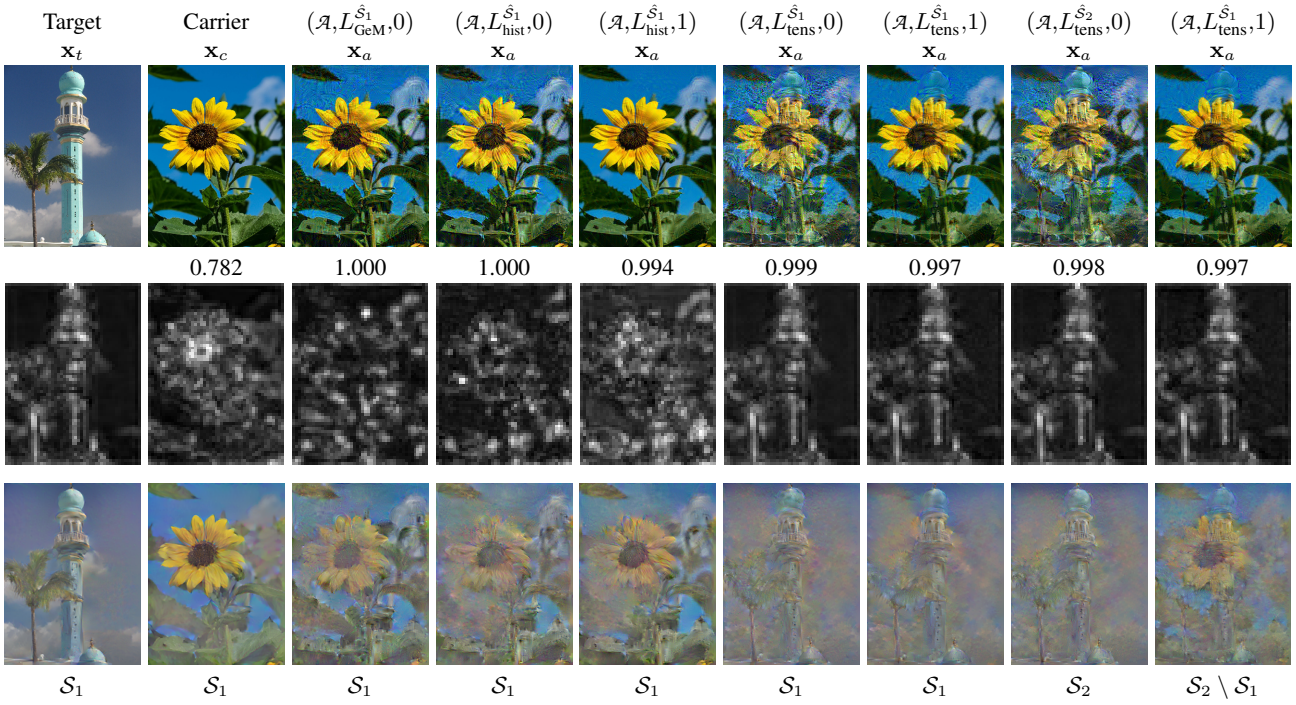


Figure 7. Target, carrier and adversarial images for different variants (top image row), a summary of tensor $\mathbf{g}_{\mathbf{x}}$ by depth-wise maximum (middle image row) and the inversion of $\mathbf{g}_{\mathbf{x}_t}$, $\mathbf{g}_{\mathbf{x}_c}$, or $\mathbf{g}_{\mathbf{x}_a}$, respectively, over multiple resolutions (bottom image row). The resolutions for inversion are reported below the bottom row. The tensor inversion shows whether the target, or any information about it, can be reconstructed from the adversarial image. The first two inversions are presented as a reference. Descriptor similarity to the target is reported below the first image row for $[\mathcal{A}$,GeM,1024$]$.

Attack	Test	$\mathcal{R}$Oxford	$\mathcal{R}$Paris	Holidays	Copydays
$(\mathcal{A},L^{\hat{\mathcal{S}}_2}_{\text{hist}},0)$	$[\mathcal{A}$,GeM,$\mathcal{S}_0]$	26.9 / +0.2	41.3 / -1.2	81.5 / +0.2	80.4 / -0.4
$(\mathcal{R},L^{\hat{\mathcal{S}}_2}_{\text{GeM}},0)$	$[\mathcal{R}$,GeM,$\mathcal{S}_0]$	21.5 / -0.7	46.9 / -0.4	82.9 / -0.3	69.3 / -0.7
	$[\mathcal{R}$,GeM,768$]$	24.0 / -2.5	48.0 / -3.9	81.7 / -4.4	75.6 / -2.8
	$[\mathcal{R}$,GeM,512$]$	22.4 / -6.7	49.7 /-11.1	82.8 / -0.6	82.1 /-10.7
$(\mathcal{R},L^{\hat{\mathcal{S}}_2}_{\text{hist}},0)$	$[\mathcal{R}$,GeM,$\mathcal{S}_0]$	21.5 / -1.2	46.9 / -1.9	82.9 / -0.6	69.3 / -1.3
	$[\mathcal{R}$,GeM,768$]$	24.0 / -3.7	48.0 / -7.2	81.7 / -2.3	75.6 / -7.1
	$[\mathcal{R}$,GeM,512$]$	22.4 /-11.2	49.7 /-20.7	82.8 /-17.1	82.1 /-20.6
$(\mathcal{R},L^{\hat{\mathcal{S}}_2}_{\text{hist}},0)$	$[\mathcal{R}$,GeM,$\mathcal{S}_0]$	21.5 / -1.4	46.9 / -1.8	82.9 / -2.4	69.3 / -1.3
	$[\mathcal{R}$,GeM,768$]$	24.0 / -5.3	48.0 / -6.0	81.7 / -1.7	75.6 / -4.2
	$[\mathcal{R}$,GeM,512$]$	22.4 / -7.4	49.7 /-11.9	82.8 / -4.9	82.1 /-11.3
$(\mathcal{R},L^{\hat{\mathcal{S}}_2}_{\mathcal{P}},0)$	$[\mathcal{R}$,CroW,$\mathcal{S}_0]$	22.0 / -1.1	45.0 / -0.5	81.0 / +0.9	67.0 / -1.6
$(\mathcal{R},L^{\hat{\mathcal{S}}_2}_{\text{hist}},0)$		22.0 / -0.3	45.0 / -0.8	81.0 / +1.3	67.0 / -1.0
$(\mathcal{R},L^{\hat{\mathcal{S}}_2}_{\text{tens}},0)$		22.0 / -0.7	45.0 / -0.0	81.0 / -0.6	67.0 / -3.0
$(\mathcal{E},L^{\hat{\mathcal{S}}_2}_{\text{hist}},0)$	$[\mathcal{A}$,GeM,$\mathcal{S}_0]$	26.9 / -2.3	41.3 / -5.5	81.5 / -3.1	80.4 / -4.9
	$[\mathcal{R}$,CroW,$\mathcal{S}_0]$	22.0 / -1.1	45.0 / -0.8	81.0 / +1.0	67.0 / -0.8
	$[\mathcal{V}$,GeM,$\mathcal{S}_0]$	38.1 /-34.9	54.0 /-47.4	85.7 /-72.6	80.0 /-72.9

Table 2. Performance evaluation for multiple attacks, test-models, and datasets. Mean average Precision over the original queries, together with the mAP difference to the original caused by the attack, is reported. The parameters of the adversarial optimization during the attack are shown in the leftmost column, while the type of test-model used is shown in the second column.

6. Conclusions

We have introduced the problem of targeted mismatch attack for image retrieval and address it in order to construct concealed query images instead of the initial intended query. We show that optimizing the first order statistics is a good way to generate images that result in the desired descriptors without disclosing the content of the intended query. We analyze the impact of image re-sampling, which is a natural component of image retrieval systems and reveal the benefits of simple image blurring in the adversarial image optimization. Finally, we show that transferring attacks to new FCNs are much more challenging than their image classification counterparts.

We focused on concealing the query in a privacy preserving scenario. In a malicious scenario the adversary might try to corrupt the search results by targeted mismatch attacks on indexed images. This is an interesting direction and an open research problem.

Acknowledgments Work supported by GAČR grant 19-23165S and OP VVV funded project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics".

References

[1] Naveed Akhtar and Ajmal Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access*, 2018. 2

[2] Laurent Amsaleg, James Bailey, Dominique Barbe, Sarah Erfani, Michael E Houle, Vinh Nguyen, and Miloš Radovanović. The vulnerability of learning to adversarial perturbation increases with intrinsic dimensionality. In *IEEE Workshop on Information Forensics and Security (WIFS)*, 2017. 2

[3] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. In *CVPR*, 2016. 3

[4] Artem Babenko and Victor Lempitsky. Aggregating deep convolutional features for image retrieval. In *ICCV*, 2015. 3

[5] Richard H Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SISC*, 1995. 2

[6] Nicholas Carlini, Pratyush Mishra, Tavish Vaidya, Yuankai Zhang, Micah Sherr, Clay Shields, David Wagner, and Wenchao Zhou. Hidden voice commands. In *USENIX Security*, 2016. 2

[7] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *SSP*, 2017. 1, 2, 3

[8] Thanh-Toan Do, Ewa Kijak, Laurent Amsaleg, and Teddy Furon. Security-oriented picture-in-picture visual modifications. In *ICMR*, 2012. 2

[9] Thanh-Toan Do, Ewa Kijak, Teddy Furon, and Laurent Amsaleg. Challenging the security of content-based image retrieval systems. In *MMSP*, 2010. 2

[10] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *CVPR*, 2018. 1, 2

[11] Matthijs Douze, Hervé Jégou, Harsimrat Sandhawalia, Laurent Amsaleg, and Cordelia Schmid. Evaluation of GIST descriptors for web-scale image search. In *CIVR*, 2009. 5

[12] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015. 2

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 5

[14] Herve Jégou, Matthijs Douze, and Cordelia Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *ECCV*, 2008. 5

[15] Yannis Kalantidis, Clayton Mellina, and Simon Osindero. Cross-dimensional weighting for aggregated deep convolutional features. In *ECCVW*, 2016. 3

[16] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 2

[17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012. 5

[18] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *ICLRW*, 2017. 2

[19] Jie Li, Rongrong Ji, Hong Liu, Xiaopeng Hong, Yue Gao, and Qi Tian. Universal perturbation attack against image retrieval. In *arXiv*, 2018. 1, 2, 3

[20] Zhuoran Liu, Zhengyu Zhao, and Martha Larson. Who's afraid of adversarial queries? the impact of image modifications on content-based image retrieval. In *arXiv*, 2019. 1, 2

[21] David Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004. 2

[22] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *CVPR*, 2015. 4, 7

[23] Eva Mohedano, Kevin McGuinness, Noel E O'Connor, Amaia Salvador, Ferran Marques, and Xavier Giro-i Nieto. Bags of local convolutional features for scalable instance search. In *ICMR*, 2016. 3

[24] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *CVPR*, 2017. 1, 2

[25] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. DeepFool: a simple and accurate method to fool deep neural networks. In *CVPR*, 2016. 1, 2

[26] Hyeonwoo Noh, Andre Araujo, Jack Sim, Tobias Weyand, and Bohyung Han. Large-scale image retrieval with attentive deep local features. In *ICCV*, 2017. 4

[27] Eng-Jon Ong, Sameed Husain, and Miroslaw Bober. Siamese network of deep fisher-vector descriptors for image retrieval. In *arXiv*, 2017. 3

[28] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. In *arXiv*, 2016. 2

[29] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *ASIACCS*, 2017. 2

[30] Filip Radenović, Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondřej Chum. Revisiting Oxford and Paris: Large-scale image retrieval benchmarking. In *CVPR*, 2018. 5

[31] Filip Radenović, Giorgos Tolias, and Ondřej Chum. Fine-tuning CNN image retrieval with no human annotation. *PAMI*, 2018. 3

[32] Ali Sharif Razavian, Josephine Sullivan, Stefan Carlsson, and Atsuto Maki. Visual instance retrieval with deep convolutional networks. *ITE Trans. MTA*, 2016. 3

[33] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 2015. 5

[34] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *arXiv*, 2014. 5

[35] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *ICLR*, 2014. 1, 2, 7

[36] Giorgos Tolias, Ronan Sicre, and Hervé Jégou. Particular object retrieval with integral max-pooling of CNN activations. In *ICLR*, 2016. 3

[37] Zhedong Zheng, Liang Zheng, Zhilan Hu, and Yi Yang. Open set adversarial examples. In *arXiv*, 2018. 1, 2

XVII Efficient Diffusion on Region Manifolds: Recovering Small Objects with Compact CNN Representations

Title: Efficient Diffusion on Region Manifolds: Recovering Small Objects with Compact CNN Representations
Authors: A. Iscen, G. Tolias, Y. Avrithis, T. Furon, O. Chum
Published at: CVPR 2017

Efficient Diffusion on Region Manifolds: Recovering Small Objects with Compact CNN Representations

Ahmet Iscen[1] Giorgos Tolias[2] Yannis Avrithis[1] Teddy Furon[1] Ondřej Chum[2]
[1]Inria Rennes [2]VRG, FEE, CTU in Prague
{ahmet.iscen,ioannis.avrithis,teddy.furon}@inria.fr
{giorgos.tolias,chum}@cmp.felk.cvut.cz

Abstract

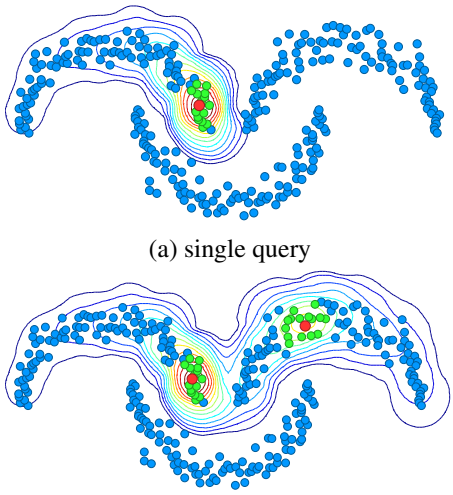
Query expansion is a popular method to improve the quality of image retrieval with both conventional and CNN representations. It has been so far limited to global image similarity. This work focuses on diffusion, a mechanism that captures the image manifold in the feature space. The diffusion is carried out on descriptors of overlapping image regions rather than on a global image descriptor like in previous approaches. An efficient off-line stage allows optional reduction in the number of stored regions. In the on-line stage, the proposed handling of unseen queries in the indexing stage removes additional computation to adjust the precomputed data. We perform diffusion through a sparse linear system solver, yielding practical query times well below one second.

Experimentally, we observe a significant boost in performance of image retrieval with compact CNN descriptors on standard benchmarks, especially when the query object covers only a small part of the image. Small objects have been a common failure case of CNN-based retrieval.

1. Introduction

Object search is a key tool behind a number of applications like content based image collection browsing [56, 34], visual localization [46, 1], and 3D reconstruction [22, 47]. Many applications benefit from retrieving images taken from various viewing angles and under different illumination, *e.g*. more information for the user while browsing, localization in day and night, and complete 3D models. Each image is represented by one or more descriptors designed or learned to exhibit a certain degree of invariance to imaging conditions. Retrieval is formulated as a nearest neighbor search in the descriptor space, performed by approximate methods [36, 25, 29, 5].

While collections of local descriptors provide good invariance, global descriptors like VLAD [26] have smaller memory footprint, but are more prone to locking onto the clutter. This mainly holds when the queried object covers only a small part of the image. In case of global CNN descriptors, the invariance is partially designed by global max [3, 52] or sum [30, 4] pooling layers or multi-scale querying [19], and partially learned by the choice of the training data. Robustness to background clutter is improved by computing descriptors over object proposals [35, 18, 57] or over a fixed grid of regions [52]. Better performance is observed at a cost of increased memory footprint [44].



(a) single query

(b) multiple queries

Figure 1. Diffusion on a synthetic dataset in $\mathbb{R}^2$. Dataset points, query points and their k-nearest neighbors are shown in blue, red, and green respectively. Contour lines correspond to ranking scores after diffusion. In this work, points are region descriptors.

In image collections, objects are depicted in various conditions. As a consequence, query and relevant images are often connected by a sequence of images, where consecutive images are similar. The descriptors of these images form a manifold in the descriptor space. Even though the images of the sequence contain the same object, the descriptors may be completely unrelated after a certain point.

This idea has been first exploited by Chum et al. [8] who introduce query expansion. The average query expansion

(AQE) is now used as a standard tool in image retrieval, due to its efficiency and significant performance boost. However, AQE only explores the neighborhood of very similar images. Recursive and scale-band recursive methods [8] further improve the results by explicitly crawling the image manifold. This is at a cost of increased query time.

Query expansion exploits the manifold of images at query time—starting from nearest neighbors of the query and using these neighbors to issue new queries. On the other hand, *diffusion* [39, 64, 13] is based on a neighborhood graph of the dataset that is constructed off-line and efficiently uses this information at query time to search on the manifold in a principled way.

We make the following contributions:

- We introduce a *regional diffusion mechanism*, which handles one or more query vectors at the same cost. There is one vector per region and a few regions per image so that constructing and storing the graph is tractable. This approach significantly improves retrieval of small objects and cluttered scenes.
- In diffusion mechanisms [39, 64, 13], query vectors are usually part of the dataset and available at the indexing stage. A novel approach to *unseen queries* with no computational overhead is proposed.
- Though a closed form solution is known to exist, it has been explicitly avoided so far [13]. We show that the commonly used alternative is in fact a well known iterative linear system solver. Since the relevant matrix is sparse and positive definite, the *conjugate gradient* method is more efficient resulting in practical query times well below one second.
- To study the dependence of performance on relative object size, we experiment on INSTRE dataset [55], which has not received much attention so far. We propose a new evaluation protocol that is in line with other well known datasets and provide a rich set of baselines to facilitate future comparisons.

Searching in parallel in more than one manifolds via diffusion and using the nearest neighbors of unseen queries are illustrated in Figure 1.

The remaining text is structured as follows. Sections 2 and 3 discuss related work and background respectively, focusing on diffusion mechanisms. Sections 4 and 5 present our contributions in detail and the experimental body.

2. Related work

This section discusses existing query expansion or reranking methods. We also review the concept of diffusion in computer vision and image retrieval in particular. Apart from AQE [8], none of these methods has been applied to retrieval in the context of convolutional features.

Query expansion. A variety of methods [8, 7, 51] employ local features and are well adapted to the Bag-of-Words model [49]. Others are generic and applicable on any global image representation [27, 42, 2, 48, 10]. In both cases, ranking is performed on the image level. Extension to regional level is not always straightforward. If even possible, such an extension would come at a significant cost, as each query region would need to be treated independently. This is unlike our regional diffusion mechanism, which has a fixed cost with respect to the number of query regions.

Diffusion. We are focusing on diffusion mechanisms, which propagate similarities through a pairwise affinity matrix [13, 39]. They are applied to many computer vision problems, such as semi-supervised classification [63], seeded image segmentation [20], saliency detection [33, 6], clustering [12] and image retrieval [28, 14, 60, 13, 58].

The power of such methods lies in capturing the intrinsic manifold structure of the data [63]. The popular PageRank algorithm [39] was originally used to estimate the importance of web pages by exploiting their links in a graph structure. Our retrieval scenario comes closer to its so called personalized [39] or query dependent versions [45], where the final ranking both respects the data manifold and the similarity to a number of query vectors.

Diffusion is used for retrieval of general scenes or shapes of particular objects [28, 14, 60, 13]. It can also fuse multiple feature modalities [61, 59] by jointly modeling them on the same graph. In these approaches, images are the nodes of the graph with edges established given a pairwise similarity measure. We differentiate by defining a graph of image *regions* linked based on region similarities while performing a single pseudo random walk for multiple query regions. Diffusion with regional similarity has been investigated before, but only to define image level affinity [62], to aggregate local features [15], or to handle bursts [16].

Donoser and Bischof [13] review a number of diffusion mechanisms for retrieval. They focus on iterative solutions arguing that closed form solutions, when existing, are impractical due to inversion of large matrices. We rather focus on a closed form solution computed approximatively with an iterative method that is particularly designed for this problem and show that this approach is faster.

3. Ranking with diffusion

Diffusion in the work of Donoser and Bischof [13] denotes a mechanism spreading the query similarities over the manifolds composing the dataset. This is only weakly related to continuous time diffusion process or random walks on graph. We mainly follow Zhou et al. [64] below.

Affinity matrix. Given a dataset $\mathcal{X} := \{\mathbf{x}_1, \dots, \mathbf{x}_n\} \subset \mathbb{R}^d$, we define the *affinity matrix* $A = (a_{ij}) \in \mathbb{R}^{n \times n}$ having as elements the pairwise similarities between points of $\mathcal{X}$:

$$a_{ij} := s(\mathbf{x}_i, \mathbf{x}_j), \quad \forall (i,j) \in [n]^2, \tag{1}$$

where $[n] := \{1, \dots, n\}$ and $s : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a similarity measure assumed to be symmetric ($A = A^\top$), positive ($A > 0$), and with zero self-similarities ($\mathrm{diag}(A) = \mathbf{0}$).

Matrix A is the adjacency matrix of a weighted undirected graph G with vertices $\mathcal{X}$. The degree matrix of the graph is $D := \mathrm{diag}(A\mathbf{1}_n)$, *i.e.* a diagonal matrix with the row-wise sum of A. The Laplacian of the graph is defined as $L := D - A$. It is usual to symmetrically normalize these matrices, for instance,

$$S := D^{-1/2} A D^{-1/2}, \tag{2}$$

for the affinity matrix and $\mathcal{L} := I_n - S$ for the Laplacian, where I_n denotes the identity matrix of size n. Matrices $L, \mathcal{L}$ are positive-semidefinite [9].

Diffusion. In the work of Zhou et al. [64], a vector $\mathbf{y} = (y_i) \in \mathbb{R}^n$ specifies a set of query points in $\mathcal{X}$, with $y_i = 1$ if $\mathbf{x}_i$ is a query and $y_i = 0$ potherwise. The objective is to obtain a ranking score f_i for each point $\mathbf{x}_i \in \mathcal{X}$, represented as vector $\mathbf{f} = (f_i) \in \mathbb{R}^n$.

We focus on a particular diffusion mechanism that, given an initial vector $\mathbf{f}^0$, iterates according to

$$\mathbf{f}^t = \alpha S \mathbf{f}^{t-1} + (1-\alpha)\mathbf{y}. \tag{3}$$

If S is a transition matrix and $\mathbf{y}$ a ℓ^1 unit vector, this defines the following 'random walk' on the graph: with probability α one jumps to an adjacent vertex according to distribution stated in S, and with $1-\alpha$ to a query point uniformly at random. In this fashion, points spread their ranking score to their neighbors in the graph. The benefit is the ability to capture the intrinsic manifold structure represented by the affinity matrix and to combine multiple query points.

Assuming $0 < \alpha < 1$, Zhou et al. [63, 64] show that sequence $\{\mathbf{f}^t\}$ defined by (3) converges to

$$\mathbf{f}^\star = (1-\alpha)\mathcal{L}_\alpha^{-1}\mathbf{y} \tag{4}$$

where $\mathcal{L}_\alpha := I_n - \alpha S$ is positive-definite. This follows since $\mathcal{L}_\alpha = \alpha\mathcal{L} + (1-\alpha)I_n \succ \alpha\mathcal{L} \succeq 0$. In this work, we focus on the *closed form* solution (4) rather than its intuitive derivation from iterative process (3).

Relation to other approaches. A diffusion mechanism also appears in seeded image segmentation [20], where query points correspond to labeled pixels (seeds) and database points to the remaining unlabeled pixels. This problem is equivalent to semi-supervised classification [63]. In our context, the approach of Grady [20] decomposes $\mathbf{f} = (\mathbf{f}_d^\top, \mathbf{f}_q^\top)^\top$ for the scores of the query (fixed $\mathbf{f}_q$) and database (unknown $\mathbf{f}_d$) points. Diffusion interpolates $\mathbf{f}_d$ from $\mathbf{f}_q$ by minimizing, w.r.t. $\mathbf{f}_d$, the quadratic cost $\sum_{i,j} a_{ij}(f_i - f_j)^2 = \mathbf{f}^\top L \mathbf{f}$ to enforce that neighboring points should have similar scores. By decomposing $L = [L_d, -S_{qd}; -S_{qd}^\top, L_q]$, it is shown [20] that the solution fulfills $L_d \mathbf{f}_d = \mathbf{y}$ with $\mathbf{y} = S_{qd}^\top \mathbf{f}_q$. In our setup, L_d would be singular, preventing us to single out a solution $\mathbf{f}_d^\star$. Yet, it is easy to show that the minimizer of the cost $\alpha\mathbf{f}^\top L\mathbf{f} + (1-\alpha)\|\mathbf{f}\|^2$ has a similar expression to (4). The regularization term singles out a solution by forcing $\mathbf{f}$ to be zero in subgraphs not connected to any query point. The details are omitted for brevity.

Local constraints. Donoser and Bischof have extensively investigated various constructions of affinity matrices in the context of image retrieval [13]. Our work uses matrix (2), which is found to be the most effective in their work, and is also used by Zhou et al. [63]. Further, to handle noise and outliers, we adopt a locally constrained random walk [31] where only pairs of points that are reciprocal (mutual) nearest neighbors are kept as edges in the graph. In particular, given $\mathbf{z} \in \mathbb{R}^d$, let

$$s_k(\mathbf{x}|\mathbf{z}) = \begin{cases} s(\mathbf{x}, \mathbf{z}), & \text{if } \mathbf{x} \in \mathrm{NN}_k(\mathbf{z}) \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

be the similarity of $\mathbf{x} \in \mathcal{X}$ given $\mathbf{z}$, that is, restricted to the k nearest neighbors $\mathrm{NN}_k(\mathbf{z})$ of $\mathbf{z}$ in $\mathcal{X}$. Then,

$$s_k(\mathbf{x}, \mathbf{z}) = \min\{s_k(\mathbf{x}|\mathbf{z}), s_k(\mathbf{z}|\mathbf{x})\} \tag{6}$$

equals $s(\mathbf{x}, \mathbf{z})$ if $\mathbf{x}, \mathbf{z}$ are the k-nearest neighbors of each other in $\mathcal{X}$, and zero otherwise. We use similarity function s_k to construct affinity matrix A like in (1).

4. Method

This section describes our contributions on image retrieval: handling new query points not in the dataset, searching for multiple regions with a single diffusion mechanism, and efficiently computing the solution.

4.1. Handling new queries

In prior work on diffusion, a query point $\mathbf{q}$ is considered to be contained in the dataset $\mathcal{X}$ [63, 13]. This does not hold in a retrieval scenario, but a query can be included in the dataset graph at query time [61] as follows. The k nearest neighbors $\mathrm{NN}_k(\mathbf{q})$ of $\mathbf{q}$ in $\mathcal{X}$ are found and reciprocity is checked. The rows and columns of the affinity matrix A corresponding to $\mathrm{NN}_k(\mathbf{q})$ are updated to maintain (6) in the presence of $\mathbf{q}$, and A is augmented by appending an extra row and column for $\mathbf{q}$. Matrix S is computed by normalizing A (2). Finally, vector $\mathbf{y}$ indicates that $\mathbf{q}$ is a query. Generalizing to multiple query points is straightforward.

Even if we ignore the time needed for the above computation, we argue that locking, modifying and augmenting the entire affinity matrix for each query is not acceptable in terms of space requirements[1]. We introduce here an alternative method which defines vector $\mathbf{y}$ in a new way rather

[1] Imagine the case of multiple users querying at the same time; a different matrix per query is required. Also, updating mutual neighbors requires k-NN lists which are not available any longer.

than modifying A. Qualitatively, instead of searching for $\mathbf{q}$, we are searching for its neighbors $\mathrm{NN}_k(\mathbf{q})$, appropriately weighted. In particular, we define $\mathbf{y}$ as

$$y_i = s_k(\mathbf{x}_i|\mathbf{q}), \quad \forall i \in [n]. \tag{7}$$

Our motivation for this choice is detailed in Section 4.2 including the more general case of multiple query points. Figure 1 shows a toy 2-dimensional example of diffusion, where the k-nearest neighbors to each query point taken into account in (7) are depicted. It is evident that multiple manifolds are captured when multiple queries are issued. Section 5 experimentally shows improved performance compared to the conventional approach.

4.2. Regional diffusion

The diffusion mechanism described so far is applicable to image retrieval when database and query images are globally represented by single vectors. We call this *global diffusion* in the rest of the paper. Unlike the traditional representation with local descriptors [49, 40], global diffusion fits perfectly with the early CNN-based global features [4, 30, 43].

Global features still fail under severe occlusion or when the object of interest is small. Local CNN features from multiple image regions have been investigated for this purpose, either aggregated [17, 52] or represented as a set [44]. Given a query image, the latter means that one searches for each query feature individually.

Fortunately, diffusion as defined in section 3 can already handle multiple queries. In the following, an image is represented by a set $X_i \subset \mathbb{R}^d$ of m points, one for each region. Dataset $\mathcal{X}$ is the union of such sets over all images; n still denotes its size. The query image is also represented by a set Q of m points. Each region feature is a point possibly lying on a different manifold. We discuss below the new definition of vector $\mathbf{y}$ and the combination of individual region ranking scores into a single score per image. We call this mechanism *regional diffusion*.

Specifying queries. In the conventional approach where query points are in the dataset, one directly applies (3) with $\mathbf{y} \in \{0,1\}^{n+m}$ with m non-zero elements indicating the query points. This situation resembles the personalized PageRank [39]. However, it is simpler to keep A as an $n \times n$ affinity matrix and to set $\mathbf{y} \in \mathbb{R}^n$ as

$$y_i := \sum_{\mathbf{q} \in Q} s_k(\mathbf{x}_i|\mathbf{q}), \quad \forall i \in [n]. \tag{8}$$

Each dataset point $\mathbf{x}_i$ is assigned a scalar that is the sum of similarities over all query points $\mathbf{q}$ for which $\mathbf{x}_i$ appears in the corresponding k-nearest neighbor set $\mathrm{NN}_k(\mathbf{q})$, and zero if it appears in no such set.

Derivation. Our work is inspired by the analysis in the work of Grady [20] that we apply to the diffusion mechanism of Zhou et al. [64], where query points Q are in the dataset. We decompose the quantities in (3) as $\mathbf{f} = (\mathbf{f}_d^\top, \mathbf{f}_q^\top)^\top$, with $\mathbf{f}_d \in \mathbb{R}^n$ and $\mathbf{f}_q \in \mathbb{R}^m$,

$$S = \begin{pmatrix} S_d & B_{dq} \\ B_{qd} & S_q \end{pmatrix}, \tag{9}$$

and $\mathbf{y} = (\mathbf{0}_n^\top, \mathbf{1}_m^\top)^\top$. Subscripts d, q denote data and query respectively. Then, (3) is written as

$$\mathbf{f}_d^t = \alpha S_d \mathbf{f}_d^{t-1} + \alpha B_{dq} \mathbf{f}_q^{t-1} \tag{10}$$

$$\mathbf{f}_q^t = \alpha B_{qd} \mathbf{f}_d^{t-1} + \alpha S_q \mathbf{f}_q^{t-1} + (1-\alpha)\mathbf{1}_m. \tag{11}$$

Provided this system converges, the data part satisfies

$$\mathbf{f}_d^\star \propto \mathcal{L}_\alpha^{-1} B_{dq} \mathbf{1}_m \tag{12}$$

if $\mathbf{f}_q^\star \propto \mathbf{1}_m$, $S_q = \mathbf{0}_{m \times m}$ and $B_{qd} = \mathbf{0}_{m \times n}$. In words, the query points are perfectly retrieved, they are dissimilar to each other, and the graph is indeed directed with query regions pointing to dataset regions, but the reverse is not allowed. Comparing (12) with (4), it follows that $B_{dq}\mathbf{1}_m$ is a good choice for $\mathbf{y}$. Since B_{dq} stores the similarities between the dataset and the query points, this analysis justifies the single query (7) and the multiple queries (8) cases.

Diffusion. Given this definition of $\mathbf{y}$, diffusion is now performed on dataset $\mathcal{X}$, jointly for all query points in Q. Affinities of multiple query points are propagated in the graph in a single process at no additional cost compared to the case of a single query point. Here we are excluding the additional cost of computing $\mathbf{y}$ itself in (8) compared to (7). This search takes place in all related work. We also do not discuss how to make this search more efficient in space and time [5], which is beyond the scope of this work.

Figure 1 illustrates the diffusion on single and multiple query points. The contour lines show the ranking score any point on the plane would be assigned given the query point(s). It is evident that multiple manifolds are captured when multiple queries are issued.

Pooling. After diffusion, each image is associated with several elements of the ranking score vector $\mathbf{f}^\star$, one for each point $\mathbf{x}$ in $X \subset \mathcal{X}$. A simple way to combine these scores is to define the score of image X as

$$f(X) = \sum_{j \in [m]} w_j f^\star_{i_X(j)}, \tag{13}$$

where $i_X(j)$ is the index of the j-th point of X in the dataset $\mathcal{X}$ and $\mathbf{w} = (w_j)$ a weighting vector. The latter is defined as $\mathbf{w} = \mathbf{1}_m$ for *sum pooling* and, assuming $m < d$,

$$\mathbf{w} = (\Phi\Phi^\top + \lambda I_m)^{-1} \mathbf{1}_m \tag{14}$$

for *generalized max pooling* (GMP) [37, 23], where $\Phi = (\mathbf{x}_{i_X(1)}^\top, \ldots, \mathbf{x}_{i_X(m)}^\top)^\top$ and $\lambda \in \mathbb{R}^+$ is a regularization parameter. Our experiments show that GMP always outperforms sum pooling.

4.3. Efficient solution

Iteration (3) works well in practice but is slow at large scale. Taking the closed-form solution (4) literally, one may be tempted to compute the inverse $\mathcal{L}_\alpha^{-1}$ offline, but this matrix is not sparse like $\mathcal{L}_\alpha$. We propose a more efficient solution by making the connection to linear system solvers.

Diffusion is an iterative solver. Eq. (3) can be seen as an iteration of the *Jacobi* solver [21]. Given a linear system $A\mathbf{x} = \mathbf{b}$[2], Jacobi decomposes A as $A = \Delta + R$ where $\Delta = \mathrm{diag}(A)$. It then iterates according to

$$\mathbf{x}^t = \Delta^{-1}(\mathbf{b} - R\mathbf{x}^{t-1}). \tag{15}$$

In our case, $\mathbf{x} = \mathbf{f}$, $\mathbf{b} = (1-\alpha)\mathbf{y}$, and $A = \mathcal{L}_\alpha = I - \alpha S$. It follows that $\Delta = I_n$ and $R = -\alpha S$, so that

$$\mathbf{f}^t = \alpha S\mathbf{f}^{t-1} + (1-\alpha)\mathbf{y}. \tag{16}$$

We have just re-derived (3). Note that a sufficient condition for Jacobi's convergence is that matrix A is strictly diagonally dominant, *i.e.* $|a_{ii}| > \sum_{j\neq i} a_{ij}$ for $i \in [n]$. It is easily checked that $\mathcal{L}_\alpha$ does satisfy this condition by construction, given that $0 < \alpha < 1$. This provides an alternative proof of the main result of Zhou et al. [63].

Conjugate gradient (CG) [38] is the method of choice for solving linear systems like ours

$$\mathcal{L}_\alpha \mathbf{f} = (1-\alpha)\mathbf{y}, \tag{17}$$

where $\mathcal{L}_\alpha$ is positive-definite, and in particular for graph-related problems [54]. It has been used for random walk problems [20], but not diffusion-based retrieval according to our knowledge. In fact, the linear system formulation has been explicitly avoided in this context [13].

Here we argue, as in [32], that it is the solution of (17) that we seek, rather than the path followed by iteration (3). However, we use CG to approximate this solution, since matrix $\mathcal{L}_\alpha$ is indeed positive-definite. At each iteration, CG minimizes the quadratic function $\phi(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\top A\mathbf{x} - \mathbf{x}^\top\mathbf{b}$ in a particular direction by analytically computing the optimal step length. More importantly, the direction chosen at each iteration is conjugate to previous ones. Thus, any update of $\mathbf{x}$ along this direction does not destroy the optimality reached in the entire subspace considered thus far.

Contrary to other iterative methods including (16), CG is guaranteed to terminate in n steps. Remarkably, it provides good approximations in very few steps.

Normalization is preconditioning. Finally, a standard improvement is preconditioning, *i.e.* , solving a related system with A replaced by $C^{-1}AC^{-\top}$, a matrix satisfying a weak condition like its eigenvalues being clustered. Unfortunately, finding an appropriate matrix C can be quite complex [54]. We observe that normalization (2) *is* preconditioning.Indeed, we could equally consider matrix $L_\alpha = D - \alpha A = \alpha L + (1-\alpha)I \succ 0$ and solve the linear system

$$L_\alpha(D^{-1/2}\mathbf{f}) = (1-\alpha)(D^{1/2}\mathbf{y}) \tag{18}$$

instead, which is equivalent to (17). By normalizing L_α into $\mathcal{L}_\alpha$, we are actually performing preconditioning with $C = \mathrm{diag}(L_\alpha)^{1/2}$. This is a simple form of symmetric preconditioning, known as *diagonal scaling* or *Jacobi* [53]. It improves convergence, be it for CG or diffusion (3).

4.4. Scaling up

Despite the efficient solution described in the previous section, there are still issues concerning space and offline pre-processing at large scale. We address these issues here.

Compact representation. At large scale, the number of region features per database image should be kept as low as possible. For this reason, we learn a Gaussian Mixture Model (GMM) on the original features of each database image and represent the image by the unit normalized means. This is an even more natural choice when dealing with overlapping regions (see Section 5). As a result, it decreases the number of region features and their redundancy.

The off-line construction of the affinity matrix is quadratic in the number of vectors in the database and might not be tractable at large scale. We employ the efficient and approximate k-NN graph construction method by Dong et al. [11]. Section 5 shows that it is orders of magnitude faster than exhaustive search and has almost no effect on performance.

Truncating the affinity matrix. Instead of ranking the full dataset, diffusion re-ranks an initial search. This baseline in our experiments is done with global descriptors and kNN search. Then we apply diffusion only on the top ranked images. We truncate the affinity matrix keeping only the rows and columns related to the regions of the top ranked images and re-normalize it according to (2). The cost of this step is not significant compared to the actual diffusion.

5. Experiments

This section presents the experimental setup and investigates the accuracy of our methods for image retrieval compared with the state-of-the-art approaches.

5.1. Experimental Setup

Datasets. We use three datasets. Two are well-known image retrieval benchmarks: Oxford Buildings [40] and Paris [41]. We refer to them as Oxford5k and Paris6k. We experiment at large-scale by adding 100k distractor images from Flickr [40], forming Oxford105k and Paris106k datasets. The third corpus is the recently introduced instance search dataset called INSTRE [55]. It contains various everyday 3D or planar objects from buildings to logos

[2] We adopt the standard linear system notation in this section; matrix A is not to be confused with our affinity matrix defined in (1).

Pooling	INSTRE	Oxf5k	Oxf105k	Par6k	Par106k
sum	79.1	92.2	90.6	96.1	94.4
GMP	80.0	93.2	91.6	96.5	94.6

Table 1. Retrieval performance (mAP) of regional diffusion with sum and generalized max pooling (GMP), with $\lambda = 1$ in (14).

with many variations such as different scales, rotations, and occlusions. Some objects cover a small part of the image, making it a challenging dataset. It consists of 28,543 images from 250 different object classes. In particular, 100 classes with images retrieved from on-line sources, 100 classes with images taken by the dataset creators, and 50 classes consisting of pairs from the second category. We differentiate from the original protocol [55], which uses all database images as queries. We randomly split the dataset into 1250 queries, 5 per class, and 27293 database images, while a bounding box defines the query region[3]. The query and the database sets have no overlap. We use mean average precision (mAP) as a performance measure in all datasets.

Representation. We employ a CNN that is fine-tuned for image retrieval [43] to extract global and regional representation. In particular, this fine-tuned VGG produces 512 dimensional descriptors. We extract regions at 3 different scales as in R-MAC [52], while we additionally include the full image as a region. In this fashion, each image has on average 21 regions. The regional descriptors are aggregated and re-normalized to unit norm in order to construct the global descriptors, which is exactly as in R-MAC. We apply supervised whitening [43] to both global and regional descriptors. We use this network to perform all our initial experiments. In Section 5.4, we also report scores with higher dimensional descriptors derived from the fine-tuned ResNet101 [19] using the same fixed grid.

Implementation details. We define the affinity function using a monomial kernel [50] as $s(\mathbf{x}, \mathbf{z}) = \max(\mathbf{x}^\top \mathbf{z}, 0)^3$. The diffusion parameter α is always 0.99, as in the work of Zhou et al. [63]. The k-NN search required by (8) is assumed to access all database vectors exhaustively. Our work does not investigate how approximate search methods [36, 25, 29, 5, 24] could improve time and space consumed by this process. After computing (8), we only keep the largest k values of $\mathbf{y}$ and set the rest to zero.

5.2. Impact of different components

Neighbors. We vary the number of nearest neighbors k for constructing the affinity matrix and evaluate performance for both global and regional diffusion. The global baseline method is k-NN search with R-MAC, while the regional one is the method by Razavian et al. [44], where image regions

[3] http://people.rennes.inria.fr/Ahmet.Iscen/diffusion.html

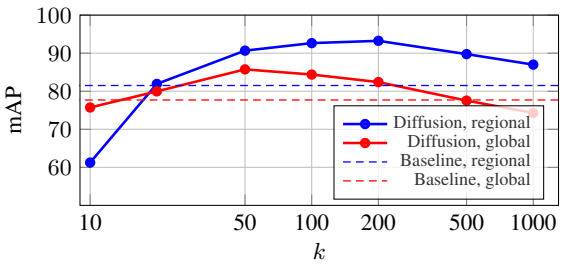


Figure 2. Impact of the number of nearest neighbors k in the affinity matrix. mAP performance for global and regional diffusion on Oxford5k; baselines are R-MAC and R-match respectively.

are indexed and cross-matched. We refer to the latter as R-match in the rest of our experiments.

Results for Oxford5k are presented in Figure 2, and are consistent in other datasets. The performance stays stable over a wide range of k. The drop for low k is due to very few neighbors being retrieved (where regional diffusion is more sensitive), whereas for high k, it is due to capturing more than the local manifold structure (where regional diffusion is superior). This behavior is consistent with the fact that small patterns appear more frequently than entire images.

We set $k = 200$ for regional diffusion, and $k = 50$ for global diffusion for the rest of our paper. Since only mutual neighbors are linked, the actual number of edges per element is less: The average number of edges per image (resp. region) is 25 (resp. 75) for global (resp. regional) diffusion, measured on INSTRE. We set $k = 200$ for the query as well in the case of the regional diffusion, while for the global one $k = 10$ is needed to achieve good performance.

Pooling. We evaluate the two pooling strategies after regional diffusion in Table 1. Generalized max pooling has a small but consistent benefit in all datasets. We use this strategy for the rest of our experiments. Weights (14) are computed off-line and only one scalar per region is stored.

Efficient diffusion with conjugate gradient. We compare the iterative diffusion (3) to our conjugate gradient solution. We iterate each method until convergence. Performance is presented in Figure 3 with timings measured on a machine with a 4-core Intel Xeon 2.00GHz CPU. CG converges in as few as 20 iterations, which are also faster, while (3) reaches the same performance as CG only after 110 iterations.

The average query time on Oxford5k including all stages for global baseline, regional baseline, global diffusion and regional diffusion without truncation is 0.001s, 0.321s, 0.02s, and 0.664s, respectively.

Handling new queries. We compare our new way of handling new queries to the conventional approach that assumes queries to be part of the dataset. Our method achieves 80.0 mAP on INSTRE compared to 77.7 achieved by the conventional approach. We therefore not only offer space

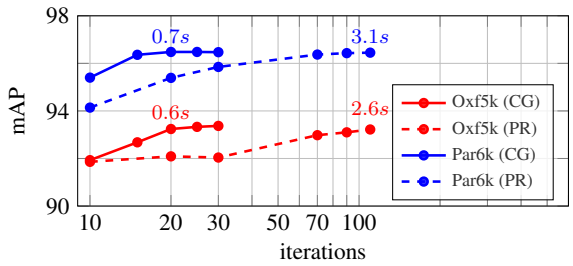


Figure 3. mAP performance of regional diffusion *vs.* number of iterations for conjugate gradient (CG) and iterative diffusion (3). Labels denote diffusion time.

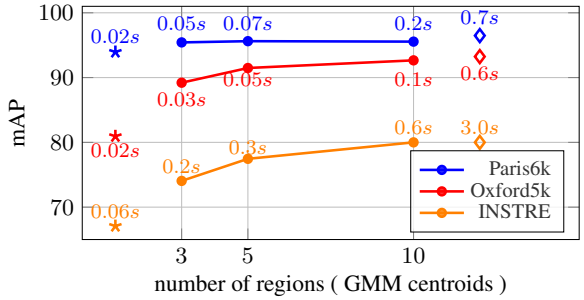


Figure 4. mAP performance for varying number of regional descriptors after learning a GMM per image. Symbol $\star$ denotes global diffusion, and $\diamond$ to the default number of regions (21) per image. Average diffusion time in seconds is shown in text labels.

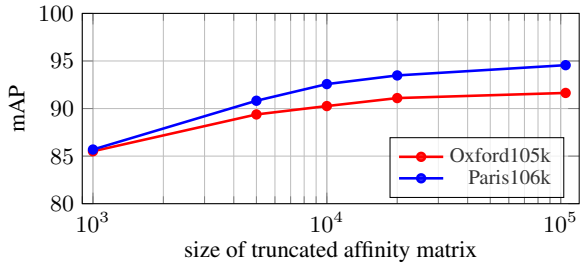


Figure 5. Retrieval performance (mAP) versus the shortlist size used for affinity matrix truncation.

improvements but also better performance,mainly in the case of regional diffusion. The main difference is that k nonzero elements are kept both per query region (8) and for the entire vector $\mathbf{y}$. This, due to the overlapping nature of the CNN regions, may filter out incorrect neighbors.

5.3. Large scale diffusion

We now focus on the large scale solutions of Section 4.4.

Reduced number of regions. Figure 4 shows the impact of reducing the number of regions with Gaussian mixture models. Having as few as 5 descriptors per image already achieves competitive performance, while reducing the online search complexity. We decrease the number of neighbors k to 50 when GMM reduction is used, as there are now less positive neighbors.

Method	$m \times d$	INSTRE	Oxf5k	Oxf105k	Par6k	Par106k
Global descriptors - nearest neighbor search						
CroW [30]†	512	-	68.2	63.2	79.8	71.0
R-MAC [43]	512	47.7	77.7	70.1	84.1	76.8
R-MAC [19]	2,048	62.6	83.9	80.8	93.8	89.9
NetVLAD [1]†	4,096	-	71.6	-	79.7	-
Global descriptors - query expansion						
R-MAC [43]+AQE [8]	512	57.3	85.4	79.7	88.4	83.5
R-MAC [43]+SCSM [48]	512	60.1	85.3	80.5	89.4	84.5
R-MAC [43]+HN [42]	512	64.7	79.9	-	92.0	-
Global diffusion	512	70.3	85.7	82.7	94.1	92.5
R-MAC [19]+AQE [8]	2,048	70.5	89.6	88.3	95.3	92.7
R-MAC [19]+SCSM [48]	2,048	71.4	89.1	87.3	95.4	92.5
Global diffusion	2,048	80.5	87.1	87.4	96.5	95.4
Regional descriptors - nearest neighbor search						
R-match [44]	21×512	55.5	81.5	76.5	86.1	79.9
R-match [44]	21×2,048	71.0	88.1	85.7	94.9	91.3
Regional descriptors - query expansion						
HQE [51]	2.4k×128	74.7	89.4†	84.0†	82.8†	-
R-match [44]+AQE [8]	21×512	60.4	83.6	78.6	87.0	81.0
Regional diffusion*	5×512	77.5	91.5	84.7	95.6	93.0
Regional diffusion*	21×512	80.0	93.2	90.3	96.5	92.6
R-match [44]+AQE [8]	21×2,048	77.1	91.0	89.6	95.5	92.5
Regional diffusion*	5×2,048	88.4	95.0	90.0	96.4	**95.8**
Regional diffusion*	21×2,048	**89.6**	**95.8**	**94.2**	**96.9**	95.3

Table 2. Performance comparison to the state of the art. Results from original publications are marked with †, otherwise they are based on our implementation. Our methods are marked with *. Points at 512D are extracted with VGG [43] and at 2048D with ResNet101 [19]. Regional diffusion with 5 regions uses GMM.

Affinity matrix with Dong's algorithm [11]. We compare the exhaustive construction of matrix A to Dong's efficient k-NN graph algorithm [11]. Exhaustive search for Oxford105k composed of 2.2M regions takes 96 hours on a machine with a 12-core Intel Xeon 2.30GHz CPU. The approximate graph only takes 45 minutes and affects the final retrieval performance only slightly. It achieves 91.6 mAP on Oxford105k and 94.6 on Paris106k, while the exhaustive construction yields 92.5 and 95.2 respectively.

Truncation is a means to handle large scale datasets, *i.e.* more than 100k images. Regional diffusion on the full dataset takes 13.9s for Oxford105k, which is not practical. We therefore rank images according to the aggregated regional descriptors, which is equivalent to the R-MAC representation [52], and then perform diffusion on a short-list.

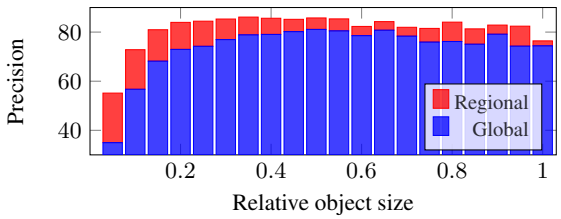


Figure 6. Precision of each positive image measured at the position where it was retrieved, averaged over positive images according relative object size. Statistics computed on INSTRE over all queries for global and regional diffusion.

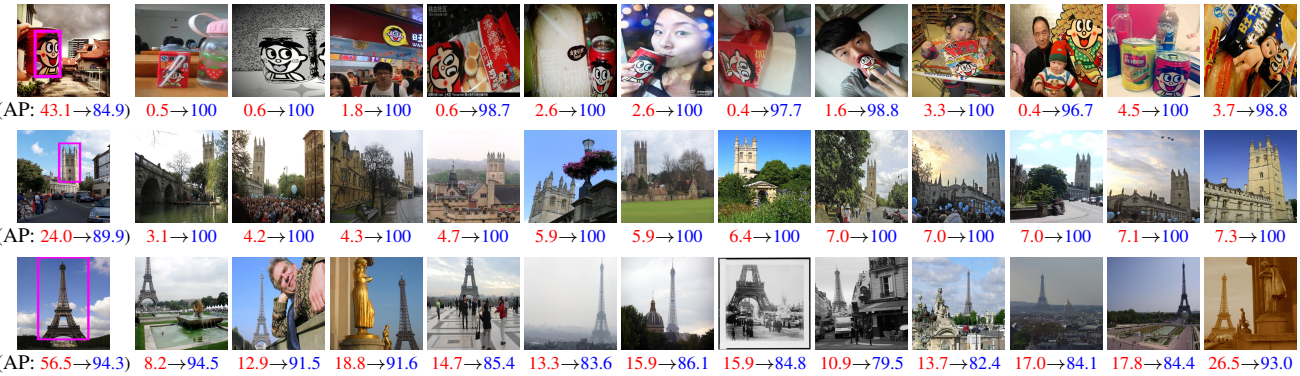


Figure 7. Query examples from INSTRE, Oxford, and Paris datasets and retrieved images ranked by decreasing order of ranking difference between global and regional diffusion. We measure precision at the position where each image is retrieved and report this under each image for global and regional diffusion. Average Precision (AP) is reported per query for the two methods.

Figure 5 reports results with truncation. The performance of the full database diffusion is nearly attained by re-ranking less than 10% of the database. The entire truncation and diffusion process on Oxford105k takes 1s, with truncation and re-normalization taking only a small part of it. In the following, search on Oxford105k and Paris105k is performed by truncating the top 10k images. This choice results in an affinity matrix A of around 200k regions. When GMM reduction is used, our short-list size is chosen so that A has 2M regions too, keeping re-ranking complexity fixed.

Our approach is scalable thanks to truncation: the short-list length is fixed and so is the re-ranking time, regardless of the database size and the dimensionality of the descriptors. Although this shortlist contains a small fraction of the database, its significantly outperforms the baseline.

Small objects. We present quantitative and qualitative results revealing that images benefit from our method mainly when the depicted object is small and the scene is cluttered. Figure 7 shows that the retrieved images with the highest increase of precision of regional compared to global diffusion contain small objects that the latter cannot see. Since the bounding boxes are available for all images of INSTRE, we quantitatively measure precision for all positive images: Figure 6 shows that the highest improvement indeed comes for objects with small relative size.

5.4. Comparison to other methods

We compare with the state-of-the-art approaches with global or regional representation, with or without query expansion. Table 2 summarizes the results. We implement three methods typically combined with BoW, namely Average Query Expansion (AQE) [8], Spatially Constrained Similarity Measure (SCSM) [48] and Hello Neighbor (HN) [42]. AQE is also effective with CNN global representation [52, 30, 18]. A baseline for the regional scenario is R-match [44]. We additionally extend AQE to regional representation[4] combined with the similarity used in R-match. Hamming Query Expansion[5] (HQE) [51] is the only method not using CNNs, but local descriptors.

Regional diffusion significantly outperforms all other methods in all datasets. Global diffusion performs well on Paris because query objects almost fully cover the image in most of the database entries. This does not hold on INSTRE, which contains a lot of small objects. The improvements of regional diffusion are in this case much larger.

6. Conclusion

We propose a retrieval approach capturing distinct manifolds in the description space at no additional cost compared to a single query. We experimentally show that it significantly improves retrieval of small objects and cluttered scenes. The conclusion is that as few as 5-10 regional CNN descriptors can convey important information on small objects while thousands of conventional local descriptors are typically needed. Thus, a regional affinity matrix becomes possible. Regional diffusion was not possible before. In contrast to prior work, we use the closed form solution of the diffusion iteration, obtained by the conjugate gradient method. Combined with our contributions on space efficiency, this achieves large scale search at reasonable query times. Using recent CNN architectures, we achieve state-of-the-art and near optimal performance on two popular benchmarks and a recent more challenging dataset.

Acknowledgments The authors were supported by the MSMT LL1303 ERC-CZ grant. The Tesla K40 used for this research was donated by the NVIDIA Corporation.

[4] AQE has not been proposed in a regional scenario. We extend it as competitive baseline derived from prior work.

[5] We evaluated HQE on INSTRE for the purposes of this work.

References

[1] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. In *CVPR*, 2016. 1, 7

[2] R. Arandjelovic and A. Zisserman. Three things everyone should know to improve object retrieval. In *CVPR*, June 2012. 2

[3] H. Azizpour, A. S. Razavian, J. Sullivan, A. Maki, and S. Carlsson. From generic to specific deep representations for visual recognition. In *CVPRW*, 2014. 1

[4] A. Babenko and V. Lempitsky. Aggregating deep convolutional features for image retrieval. In *ICCV*, 2015. 1, 4

[5] A. Babenko and V. Lempitsky. Efficient indexing of billion-scale datasets of deep descriptors. In *CVPR*, 2016. 1, 4, 6

[6] S. Chen, L. Zheng, X. Hu, and P. Zhou. Discriminative saliency propagation with sink points. *Pattern recognition*, 60:2–12, 2016. 2

[7] O. Chum, A. Mikulik, M. Perdoch, and J. Matas. Total recall II: Query expansion revisited. In *CVPR*, June 2011. 2

[8] O. Chum, J. Philbin, J. Sivic, M. Isard, and A. Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *ICCV*, October 2007. 1, 2, 7, 8

[9] F. R. Chung. *Spectral graph theory*, volume 92. American Mathematical Soc., 1997. 3

[10] A. Delvinioti, H. Jégou, L. Amsaleg, and M. Houle. Image retrieval with reciprocal and shared nearest neighbors. In *VISAPP*, 2014. 2

[11] W. Dong, M. Charikar, and K. Li. Efficient k-nearest neighbor graph construction for generic similarity measures. In *WWW*, March 2011. 5, 7

[12] M. Donoser. Replicator graph clustering. In *BMVC*, 2013. 2

[13] M. Donoser and H. Bischof. Diffusion processes for retrieval revisited. In *CVPR*, 2013. 2, 3, 5

[14] A. Egozi, Y. Keller, and H. Guterman. Improving shape retrieval by spectral matching and meta similarity. IEEE *Transactions on Image Processing*, 19(5):1319–1327, 2010. 2

[15] T. Furuya and R. Ohbuchi. Diffusion-on-manifold aggregation of local features for shape-based 3d model retrieval. In *ICMR*, 2015. 2

[16] Z. Gao, J. Xue, W. Zhou, S. Pang, and Q. Tian. Democratic diffusion aggregation for image retrieval. *IEEE Trans. on Multimedia*, 18:1661 – 1674, 2016. 2

[17] Y. Gong, L. Wang, R. Guo, and S. Lazebnik. Multi-scale orderless pooling of deep convolutional activation features. In *ECCV*, 2014. 4

[18] A. Gordo, J. Almazan, J. Revaud, and D. Larlus. Deep image retrieval: Learning global representations for image search. *ECCV*, 2016. 1, 8

[19] A. Gordo, J. Almazan, J. Revaud, and D. Larlus. End-to-end learning of deep visual representations for image retrieval. In *arXiv*, 2016. 1, 6, 7

[20] L. Grady. Random walks for image segmentation. *IEEE Trans. PAMI*, 28(11):1768–1783, 2006. 2, 3, 4, 5

[21] W. Hackbusch. *Iterative solution of large sparse systems of equations*. Springer Verlag, 1994. 5

[22] J. Heinly, J. L. Schonberger, E. Dunn, and J.-M. Frahm. Reconstructing the world* in six days*(as captured by the yahoo 100 million image dataset). In *CVPR*, 2015. 1

[23] A. Iscen, T. Furon, V. Gripon, M. Rabbat, and H. Jégou. Memory vectors for similarity search in high-dimensional spaces. In *arXiv*, 2014. 4

[24] A. Iscen, M. Rabbat, and T. Furon. Efficient large-scale similarity search using matrix factorization. In *CVPR*, 2016. 6

[25] H. Jégou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE Trans. PAMI*, 33(1):117–128, January 2011. 1, 6

[26] H. Jégou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *CVPR*, June 2010. 1

[27] H. Jégou, H. Harzallah, and C. Schmid. A contextual dissimilarity measure for accurate and efficient image search. In *CVPR*, 2007. 2

[28] Y. Jing and S. Baluja. Visualrank: Applying pagerank to large-scale image search. *IEEE Trans. PAMI*, 30(11):1877–1890, 2008. 2

[29] Y. Kalantidis and Y. Avrithis. Locally optimized product quantization for approximate nearest neighbor search. In *CVPR*, 2014. 1, 6

[30] Y. Kalantidis, C. Mellina, and S. Osindero. Cross-dimensional weighting for aggregated deep convolutional features. In *ECCVW*, 2016. 1, 4, 7, 8

[31] P. Kontschieder, M. Donoser, and H. Bischof. Beyond pairwise shape similarity analysis. In *ACCV*, 2009. 3

[32] A. N. Langville and C. D. Meyer. Deeper inside pagerank. *Internet Mathematics*, 1(3):335–380, 2004. 5

[33] S. Lu, V. Mahadevan, and N. Vasconcelos. Learning optimal seeds for diffusion-based salient object detection. In *CVPR*, 2014. 2

[34] A. Mikulik, O. Chum, and J. Matas. Image retrieval for online browsing in large image collections. In *International Conference on Similarity Search and Applications*, 2013. 1

[35] K. R. Mopuri and R. V. Babu. Object level deep feature pooling for compact image representation. *CVPRW*, 2015. 1

[36] M. Muja and D. G. Lowe. Scalable nearest neighbor algorithms for high dimensional data. *IEEE Trans. PAMI*, 36, 2014. 1, 6

[37] N. Murray and F. Perronnin. Generalized max-pooling. In *CVPR*, June 2014. 4

[38] J. Nocedal and S. Wright. *Numerical optimization*. Springer, 2006. 5

[39] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: bringing order to the web. 1999. 2, 4

[40] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, June 2007. 4, 5

[41] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *CVPR*, June 2008. 5

[42] D. Qin, S. Gammeter, L. Bossard, T. Quack, and L. Van Gool. Hello neighbor: Accurate object retrieval with k-reciprocal nearest neighbors. In *CVPR*, 2011. 2, 7, 8

[43] F. Radenović, G. Tolias, and O. Chum. CNN image retrieval learns from BoW: Unsupervised fine-tuning with hard examples. *ECCV*, 2016. 4, 6, 7

[44] A. S. Razavian, J. Sullivan, S. Carlsson, and A. Maki. Visual instance retrieval with deep convolutional networks. *ITE Transactions on Media Technology and Applications*, 4:251–258, 2016. 1, 4, 6, 7, 8

[45] M. Richardson and P. M. Domingos. The intelligent surfer: Probabilistic combination of link and content information in pagerank. In *NIPS*, 2001. 2

[46] T. Sattler, T. Weyand, B. Leibe, and L. Kobbelt. Image retrieval for image-based localization revisited. In *BMVC*, 2012. 1

[47] J. L. Schonberger, F. Radenovic, O. Chum, and J.-M. Frahm. From single image query to detailed 3d reconstruction. In *CVPR*, 2015. 1

[48] X. Shen, Z. Lin, J. Brandt, and Y. Wu. Spatially-constrained similarity measure for large-scale object retrieval. *IEEE Trans. PAMI*, 36(6):1229–1241, 2014. 2, 7, 8

[49] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *ICCV*, 2003. 2, 4

[50] G. Tolias, Y. Avrithis, and H. Jégou. To aggregate or not to aggregate: Selective match kernels for image search. In *ICCV*, December 2013. 6

[51] G. Tolias and H. Jégou. Visual query expansion with or without geometry: refining local descriptors by feature aggregation. *Pattern recognition*, 47(10):3466–3476, 2014. 2, 7, 8

[52] G. Tolias, R. Sicre, and H. Jégou. Particular object retrieval with integral max-pooling of cnn activations. *ICLR*, 2016. 1, 4, 6, 7, 8

[53] L. N. Trefethen and D. Bau III. *Numerical linear algebra*. SIAM, 1997. 5

[54] N. K. Vishnoi. Laplacian solvers and their algorithmic applications. *Theoretical Computer Science*, 8(1-2):1–141, 2012. 5

[55] S. Wang and S. Jiang. INSTRE: a new benchmark for instance-level object retrieval and recognition. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 11:37, 2015. 2, 5, 6

[56] T. Weyand and B. Leibe. Discovering favorite views of popular places with iconoid shift. In *ICCV*, 2011. 1

[57] L. Xie, R. Hong, B. Zhang, and Q. Tian. Image classification and retrieval are one. In *ICMR*, 2015. 1

[58] L. Xie, Q. Tian, W. Zhou, and B. Zhang. Fast and accurate near-duplicate image search with affinity propagation on the imageweb. *CVIU*, 124, 2014. 2

[59] F. Yang, B. Matei, and L. S. Davis. Re-ranking by multi-feature fusion with diffusion for image retrieval. In *WACV*, 2015. 2

[60] X. Yang, S. Koknar-Tezel, and L. J. Latecki. Locally constrained diffusion process on locally densified distance spaces with applications to shape retrieval. In *CVPR*, 2009. 2

[61] S. Zhang, M. Yang, T. Cour, K. Yu, and D. N. Metaxas. Query specific fusion for image retrieval. In *ECCV*, 2012. 2, 3

[62] W. Zhang, C.-W. Ngo, and X. Cao. Hyperlink-aware object retrieval. IEEE *Transactions on Image Processing*, 25(9):4186–4198, 2016. 2

[63] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *NIPS*, 2003. 2, 3, 5, 6

[64] D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Schölkopf. Ranking on data manifolds. In *NIPS*, 2003. 2, 3, 4

XVIII Fast Spectral Ranking for Similarity Search

Title: Fast Spectral Ranking for Similarity Search
Authors: A. Iscen, Y. Avrithis, G. Tolias, T. Furon, O. Chum
Published at: CVPR 2018

Fast Spectral Ranking for Similarity Search

Ahmet Iscen[1] Yannis Avrithis[2] Giorgos Tolias[1] Teddy Furon[2] Ondřej Chum[1]
[1]VRG, FEE, CTU in Prague [2]Inria Rennes
{ahmet.iscen,giorgos.tolias,chum}@cmp.felk.cvut.cz
{ioannis.avrithis,teddy.furon}@inria.fr

Abstract

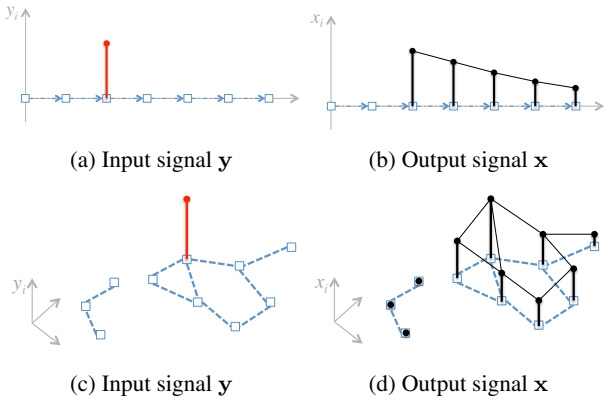
Despite the success of deep learning on representing images for particular object retrieval, recent studies show that the learned representations still lie on manifolds in a high dimensional space. This makes the Euclidean nearest neighbor search biased for this task. Exploring the manifolds online remains expensive even if a nearest neighbor graph has been computed offline.

This work introduces an explicit embedding reducing manifold search to Euclidean search followed by dot product similarity search. This is equivalent to linear graph filtering of a sparse signal in the frequency domain. To speed up online search, we compute an approximate Fourier basis of the graph offline. We improve the state of art on particular object retrieval datasets including the challenging Instre dataset containing small objects. At a scale of 10^5 images, the offline cost is only a few hours, while query time is comparable to standard similarity search.

1. Introduction

Image retrieval based on deep learned features has recently achieved near perfect performance on all standard datasets [45, 14, 15]. It requires fine-tuning on a properly designed image matching task involving little or no human supervision. Yet, retrieving particular *small* objects is a common failure case. Representing an image with several regions rather than a global descriptor is indispensable in this respect [46, 60]. A recent study [24] uses a particularly challenging dataset [67] to investigate graph-based query expansion and re-ranking on regional search.

Query expansion [7] explores the image manifold by recursive Euclidean or similarity search on the nearest neighbors (NN) at increased online cost. *Graph-based* methods [44, 53] help reducing this cost by computing a k-NN graph offline. Given this graph, *random walk*[1] processes [39, 70] provide a principled means of ranking. Iscen *et al*. [24] transform the problem into finding a solution $\mathbf{x}$ of a linear system $A\mathbf{x} = \mathbf{y}$ for a large sparse dataset-dependent matrix A and a sparse query-dependent vector $\mathbf{y}$. Such a solution can be found efficiently on-the-fly with *conjugate gradients* (CG). Even for an efficient solver, the query times are still in the order of one second at large scale.



(a) Input signal $\mathbf{y}$ (b) Output signal $\mathbf{x}$

(c) Input signal $\mathbf{y}$ (d) Output signal $\mathbf{x}$

Figure 1: The low-pass filtering of an impulse over the real line (top) and a graph (bottom). In a weighted undirected graph the information "flows" in all directions, controlled by edge weights. In retrieval, the impulse in red is the query, and the output $\mathbf{x}$ is its similarity to all samples.

In this work, we shift more computation offline: we exploit a low-rank spectral decomposition $A \approx U\Lambda U^\top$ and express the solution in closed form as $\mathbf{x} = U\Lambda^{-1}U^\top\mathbf{y}$. We thus treat the query as a signal $\mathbf{y}$ to be smoothed over the graph, connecting query expansion to *graph signal processing* [50]. Figure 1 depicts 1d and graph miniatures of this interpretation. We then generalize, improve and interpret this *spectral ranking* idea on large-scale image retrieval. In particular, we make the following contributions:

1. We cast image retrieval as *linear filtering* over a graph, efficiently performed in the *frequency domain*.
2. We provide a truly scalable solution to computing an *approximate Fourier basis* of the graph offline, accompanied by performance bounds.

[1]We avoid the term *diffusion* [11, 24] in this work.

3. *We reduce manifold search to a two-stage similarity search* thanks to an explicit embedding.
4. A rich set of interpretations connects to different fields.

The text is structured as follows. Section 2 describes the addressed problem while Sections 3 and 4 present a description and an analysis of our method respectively. Section 5 gives a number of interpretations and connections to different fields. Section 6 discusses our contributions against related work. We report experimental findings in Section 8 and draw conclusions in Section 9.

2. Problem

In this section we state the problem addressed by this paper in detail. We closely follow the formulation of [24].

2.1. Representation

A set of n descriptor vectors $\mathcal{V} = \{\mathbf{v}_1, \dots, \mathbf{v}_n\}$, with each $\mathbf{v}_i$ associated to vertex v_i of a weighted undirected graph G is given as an input. The graph G with n vertices $V = \{v_1, \dots, v_n\}$ and ℓ edges is represented by its $n \times n$ symmetric nonnegative adjacency matrix W. Graph G contains no self-loops, *i.e.* W has zero diagonal. We assume W is sparse with $2\ell \ll n(n-1)$ nonzero elements.

We define the $n \times n$ *degree matrix* $D := \operatorname{diag}(W\mathbf{1})$ where $\mathbf{1}$ is the all-ones vector, and the *symmetrically normalized adjacency matrix* $\mathcal{W} := D^{-1/2}WD^{-1/2}$ with the convention $0/0 = 0$. We also define the *Laplacian* and *normalized Laplacian* of G as $L := D - W$ and $\mathcal{L} := D^{-1/2}LD^{-1/2} = I - \mathcal{W}$, respectively. Both are singular and positive-semidefinite; the eigenvalues of $\mathcal{L}$ are in the interval $[0, 2]$ [8]. Hence, if $\lambda_1, \dots, \lambda_n$ are the eigenvalues of $\mathcal{W}$, its *spectral radius* $\varrho(\mathcal{W}) := \max_i |\lambda_i|$ is 1. Each eigenvector $\mathbf{u}$ of L associated to eigenvalue 0 is constant within connected components (*e.g.*, $L\mathbf{1} = D\mathbf{1} - W\mathbf{1} = \mathbf{0}$), while the corresponding eigenvector of $\mathcal{L}$ is $D^{1/2}\mathbf{u}$.

2.2. Transfer function

We define the $n \times n$ matrices $L_\alpha := \beta^{-1}(D - \alpha W)$ and $\mathcal{L}_\alpha := D^{-1/2}L_\alpha D^{-1/2} = \beta^{-1}(I - \alpha\mathcal{W})$, where $\alpha \in [0, 1)$ and $\beta := 1 - \alpha$. Both are positive-definite. Given the $n \times 1$ sparse observation vector $\mathbf{y}$ online, [24] computes the $n \times 1$ *ranking vector* $\mathbf{x}$ as the solution of the linear system

$$\mathcal{L}_\alpha \mathbf{x} = \mathbf{y}. \tag{1}$$

We can write the solution as $h_\alpha(\mathcal{W})\mathbf{y}$, where

$$h_\alpha(\mathcal{W}) := (1 - \alpha)(I - \alpha\mathcal{W})^{-1} \tag{2}$$

for a matrix $\mathcal{W}$ such that $I - \alpha\mathcal{W}$ is nonsingular; indeed, $\mathcal{L}_\alpha^{-1} = h_\alpha(\mathcal{W})$. Here we generalize this problem by considering any given *transfer function* $h : \mathcal{S} \to \mathcal{S}$, where $\mathcal{S}$ is the set of real symmetric matrices including scalars, $\mathbb{R}$. The general problem is then to compute

$$\mathbf{x}^* := h(\mathcal{W})\mathbf{y} \tag{3}$$

efficiently, in the sense that $h(\mathcal{W})$ is never explicitly computed or stored: $\mathcal{W}$ is given in advance and we are allowed to pre-process it *offline*, while both $\mathbf{y}$ and h are given *online*. For h_α in particular, we look for a more efficient solution than solving linear system (1).

2.3. Retrieval application

The descriptors $\mathcal{V}$ are generated by extracting image descriptors from either whole images, or from multiple sampled rectangular image regions, which can be optionally reduced by a Gaussian mixture model as in [24]. Note that the global descriptor is a special case of the regional one, using a single region per image. In the paper, we use CNN-based descriptors [45].

The undirected graph G is a k-NN similarity graph constructed as follows. Given two descriptors $\mathbf{v}, \mathbf{z}$ in $\mathbb{R}^d$, their *similarity* is measured as $s(\mathbf{v}, \mathbf{z}) = [\mathbf{v}^\top \mathbf{z}]_+^\gamma$, where exponent $\gamma > 0$ is a parameter. We denote by $s(\mathbf{v}_i|\mathbf{z})$ the similarity $s(\mathbf{v}_i, \mathbf{z})$ if $\mathbf{v}_i$ is a k-NN of $\mathbf{z}$ in $\mathcal{V}$ and zero otherwise. The symmetric *adjacency matrix* W is defined as $w_{ij} := \min(s(\mathbf{v}_i|\mathbf{v}_j), s(\mathbf{v}_j|\mathbf{v}_i))$, representing *mutual* neighborhoods. Online, given a query image represented by descriptors $\{\mathbf{q}_1, \dots, \mathbf{q}_m\} \subset \mathbb{R}^d$, the *observation vector* $\mathbf{y} \in \mathbb{R}^n$ is formed with elements $y_i := \sum_{j=1}^m s(\mathbf{v}_i|\mathbf{q}_j)$ by pooling over query regions. We make $\mathbf{y}$ sparse by keeping the k largest entries and dropping the rest.

3. Method

This section presents our *fast spectral ranking* (FSR) algorithm in abstract form first, then with concrete choices.

3.1. Algorithm

We describe our algorithm given an arbitrary $n \times n$ matrix $A \in \mathcal{S}$ instead of $\mathcal{W}$. Our solution is based on a sparse low-rank approximation of A computed offline such that online, $\mathbf{x} \approx h(A)\mathbf{y}$ is reduced to a sequence of sparse matrix-vector multiplications. The approximation is based on a randomized algorithm [47] that is similar to *Nyström sampling* [12] but comes with performance guarantees [18, 68]. In the following, $r \ll n$, $p < r$, q and τ are given parameters, and $\hat{r} = r + p$.

1. (*Offline*) Using *simultaneous iteration* [62, §28], compute an $n \times \hat{r}$ matrix Q with orthonormal columns that represents an approximate basis for the range of A, *i.e.* $QQ^\top A \approx A$. In particular, this is done as follows [18, §4.5]: randomly draw an $n \times \hat{r}$ standard Gaussian matrix $B^{(0)}$ and repeat for $t = 0, \dots, q-1$:

(a) Compute QR factorization $Q^{(t)}R^{(t)} = B^{(t)}$.
(b) Define the $n \times \hat{r}$ matrix $B^{(t+1)} := AQ^{(t)}$.

Finally, set $Q := Q^{(q-1)}$, $B := B^{(q)} = AQ$.

2. (*Offline–Fourier basis*) Compute a rank-r eigenvalue decomposition $U\Lambda U^\top \approx A$, where $n \times r$ matrix U has orthonormal columns and $r \times r$ matrix Λ is diagonal. In particular, roughly following [18, §5.3]:
   (a) Form the $\hat{r} \times \hat{r}$ matrix $C := Q^\top B = Q^\top AQ$.
   (b) Compute its eigendecomposition $\hat{V}\hat{\Lambda}\hat{V}^\top = C$.
   (c) Form (V, Λ) by keeping from $(\hat{V}, \hat{\Lambda})$ the slices (rows/columns) corresponding to the r largest eigenvalues.
   (d) Define the matrix $U := QV$.
3. (*Offline*) Make U sparse by keeping its τ largest entries and dropping the rest.
4. (*Online*) Given $\mathbf{y}$ and h, compute

$$\mathbf{x} := Uh(\Lambda)U^\top \mathbf{y}. \tag{4}$$

Observe that $U^\top$ projects $\mathbf{y}$ onto $\mathbb{R}^r$. With Λ being diagonal, $h(\Lambda)$ is computed element-wise. Finally, multiplying by U and ranking $\mathbf{x}$ amounts to dot product similarity search in $\mathbb{R}^r$. The online stage is very fast, provided U only contains few leading eigenvectors and $\mathbf{y}$ is sparse. We consider the following variants:

- FSR.SPARSE: This is the complete algorithm.
- FSR.APPROX: Drop sparsification stage 3.
- FSR.RANK-r: Drop approximation stage 1 and sparsification stage 3. Set $\hat{r} = n$, $Q = I$, $B = A$ in stage 2.
- FSR.EXACT: same as FSR.RANK-r for $r = n$.

To see why FSR.EXACT works, consider the case of $h_\alpha(\mathcal{W})$. Let $\mathcal{W} \simeq U\Lambda U^\top$. It follows that $h_\alpha(\mathcal{W})/\beta = (I - \alpha\mathcal{W})^{-1} \simeq U(I - \alpha\Lambda)^{-1}U^\top$, where $(I - \alpha\Lambda)^{-1}$ is computed element-wise. Then, $\mathbf{x}^* \simeq \beta U(I - \alpha\Lambda)^{-1}U^\top\mathbf{y}$. The general case is discussed in section 4.

3.2. Retrieval application

Returning to the retrieval problem, we compute the ranking vector $\mathbf{x} \in \mathbb{R}^n$ by (4), containing the *ranking score* x_i of each dataset region $\mathbf{v}_i$. To obtain a score per image, we perform a linear pooling operation [24] represented as $\overline{\mathbf{x}} := \Sigma\mathbf{x}$ where Σ is a sparse $N \times n$ *pooling matrix*. The $N \times r$ matrix $\overline{U} := \Sigma U$ is indeed computed offline so that we directly compute $\overline{\mathbf{x}} = \overline{U}h(\Lambda)U^\top\mathbf{y}$ online.

Computing $\mathbf{y}$ involves Euclidean search in $\mathbb{R}^d$, which happens to be dot product because vectors are ℓ^2-normalized. Applying $\overline{U}$ and ranking $\mathbf{x}$ amounts to a dot product similarity search in $\mathbb{R}^r$. We thus **reduce manifold search to Euclidean followed by dot product search**. The number of nonzero elements of $\mathbf{y}$ and rows of $\overline{U}$, whence the cost, are the same for global or regional search.

4. Analysis

We derive the asymptotic space and time complexity of different algorithm variants and derive necessary condition for correctness and error bounds of approximate variants.

4.1. Complexity

The offline complexity is mainly determined by the number of columns $\hat{r}$ of matrix Q: Stage 1 reduces the size of the problem from n^2 down to $n\hat{r}$. The online complexity is determined by the number of nonzero entries in matrix U. A straightforward analysis leads to the following:

- FSR.APPROX: The offline complexity is $O(qn(k + \hat{r})\hat{r})$ time and $O(n\hat{r})$ space; its online (time and space) complexity is $O(nr)$.
- FSR.SPARSE: The offline complexity is $O(qn(k + \hat{r})\hat{r} + \tau \log \tau)$ time and $O(n\hat{r})$ space; its online complexity is $O(\tau)$.

Stage 1 is "embarrassingly parallelizable" meaning that it is dramatically accelerated on parallel and distributed platforms. Since the online stage 4 amounts to NN search, any approximate method applies, making it sublinear in n.

4.2. Correctness

We derive here the conditions on h and A under which our algorithm is correct under no truncation, *i.e.*, $\text{FSR.EXACT}(\mathbf{y}|A, h) = h(A)\mathbf{y}$. We also show, that h_α and $\mathcal{W}$ satisfy these conditions, which is an alternative proof of correctness to the one in Section 3.1.

Starting from the fact a real symmetric matrix A is diagonalizable, there exists an exact eigenvalue decomposition $U\Lambda U^\top = A$, where U is orthogonal. According to [1, §9.14,9.2], we have $h(A) = Uh(\Lambda)U^\top = U\,\mathrm{diag}(h(\lambda_1), \ldots, h(\lambda_n))U^\top$ if and only if there exists a series expansion of h converging for this specific A:

$$h(A) = \sum_{t=0}^{\infty} c_t A^t. \tag{5}$$

This holds in particular for h_α admitting the *geometric progression* expansion

$$h_\alpha(A) := \beta(I - \alpha A)^{-1} = \beta \sum_{t=0}^{\infty} (\alpha A)^t, \tag{6}$$

which converges absolutely if $\varrho(\alpha A) < 1$ [1, §9.6,9.19]. This holds for $A = \mathcal{W}$ because $\alpha < 1$ and $\varrho(\mathcal{W}) = 1$.

4.3. Error bound

We present main ideas for bounding the approximation error of FSR.RANK-r and FSR.APPROX coming from literature, and we derive another condition on h under which

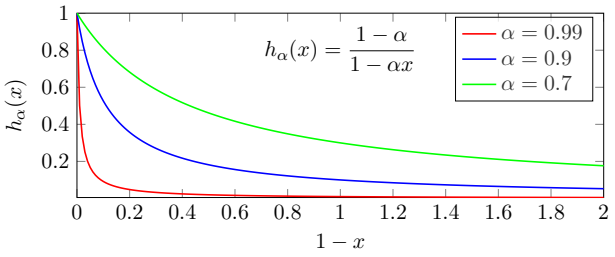


Figure 2: Function h_α (2) is a 'low-pass filter'; $1-x$ represents eigenvalues of $\mathcal{L}$, where 0 is the DC component.

our algorithm is valid under truncation. The approximation $QQ^\top A \approx A$ of stage 1 is studied in [18, §9.3,10.4]: an average-case bound on $\left\|A - QQ^\top A\right\|$ decays exponentially fast in the number of iterations q to $|\lambda_{r+1}|$. Stage 2 yields an approximate eigenvalue decomposition of A: Since A is symmetric, $A \approx QQ^\top AQQ^\top = QCQ^\top \approx QV\Lambda V^\top Q^\top = U\Lambda U^\top$. The latter approximation $C \approx V\Lambda V^\top$ is essentially a best rank-r approximation of $C = Q^\top AQ$. This is also studied in [18, §9.4] for the truncated SVD case of a non-symmetric matrix. It involves an additional term of $|\lambda_{r+1}|$ in the error.

We are actually approximating $h(A)$ by $Uh(\Lambda)U^\top$, so that $|h(\lambda_{r+1})|$ governs the error instead of $|\lambda_{r+1}|$. A similar situation appears in [61, §3.3]. Therefore, our method makes sense only when the restriction of h to scalars is *non-decreasing*. This is the case for h_α.

5. Interpretation

Our work is connected to studies in different fields with a long history. Here we give a number of interpretations both in general and in the particular case $h = h_\alpha$.

5.1. Graph signal processing

In *signal processing* [38], a discrete-time *signal* of period n is a vector $\mathbf{s} \in \mathbb{R}^n$ where indices are represented by integers modulo n, that is, $s_{\bar{i}} := s_{(i \bmod n)+1}$ for $i \in \mathbb{Z}$. A *shift* (or translation, or delay) of $\mathbf{s}$ by one sample is the mapping $s_{\bar{i}} \mapsto s_{\overline{i-1}}$. If we define the $n \times n$ circulant matrix $C_n := (\mathbf{e}_2\ \mathbf{e}_3 \dots \mathbf{e}_n\ \mathbf{e}_1)$[2], a shift can be represented by $\mathbf{s} \mapsto C_n\mathbf{s}$ [50]. A linear, time (or shift) invariant *filter* is the mapping $\mathbf{s} \mapsto H\mathbf{s}$ where H is an $n \times n$ matrix with a series representation $H := h(C_n) = \sum_{t=0}^{\infty} h_t C_n^t$. Matrix C_n has the eigenvalue decomposition $U\Lambda U^\top$ where $U^\top$ is the $n \times n$ *discrete Fourier transform* matrix $\mathcal{F}$. If the series $h(C_n)$ converges, filtering $\mathbf{s} \mapsto H\mathbf{s}$ is written as

$$\mathbf{s} \mapsto \mathcal{F}^{-1} h(\Lambda) \mathcal{F}\mathbf{s}. \tag{7}$$

That is, $\mathbf{s}$ is mapped to the *frequency domain*, scaled element-wise, and mapped back to the time domain.

[2]Observe that C_n is the adjacency matrix of the directed graph of Figure 1 after adding an edge from the rightmost to the leftmost vertex.

Graph signal processing [50, 54] generalizes the above concepts to graphs by replacing C_n by $\mathcal{W}$, an appropriately normalized adjacency matrix of an arbitrary graph. If $U\Lambda U^\top$ is the eigenvalue decomposition of $\mathcal{W}$, we realize that (4) treats $\mathbf{y}$ as a (sparse) *signal* and filters it in the frequency domain via transfer function h to obtain $\mathbf{x}$. Function h_α in particular is a *low-pass filter*, as illustrated in Figure 2. By varying α from 0 to 1, the frequency response varies from all-pass to sharp low-pass.

5.2. Random walks

Consider the iterating process: for $t = 1, 2, \dots$

$$\mathbf{x}^{(t)} := \alpha A\mathbf{x}^{(t-1)} + (1-\alpha)\mathbf{y}. \tag{8}$$

If A is a stochastic *transition matrix* and $\mathbf{x}^{(0)}, \mathbf{y}$ are distributions over vertices, this specifies a random walk on a (directed) graph: at each iteration a particle moves to a neighboring vertex with probability α or jumps to a vertex according to distribution $\mathbf{y}$ with probability $1-\alpha$. This is called a *Markov chain with restart* [2] or *random walk with restart* [40]. State $\mathbf{x}^{(t)}$ converges to $\mathbf{x}^* = h_\alpha(A)\mathbf{y}$ as $t \to \infty$ provided $\varrho(\alpha A) < 1$ [69]. In fact, (8) is equivalent to *Jacobi solver* [17] on linear system (1) [24].

If $\mathbf{y} = \mathbf{e}_i$, the i-th canonical vector, then $\mathbf{x}^*$ is used to rank the vertices of G, expressing a measure of "similarity" to v_i [70]. Parameter α controls how much $\mathbf{x}^*$ is affected by *boundary condition* $\mathbf{y}$ [64]: $\mathbf{x}^*$ equals $\mathbf{y}$ for $\alpha = 0$, while in the limit $\alpha \to 1$, $\mathbf{x}^*$ tends to a dominant eigenvector of A. Indeed, for $\alpha = 1$, (8) becomes a power iteration.

5.3. Random fields

Given a positive-definite $n \times n$ *precision matrix* $A \in \mathcal{S}$ and a *mean vector* $\boldsymbol{\mu} \in \mathbb{R}^n$, a *Gaussian Markov random field* (GMRF) [49] with respect to an undirected graph G is a random vector $\mathbf{x} \in \mathbb{R}^n$ with normal density $p(\mathbf{x}) := \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, A^{-1})$ iff A has the same nonzero off-diagonal entries as the adjacency matrix of G. Its *canonical parametrization* $p(\mathbf{x}) \propto e^{-E(\mathbf{x}|\mathbf{b},A)}$ where $E(\mathbf{x}|\mathbf{b}, A) := \frac{1}{2}\mathbf{x}^\top A\mathbf{x} - \mathbf{b}^\top\mathbf{x}$ is a quadratic *energy*. Its expectation $\boldsymbol{\mu} = A^{-1}\mathbf{b}$ is the minimizer of this energy. Now, $\mathbf{x}^* = \mathcal{L}_\alpha^{-1}\mathbf{y}$ (1) is the expectation of a GMRF with energy

$$f_\alpha(\mathbf{x}) := E(\mathbf{x}|\mathbf{y}, \mathcal{L}_\alpha) = \frac{1}{2}\mathbf{x}^\top \mathcal{L}_\alpha \mathbf{x} - \mathbf{y}^\top\mathbf{x}. \tag{9}$$

A *mean field* method on this GMRF is equivalent to *Jacobi* or *Gauss-Seidel* solvers on (1) [66]. Yet, *conjugate gradients* (CG) [37] is minimizing $f_\alpha(\mathbf{x})$ more efficiently [24, 5].

If we expand $f_\alpha(\mathbf{x})$ using $\beta\mathcal{L}_\alpha = \alpha\mathcal{L} + (1-\alpha)I$, we find that it has the same minimizer as

$$\alpha \sum_{i,j} w_{ij} \left\|\hat{x}_i - \hat{x}_j\right\|^2 + (1-\alpha)\left\|\mathbf{x} - \mathbf{y}\right\|^2, \tag{10}$$

where $\hat{\mathbf{x}} := D^{-1/2}\mathbf{x}$. The pairwise *smoothness term* encourages $\mathbf{x}$ to vary little across edges with large weight whereas the unary *fitness term* to stay close to observation $\mathbf{y}$ [69]. Again, α controls the trade-off: $\mathbf{x}^*$ equals $\mathbf{y}$ for $\alpha = 0$, while for $\alpha \to 1$, $\mathbf{x}^*$ tends to be constant over connected components like dominant eigenvectors of $\mathcal{W}$.

5.4. Regularization and kernels

The first term of (9) is interpreted as a *regularization operator* related to a kernel $K = \mathcal{L}_\alpha^{-1}$ [58, 57, 31]. In a finite graph, a *kernel* can be seen either as an $n \times n$ matrix K or a function $\kappa : V^2 \to \mathbb{R}$ operating on pairs of vertices. More generally, if $h(x) > 0$ for $x \in \mathbb{R}$, which holds for h_α, then $K := h(\mathcal{W})$ is positive-definite and there is an $n \times n$ matrix Φ such that $K = \Phi^\top \Phi$, or $\kappa(v_i, v_j) = \phi(v_i)^\top \phi(v_j)$ where *feature map* $\phi : V \to \mathbb{R}^n$ is given by $\phi(v_i) := \Phi \mathbf{e}_i$. A particular choice for Φ is

$$\Phi := h(\Lambda)^{1/2} U^\top \tag{11}$$

where $U \Lambda U^\top$ is the eigenvalue decomposition of $\mathcal{W}$. If we choose a rank-r approximation instead, then Φ is an $r \times n$ matrix and ϕ is a low-dimensional *embedding* onto $\mathbb{R}^r$.

The goal of *out-of-sample extension* is to compute a "similarity" $\hat{\kappa}(\mathbf{z}_1, \mathbf{z}_2)$ between two unseen vectors $\mathbf{z}_1, \mathbf{z}_2 \in \mathbb{R}^d$ not pertaining to the graph. Here we define

$$\hat{\kappa}(\mathbf{z}_1, \mathbf{z}_2) := \psi(\mathbf{z}_1)^\top \Phi^\top \Phi \psi(\mathbf{z}_2) \tag{12}$$

given any mapping $\psi : \mathbb{R}^d \to \mathbb{R}^n$, *e.g.* $\psi(\mathbf{z})_i := s(\mathbf{v}_i|\mathbf{z})$ discussed in section 2. This extended kernel is also positive-definite and its embedding $\hat{\phi}(\mathbf{z}) = \Phi \psi(\mathbf{z})$ is a linear combination of the dataset embeddings. For $r \ll n$, our method allows rapid computation of κ or $\hat{\kappa}$ for any given function h, without any dense $n \times n$ matrix involved.

5.5. Paths on graphs

Many *nonlinear dimension reduction* methods replace Euclidean distance with an approximate *geodesic distance*, assuming the data lie on a *manifold* [33]. This involves the *all-pairs shortest path* (APSP) problem and Dijkstra's algorithm is a common choice. Yet, it is instructive to consider a naïve algorithm [9, §25.1]. We are given a *distance matrix* where missing edges are represented by ∞ and define similarity weight $w_{ij} = e^{-d_{ij}}$. A path weight is a now a product of similarities and "shortest" means "of maximum weight". Defining matrix power $A^{\otimes t}$ as A^t with $+$ replaced by $\max$, the algorithm is reduced to computing $\max_t W^{\otimes t}$ (element-wise). Element i, j of $W^{\otimes t}$ is the weight of the shortest path of length t between v_i, v_j.

Besides their complexity, shortest paths are sensitive to changes in the graph. An alternative is the *sum*[3] of weights over paths of length t, recovering the ordinary matrix power $\mathcal{W}^t$, and the weighted sum over all lengths $\sum_{t=0}^{\infty} c_t \mathcal{W}^t$, where coefficients $(c_t)_{t \in \mathbb{N}}$ allow for convergence [64], [52, §9.4]. This justifies (5) and reveals that coefficients control the contribution of paths depending on length. A common choice is $c_t = \beta \alpha^t$ with $\beta = 1 - \alpha$ and $\alpha \in [0, 1)$ being a *damping factor* [64], which justifies function h_α (6).

[3] In fact, similar to *softmax* due to the exponential and normalization.

6. Related work

The history of the particular case $h = h_\alpha$ is the subject of the excellent study of *spectral ranking* [64]. The fundamental contributions originate in the social sciences and include the eigenvector formulation by Seeley [51], damping by α (6) by Katz [29] and the boundary condition $\mathbf{y}$ (1) by Hubbell [22]. The most well-known follower is PageRank [39]. In machine learning, h_α has been referred to as the *von Neumann* [27, 52] or *regularized Laplacian* kernel [57]. Along with the *diffusion kernel* [32, 31], it has been studied in connection to *regularization* [58, 57].

Random fields are routinely used for low-level vision tasks where one is promoting smoothness while respecting a noisy observation, like in *denoising* or *segmentation*, where both the graph and the observation originate from a single image [59, 5]. A similar mechanism appears in *semi-supervised learning* [69, 73, 71, 6] or *interactive segmentation* [16, 30] where the observation is composed of labels over a number of samples or pixels. In our *retrieval* scenario, the observation is formed by the neighbors in the graph of an external query image (or its regions).

The *random walk* or *random walk with restart* (RWR) formulation [70, 69, 40] is an alternative interpretation to retrieval [11]. Yet, directly solving a linear system is superior [24]. Offline matrix decomposition has been studied for RWR [61, 13, 26]. All three methods are limited to h_α while sparse LU decomposition [13, 26] assumes an uneven distribution of vertex degrees [28], which is not the case for k-NN graphs. In addition, we reduce *manifold search* to two-stage Euclidean search via an explicit embedding, which is data dependent through the kernel $K = \mathcal{L}_\alpha^{-1}$.

In the general case, the spectral formulation (4) has been known in machine learning [6, 52, 36, 72, 65] and in graph signal processing [50, 54, 19]. The latter is becoming popular in the form of *graph-based convolution* in deep learning [4, 21, 10, 3, 34, 43]. However, with few exceptions [4, 21], which rely on an expensive decomposition, there is nothing spectral when it comes to actual computation. It is rather preferred to work with finite polynomial approximations of the graph filter [10, 3] using *Chebyshev polynomials* [19, 55] or translation-invariant neighborhood templates in the spatial domain [34, 43].

We cast *retrieval as graph filtering* by constructing an appropriate observation vector. We actually perform the computation in the *frequency domain* via a scalable so-

lution. Comparing to other applications, retrieval conveniently allows offline computation of the graph Fourier basis and online reuse to embed query vectors. An alternative is to use *random projections* [63, 48]. This roughly corresponds to a single iteration of our step 1. Our solution is thus more accurate, while h is specified online.

7. Practical considerations

Block diagonal case. Each connected component of G has a maximal eigenvalue 1. These maxima of small components dominate the eigenvalues of the few (or one) "giant" component that contain the vast majority of data [28]. For this reason we find the connected components with the *union-find* algorithm [9] and reorder vertices such that A is block diagonal: $A = \mathrm{diag}(A_1, \ldots, A_c)$. For each $n_l \times n_l$ matrix A_l, we apply offline stages 1-3 to obtain an approximate rank-r_l eigenvalue decomposition $\hat{U}_l \hat{\Lambda}_l \hat{U}_l^\top \approx A_l$ with $r_l = \max(\rho, \lceil r n_l / n \rceil)$ if $n_l > \rho$, otherwise we compute an exact decomposition. Integer ρ is a given parameter. We form (U_l, Λ_l) by keeping up to ρ slices from each pair $(\hat{U}_l, \hat{\Lambda}_l)$ and complete with up to r slices in total, associated to the largest eigenvalues of the entire set $\mathrm{diag}(\hat{\Lambda}_1, \ldots, \hat{\Lambda}_c)$. Online, we partition $(\mathbf{y}_1; \ldots; \mathbf{y}_c) = \mathbf{y}$, compute each $\mathbf{x}_l$ from $\mathbf{y}_l$ by (4) and form back vector $\mathbf{x} = (\mathbf{x}_1; \ldots; \mathbf{x}_c)$.

Sparse neighborhoods. Denote by η_i the ℓ_2-norm of the i-th row of U. FSR.EXACT yields $\boldsymbol{\eta} = \mathbf{1}$ but this is not the case for FSR.RANK-r. Larger (smaller) values appear to correspond to densely (sparsely) populated parts of the graph. For small rank r, norms η_i are more severely affected for uncommon vectors in the dataset. We propose replacing each element x_i of (4) by

$$x_i' = x_i + (1 - \eta_i) \mathbf{v}_i^\top \mathbf{q}, \tag{13}$$

for global descriptors, with a straightforward extension for regional ones. This is referred to as FSRw and is a weighted combination of manifold search and Euclidean search. It approaches the former for common elements and to the latter for uncommon ones. Our experiments show that this is essential at large scale.

8. Experiments

This section introduces our experimental setup, investigates the performance and behavior of the proposed method and its application to large-scale image retrieval.

8.1. Experimental Setup

Datasets. We use three image retrieval benchmarks: Oxford Buildings (Oxford5k) [41], Paris (Paris6k) [42] and Instre [67], with the evaluation protocol introduced in [24] for the latter. We conduct large-scale experiments by following a standard protocol of adding 100k distractor images from Flickr [41] to Oxford5k and Paris6k, forming the so called Oxford105k and Paris106k. Mean average precision (mAP) evaluates the retrieval performance in all datasets.

Image Descriptors. We apply our method on the same global and regional image descriptors as in [24]. In particular, we work with d-dimensional vectors extracted from VGG [56] ($d = 512$) and ResNet101 [20] ($d = 2,048$) networks fine-tuned specifically for image retrieval [45, 15]. Global description is R-MAC with 3 different scales [60], including the full image as a separate region. Regional descriptors consist of the same regions as those involved in R-MAC but without sum pooling, resulting in 21 vectors per image on average. Global and regional descriptors are processed by supervised whitening [45].

Implementation. We adopt the same parameters for graph construction and search as in [24]. The pairwise descriptor similarity is defined as $s(\mathbf{v}, \mathbf{z}) = [\mathbf{v}^\top \mathbf{z}]_+^3$. We use $\alpha = 0.99$, and keep the top $k = 50$ and $k = 200$ mutual neighbors in the graph for global and regional vectors, respectively. These choices make our experiments directly comparable to prior results on manifold search for image retrieval with CNN-based descriptors [24]. In all our FSR.APPROX experiments, we limit the algorithm within the largest connected component only, while each element x_i for vertex v_i in any other component is just copied from y_i. This choice works well because the largest component holds nearly all data in practice. Following [24], generalized max-pooling [35, 23] is used to pool regional diffusion scores per image. Reported *search times* exclude the construction of the observation vector $\mathbf{y}$, since this task is common to all baseline and our methods. Time measurements are reported with a 4-core Intel Xeon 2.00GHz CPU.

8.2. Retrieval Performance

Rank-r. We evaluate the performance of FSR.RANK-r for varying rank r, which affects the quality of the approximation and defines the dimensionality of the embedding space. As shown in Figure 4, the effect of r depends on the dataset. In all cases the optimal performance is already reached at $r = 1$k. On Paris6k in particular, this happens as soon as $r = 100$. Compared to FSR.EXACT as implemented in [24], it achieves the same mAP but 150 times faster on Oxford5k and Paris6k and 300 times faster on Instre. Global search demonstrates a similar behavior.

We achieve 97.0 mAP on Paris6k, which is near-perfect. Figure 3 shows the two queries with the lowest AP and their top-ranked negative images. In most cases the ground-truth is incorrect, as these images have visual overlap with the query bounding box. The first correct negative image for "La Défense" appears at rank 126, where buildings from the surroundings are retrieved due to "topic drift". The same happens with "Pyramide du Louvre", where the first correct negative image is at rank 108.

Figure 3: Two queries with the *lowest* AP from Paris6k (left) and the corresponding top-ranked negative images based on the ground-truth, with their rank underneath. Ranks are marked in blue for incorrectly labeled images, and red otherwise.

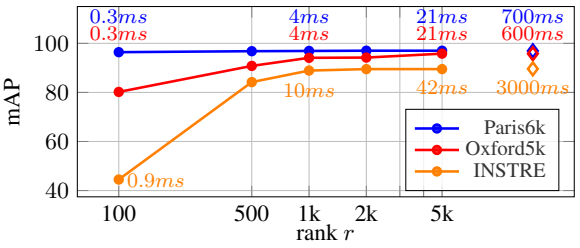


Figure 4: Performance of regional search with FSR.RANK-r. Runtimes are reported in text labels. $\diamond$ refers to FSR.EXACT performed with conjugate gradients as in [24]

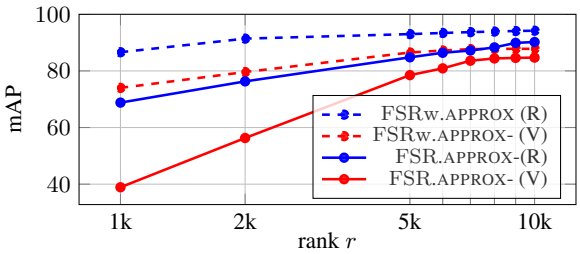


Figure 5: mAP *vs.* r on Oxford105k with FSR.APPROX and FSRw.APPROX, using Resnet101(R) and VGG(V).

Regional search performs better than global [24] at the cost of more memory and slower query. We unlock this bottleneck thanks to the offline pooling $\overline{U} = \Sigma U$. Indeed, global and regional search on Instre take $0.040s$ and $0.042s$ respectively with our method, while the corresponding times for FSR.EXACT are $0.055s$ and $3s$.

Approximate eigendecomposition keeps the off-line stage tractable at large scale. With 570k regional descriptors on Instre, FSR.RANK-5000 and FSR.APPROX yield a mAP of 89.5 and 89.2 respectively, with offline cost 60 and 3 hours respectively, using 16-core Intel Xeon 2.00GHz CPU. This is important at large scale because the off-line complexity of FSR.RANK-r is polynomial.

When new images are added, one can express them according to existing ones, as in (12). We evaluate such *extension* by constructing the graph on a random subset of 100%, 90%, 70%, 50%, 30% and 10% of Instre, yielding 80.5, 80.1, 78.3, 75.8, 70.2 and 40.6 mAP respectively on the entire dataset, with global search. The drop is graceful until 30%; beyond that, the graph needs to be updated.

8.3. Large-scale experiments

We now apply our approach to a larger scale by using only 5 descriptors per image using GMM reduction [24]. This choice improves scalability while minimizing the accuracy loss.

FSRw.**APPROX** becomes crucial, especially at large scale, because vectors of sparsely populated parts of the graph are not well represented. Figure 5 shows the comparison between FSRw.APPROX and FSR.APPROX. We achieve 90.2 and 94.2 with FSR.APPROX and FSRw.APPROX respectively, with $r = 10$k and Resnet101 descriptors.

We further report the performance separately for each of the 11 queries of Oxford105k dataset. Results are shown in Figure 6. Low values of r penalize sparsely populated parts of the graph, *i.e.* landmarks with less similar instances in the dataset. FSRw.APPROX partially solves this issue.

The search time is 0.14s and 0.3s per query for $r = 5$k and $r = 10$k respectively on Oxford105k. It is two orders of magnitude faster than FSR.EXACT: The implementation of [24] requires about 14s per query, which is reduced to 1s with dataset truncation: manifold search is a re-ranking only applied to top-ranked images. We do *not* use any truncation. This improves the mAP by 3% and our method is still one order of magnitude faster.

Sparse embeddings. Most descriptors belong only to few manifolds and each embedding vector has high energy in the corresponding components. Setting $r = 10$k, large enough to avoid compromising accuracy, Figure 7 shows the effect of sparsifying the embeddings with FSRw.SPARSE on Oxford105k. Remarkably, we can make up to 90% memory savings with only %2 drop of mAP.

Quantized descriptors. Construction of the observation vector requires storing the initial descriptors. We further

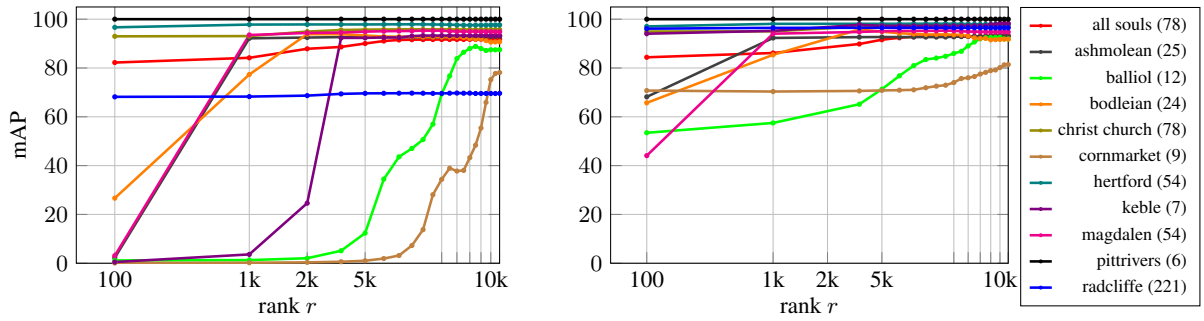


Figure 6: mAP *vs.* rank r separately per landmark in Oxford105k with FSR.APPROX (left) and FSRw.APPROX (right). Number of positive images per landmark is shown in the legend.

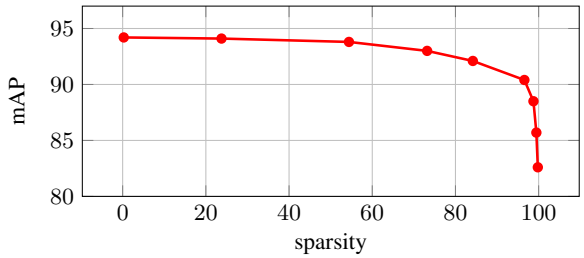


Figure 7: mAP *vs.* sparsity of U by keeping its τ largest values and varying τ with FSRw.SPARSE on Oxford105k, Resnet101 descriptors and rank $r = 10$k.

use product quantization (PQ) [25] to compress them. Using FSRw.APPROX on Oxford105k, mAP drops from 94.4 with uncompressed descriptors to 94.2 and 91.1 with 256- and 64-byte PQ codes, respectively.

8.4. Comparison to other methods

Table 1 compares our method with the state-of-the-art. We report results for $r = 5$k, FSR.RANK-r for global description, FSR.APPROX for regional description, and FSRw.APPROX in large-scale (with 100k distractors) and regional experiments. GMM reduces the number of regions per image from 21 to 5 [24]. We do not experiment at large-scale without GMM since there is not much improvement and it is less scalable. Our method reaches performance similar to that of FSR.EXACT as evaluated with CG [24]. Our benefit comes from the dramatic speed-up. For the first time, manifold search runs almost as fast as Euclidean search. Consequently, dataset truncation is no longer needed and this improves the mAP.

Method	$m \times d$	INSTRE	Oxf5k	Oxf105k	Par6k	Par106k
Global descriptors - Euclidean search						
R-MAC [45]	512	47.7	77.7	70.1	84.1	76.8
R-MAC [15]	2,048	62.6	83.9	80.8	93.8	89.9
Global descriptors - Manifold search						
Diffusion [24]	512	70.3	85.7	82.7	94.1	92.5
FSR.RANK-r	512	70.3	85.8	85.0	93.8	92.4
Diffusion [24]	2,048	80.5	87.1	87.4	96.5	95.4
FSR.RANK-r	2,048	80.5	87.5	87.9	96.4	95.3
Regional descriptors - Euclidean search						
R-match [46]	21×512	55.5	81.5	76.5	86.1	79.9
R-match [46]	21×2,048	71.0	88.1	85.7	94.9	91.3
Regional descriptors - Manifold search						
Diffusion [24]	5×512	77.5	91.5	84.7	95.6	93.0
FSR.APPROX	5×512	78.4	91.6	86.5	95.6	92.4
Diffusion [24]	21×512	80.0	93.2	90.3	96.5	92.6
FSR.APPROX	21×512	80.4	93.0	-	96.5	-
Diffusion [24]	5×2,048	88.4	95.0	90.0	96.4	95.8
FSR.APPROX	5×2,048	88.5	95.1	93.0	96.5	95.2
Diffusion [24]	21×2,048	89.6	95.8	94.2	96.9	95.3
FSR.APPROX	21×2,048	89.2	95.8	-	97.0	-

Table 1: Performance comparison to the baseline methods and to the state of the art on manifold search [24]. Points at 512D are extracted with VGG [45] and at 2048D with ResNet101 [15]. Regional representation with $m = 5$ descriptors per image uses GMM. Large-scale regional experiments use the FSRw.APPROX variant. Dataset truncation is used in [24] at large scale.

9. Discussion

This work reproduces the excellent results of online linear system solution [24] at fraction of query time. We even improve performance by avoiding to truncate the graph online. The offline stage is linear in the dataset size, embarrassingly parallelizable and takes a few hours in practice for the large scale datasets of our experiments. The approximation quality is arbitrarily close to the optimal one at a given embedding dimensionality. The required dimensionality for good performance is large but in practice the embedded vectors are very sparse. This resembles an encoding based on a large vocabulary, searched via an inverted index. Our method is generic and may be used for problems other than search, including clustering and unsupervised or semi-supervised learning.

Acknowledgments The authors were supported by the MSMT LL1303 ERC-CZ grant. The Tesla K40 used for this research was donated by the NVIDIA Corporation. The authors would like to thank James Pritts for fruitful discussions during this work.

References

[1] K. M. Abadir and J. R. Magnus. *Matrix algebra*. Cambridge University Press, 2005. 3

[2] P. Boldi, V. Lonati, M. Santini, and S. Vigna. Graph fibrations, graph isomorphism, and PageRank. *RAIRO-Theoretical Informatics and Applications*, 40(2):227–253, 2006. 4

[3] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst. Geometric deep learning: going beyond euclidean data. *arXiv preprint arXiv:1611.08097*, 2016. 5

[4] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013. 5

[5] S. Chandra and I. Kokkinos. Fast, exact and multi-scale inference for semantic image segmentation with deep Gaussian CRFs. In *ECCV*, pages 402–418, 2016. 4, 5

[6] O. Chapelle, J. Weston, and B. Scholkopf. Cluster kernels for semi-supervised learning. *NIPS*, pages 601–608, 2003. 5

[7] O. Chum, J. Philbin, J. Sivic, M. Isard, and A. Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *ICCV*, October 2007. 1

[8] F. R. Chung. *Spectral graph theory*, volume 92. American Mathematical Soc., 1997. 2

[9] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to algorithms*. Massachusetts Institute of Technology, 2009. 5, 6

[10] M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *NIPS*, pages 3837–3845, 2016. 5

[11] M. Donoser and H. Bischof. Diffusion processes for retrieval revisited. In *CVPR*, 2013. 1, 5

[12] P. Drineas and M. W. Mahoney. On the Nyström method for approximating a gram matrix for improved kernel-based learning. *Journal of Machine Learning Research*, 6(Dec):2153–2175, 2005. 2

[13] Y. Fujiwara, M. Nakatsuji, M. Onizuka, and M. Kitsuregawa. Fast and exact top-k search for random walk with restart. *Proceedings of the VLDB Endowment*, 5(5):442–453, 2012. 5

[14] A. Gordo, J. Almazan, J. Revaud, and D. Larlus. Deep image retrieval: Learning global representations for image search. *ECCV*, 2016. 1

[15] A. Gordo, J. Almazan, J. Revaud, and D. Larlus. End-to-end learning of deep visual representations for image retrieval. *arXiv preprint arXiv:1610.07940*, 2016. 1, 6, 8

[16] L. Grady. Random walks for image segmentation. *IEEE Trans. PAMI*, 28(11):1768–1783, 2006. 5

[17] W. Hackbusch. *Iterative solution of large sparse systems of equations*. Springer Verlag, 1994. 4

[18] N. Halko, P.-G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2):217–288, 2011. 2, 3, 4

[19] D. K. Hammond, P. Vandergheynst, and R. Gribonval. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2):129–150, 2011. 5

[20] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 6

[21] M. Henaff, J. Bruna, and Y. LeCun. Deep convolutional networks on graph-structured data. *arXiv preprint arXiv:1506.05163*, 2015. 5

[22] C. H. Hubbell. An input-output approach to clique identification. *Sociometry*, 1965. 5

[23] A. Iscen, T. Furon, V. Gripon, M. Rabbat, and H. Jégou. Memory vectors for similarity search in high-dimensional spaces. *IEEE Trans. Big Data*, 4(1), 2018. 6

[24] A. Iscen, G. Tolias, Y. Avrithis, T. Furon, and O. Chum. Efficient diffusion on region manifolds: Recovering small objects with compact cnn representations. In *CVPR*, 2017. 1, 2, 3, 4, 5, 6, 7, 8

[25] H. Jégou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE Trans. PAMI*, 33(1):117–128, January 2011. 8

[26] J. Jung, K. Shin, L. Sael, and U. Kang. Random walk with restart on large graphs using block elimination. *ACM Transactions on Database Systems*, 41(2):12, 2016. 5

[27] J. Kandola, J. Shawe-Taylor, and N. Cristianini. Learning semantic similarity. In *NIPS*, 2002. 5

[28] U. Kang and C. Faloutsos. Beyond 'caveman communities': Hubs and spokes for graph compression and mining. In *Proceedings of the IEEE International Conference on Data Mining*, pages 300–309. IEEE, 2011. 5, 6

[29] L. Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, 1953. 5

[30] T. H. Kim, K. M. Lee, and S. U. Lee. Generative image segmentation using random walks with restart. In *ECCV*, pages 264–275. Springer, 2008. 5

[31] R. Kondor and J.-P. Vert. Diffusion kernels. *Kernel Methods in Computational Biology*, pages 171–192, 2004. 5

[32] R. I. Kondor and J. Lafferty. Diffusion kernels on graphs and other discrete structures. In *ICML*, 2002. 5

[33] J. A. Lee and M. Verleysen. *Nonlinear dimensionality reduction*. Springer Science & Business Media, 2007. 5

[34] F. Monti, D. Boscaini, J. Masci, E. Rodolà, J. Svoboda, and M. M. Bronstein. Geometric deep learning on graphs and manifolds using mixture model cnns. *arXiv preprint arXiv:1611.08402*, 2016. 5

[35] N. Murray and F. Perronnin. Generalized max-pooling. In *CVPR*, June 2014. 6

[36] B. Nadler, S. Lafon, R. R. Coifman, and I. G. Kevrekidis. Diffusion maps, spectral clustering and eigenfunctions of fokker-planck operators. *NIPS*, 2005. 5

[37] J. Nocedal and S. Wright. *Numerical optimization*. Springer, 2006. 4

[38] A. V. Oppenheim and R. W. Schafer. *Discrete-Time Signal Processing: Pearson New International Edition*. Pearson Higher Ed, 2010. 4

[39] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: bringing order to the web. 1999. 1, 5

[40] J.-Y. Pan, H.-J. Yang, C. Faloutsos, and P. Duygulu. Automatic multimedia cross-modal correlation discovery. In *International Conference on Knowledge Discovery and Data Mining*. ACM, 2004. 4, 5

[41] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, June 2007. 6

[42] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *CVPR*, June 2008. 6

[43] G. Puy, S. Kitic, and P. Pérez. Unifying local and non-local signal processing with graph cnns. *arXiv preprint arXiv:1702.07759*, 2017. 5

[44] D. Qin, S. Gammeter, L. Bossard, T. Quack, and L. Van Gool. Hello neighbor: Accurate object retrieval with k-reciprocal nearest neighbors. In *CVPR*, 2011. 1

[45] F. Radenović, G. Tolias, and O. Chum. CNN image retrieval learns from bow: Unsupervised fine-tuning with hard examples. *ECCV*, 2016. 1, 2, 6, 8

[46] A. S. Razavian, J. Sullivan, S. Carlsson, and A. Maki. Visual instance retrieval with deep convolutional networks. *ITE Transactions on Media Technology and Applications*, 4:251–258, 2016. 1, 8

[47] V. Rokhlin, A. Szlam, and M. Tygert. A randomized algorithm for principal component analysis. *SIAM Journal on Matrix Analysis and Applications*, 31(3):1100–1124, 2009. 2

[48] S. Roux, N. Tremblay, P. Borgnat, P. Abry, H. Wendt, and P. Messier. Multiscale anisotropic texture unsupervised clustering for photographic paper. In *IEEE International Workshop on Information Forensics and Security*, pages 1–6, 2015. 6

[49] H. Rue and L. Held. *Gaussian Markov random fields: theory and applications*. CRC Press, 2005. 4

[50] A. Sandryhaila and J. M. Moura. Discrete signal processing on graphs. IEEE *Transactions on Signal Processing*, 61(7):1644–1656, 2013. 1, 4, 5

[51] J. R. Seeley. The net of reciprocal influence. a problem in treating sociometric data. *Canadian Journal of Experimental Psychology*, 3:234, 1949. 5

[52] J. Shawe-Taylor and N. Cristianini. *Kernel methods for pattern analysis*. Cambridge university press, 2004. 5

[53] X. Shen, Z. Lin, J. Brandt, and Y. Wu. Spatially-constrained similarity measure for large-scale object retrieval. *IEEE Trans. PAMI*, 36(6):1229–1241, 2014. 1

[54] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98, 2013. 4, 5

[55] D. I. Shuman, P. Vandergheynst, and P. Frossard. Chebyshev polynomial approximation for distributed signal processing. In *International Conference on Distributed Computing in Sensor Systems and Workshops*, pages 1–8. IEEE, 2011. 5

[56] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2014. 6

[57] A. J. Smola and R. Kondor. Kernels and regularization on graphs. In *Learning Theory and Kernel Machines*, pages 144–158. Springer, 2003. 5

[58] A. J. Smola, B. Scholkopf, and K.-R. Muller. The connection between regularization operators and support vector kernels. *Neural Networks*, 11(4):637–649, 1998. 5

[59] M. F. Tappen, C. Liu, E. H. Adelson, and W. T. Freeman. Learning Gaussian conditional random fields for low-level vision. In *CVPR*, pages 1–8. IEEE, 2007. 5

[60] G. Tolias, R. Sicre, and H. Jégou. Particular object retrieval with integral max-pooling of cnn activations. *ICLR*, 2016. 1, 6

[61] H. Tong, C. Faloutsos, and J. Y. Pan. Fast random walk with restart and its applications. In *Proceedings of the IEEE International Conference on Data Mining*, pages 613–622, 2006. 4, 5

[62] L. N. Trefethen and D. Bau III. *Numerical linear algebra*. SIAM, 1997. 2

[63] N. Tremblay and P. Borgnat. Graph wavelets for multiscale community mining. IEEE *Transactions on Signal Processing*, 62(20):5227–5239, 2014. 6

[64] S. Vigna. Spectral ranking. *arXiv preprint arXiv:0912.0238*, 2009. 4, 5

[65] S. V. N. Vishwanathan, N. N. Schraudolph, R. Kondor, and K. M. Borgwardt. Graph kernels. *Journal of Machine Learning Research*, 11(Apr):1201–1242, 2010. 5

[66] M. Wainwright and M. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 649, 2008. 4

[67] S. Wang and S. Jiang. Instre: a new benchmark for instance-level object retrieval and recognition. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 11:37, 2015. 1, 6

[68] R. Witten and E. Candes. Randomized algorithms for low-rank matrix factorizations: Sharp performance bounds. *arXiv preprint arXiv:1308.5697*, 2013. 2

[69] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *NIPS*, 2003. 4, 5

[70] D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Schölkopf. Ranking on data manifolds. In *NIPS*, 2003. 1, 4, 5

[71] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In *ICML*, 2003. 5

[72] X. Zhu, J. Kandola, J. Lafferty, and Z. Ghahramani. Graph kernels by spectral transforms. *Semi-Supervised Learning*, pages 277–291, 2006. 5

[73] X. Zhu, J. D. Lafferty, and Z. Ghahramani. Semi-supervised learning: From Gaussian fields to Gaussian processes. Technical report, 2003. 5

XIX Hybrid Diffusion: Spectral-Temporal Graph Filtering for Manifold Ranking

Title: Hybrid Diffusion: Spectral-Temporal Graph Filtering for Manifold Ranking
Authors: A. Iscen, Y. Avrithis, G. Tolias, T. Furon, O. Chum
Published at: ACCV 2018

Hybrid Diffusion: Spectral-Temporal Graph Filtering for Manifold Ranking

Ahmet Iscen[1] Yannis Avrithis[2] Giorgos Tolias[1]
Teddy Furon[2] Ondřej Chum[1]

[1]VRG, FEE, CTU in Prague [2]Univ Rennes, Inria, CNRS, IRISA

Abstract. State of the art image retrieval performance is achieved with CNN features and manifold ranking using a k-NN similarity graph that is pre-computed off-line. The two most successful existing approaches are *temporal filtering*, where manifold ranking amounts to solving a sparse linear system online, and *spectral filtering*, where eigen-decomposition of the adjacency matrix is performed off-line and then manifold ranking amounts to dot-product search online. The former suffers from expensive queries and the latter from significant space overhead. Here we introduce a novel, theoretically well-founded *hybrid filtering* approach allowing full control of the space-time trade-off between these two extremes. Experimentally, we verify that our hybrid method delivers results on par with the state of the art, with lower memory demands compared to spectral filtering approaches and faster compared to temporal filtering.

1 Introduction

Most image retrieval methods obtain their initial ranking of the database images by computing similarity between the query descriptor and descriptors of the database images. Descriptors based on local features [22, 16] have been largely replaced by more efficient CNN-based image descriptors [9, 20]. Regardless of the initial ranking, the retrieval performance is commonly boosted by considering the manifold structure of the database descriptors, rather than just independent distances of query to database images. Examples are query expansion [5, 1] and diffusion [8, 12, 11]. *Query expansion* uses the results of initial ranking to issue a novel, enriched, query [5] on-line only. *Diffusion* on the other hand, is based on the k-NN graph of the dataset that is constructed off-line, so that, assuming novel queries are part of the dataset, their results are essentially pre-computed. Diffusion can then be seen as infinite-order query expansion [12].

The significance of the performance boost achieved by diffusion has been recently demonstrated at the "Large-Scale Landmark Recognition"[1] challenge in conjunction with CVPR 2018. The vast majority of top-ranked teams have used query expansion or diffusion as the last step of their method.

Recently, efficient diffusion methods have been introduced to the image retrieval community. Iscen *et al.* [12] apply diffusion to obtain the final ranking,

[1] https://landmarkscvprw18.github.io/

in particular by solving a large and sparse system of linear equations. Even though an efficient *conjugate gradient* (CG) [10] solver is used, query times on large-scale datasets are in a range of several seconds. A significant speed-up is achieved by *truncating* the system of linear equations. Such an approximation, however, brings a slight degradation in the retrieval performance. Their method can be interpreted as graph filtering in the *temporal* domain.

In the recent work of Iscen *et al*. [11], more computation is shifted to the off-line phase to accelerate the query. The solution of the linear system is estimated by low-rank approximation of the k-NN graph Laplacian. Since the eigenvectors of the Laplacian represent a Fourier basis of the graph, this is interpreted as graph filtering in the *spectral* domain. The price to pay is increased space complexity to store the embeddings of the database descriptors. For comparable performance, a 5k-10k dimensional vector is needed per image.

In this paper, we introduce a *hybrid* method that combines spectral filtering [11] and temporal filtering [12]. This hybrid method offers a trade-off between speed (*i.e*., the number of iterations of CG) and the additional memory required (*i.e*., the dimensionality of the embedding). The two approaches [12, 11] are extreme cases of our hybrid method. We show that the proposed method pairs or outperforms the previous methods while either requiring less memory or being significantly faster – only three to five iterations of CG are necessary for embeddings of 100 to 500 dimensions.

While both temporal and spectral filtering approaches were known in other scientific fields before being successfully applied to image retrieval, to our knowledge the proposed method is novel and can be applied to other domains.

The rest of the paper is organized as follows. Related work is reviewed in Section 2. Previous work on temporal and spectral filtering is detailed in Sections 3.1 and 3.2 respectively, since the paper builds on this work. The proposed method is described in Section 4 and its behavior is analyzed in Section 5. Experimental results are provided in Section 6. Conclusions are drawn in Section 7.

2 Related work

Query expansion (QE) has been a standard way to improve recall of image retrieval since the work of Chum *et al*. [5]. A variety of approaches exploit local feature matching and perform various types of verification. Such matching ranges from selective kernel matching [23] to geometric consensus [5, 4, 14] with RANSAC-like techniques. The verified images are then used to refine the global or local image representation of a novel query.

Another family of QE methods are more generic and simply assume a global image descriptor [21, 13, 7, 18, 27, 8, 1]. A simple and popular one is average-QE [5], recently extended to α-QE [20]. At some small extra cost, recall is significantly boosted. This additional cost is restricted to the on-line query phase. This is in contrast to another family of approaches that considers an off-line preprocessing of the database. Given the nearest neighbors list for database images, QE is performed by adapting the local similarity measure [13], using reciprocity

constraints [7, 18] or graph-based similarity propagation [27, 8, 12]. The graph-based approaches, also known as diffusion, are shown to achieve great performance [12] and to be a good way for feature fusion [27]. Such on-line re-ranking is typically orders of magnitude more costly than simple average-QE.

The advent of CNN-based features, especially global image descriptors, made QE even more attractive. Average-QE or α-QE are easily applicable and very effective with a variety of CNN-based descriptors [20, 9, 24, 15]. State-of-the-art performance is achieved with diffusion on global or regional descriptors [12]. The latter is possible due to the small number of regions that are adequate to represent small objects, in contrast to thousands in the case of local features.

Diffusion based on tensor products can be attractive in terms of performance [2, 3]. However, in this work, we focus on the page-rank like diffusion [12, 28] due to its reasonable query times. An iterative on-line solution was commonly preferred [8] until the work of Iscen *et al*. [12], who solve a linear system to speed up the process. Additional off-line pre-processing and the construction and storage of additional embeddings reduce diffusion to inner product search in the spectral ranking of Iscen *et al*. [11]. This work lies exactly in between these two worlds and offers a trade-off exchanging memory for speed and vice versa.

Fast random walk with restart [25] is very relevant in the sense that it follows the same diffusion model as [12, 28] and is a hybrid method like ours. It first disconnects the graph into distinct components through clustering and then obtains a low-rank spectral approximation of the residual error. Apart from the additional complexity, parameters *etc*. of the off-line clustering process and the storage of both eigenvectors and a large inverse matrix, its online phase is also complex, involving the Woodbury matrix identity and several dense matrix-vector multiplications. Compared to that, we first obtain a *very* low-rank spectral approximation of the original graph, and then solve a sparse linear system of the residual error. Thanks to orthogonality properties, the online phase is nearly as simple as the original one and significantly faster.

3 Problem formulation and background

The methods we consider are based on a nearest neighbor graph of a dataset of n items, represented by $n \times n$ *adjacency matrix* W. The graph is undirected and weighted according to similarity: W is sparse, symmetric, nonnegative and zero-diagonal. We symmetrically normalize W as $\mathcal{W} := D^{-1/2} W D^{-1/2}$, where $D := \mathrm{diag}(W\mathbf{1})$ is the diagonal *degree matrix*, containing the row-wise sum of W on its diagonal. The eigenvalues of $\mathcal{W}$ lie in $[-1, 1]$ [6].

At query time, we are given a sparse $n \times 1$ *observation vector* $\mathbf{y}$, which is constructed by searching for the nearest neighbors of a query item in the dataset and setting its nonzero entries to the corresponding similarities. The problem is to obtain an $n \times 1$ *ranking vector* $\mathbf{x}$ such that retrieved items of the dataset are ranked by decreasing order of the elements of $\mathbf{x}$. Vector $\mathbf{x}$ should be close to $\mathbf{y}$ but at the same time similar items are encouraged to have similar ranks in $\mathbf{x}$, essentially by exploring the graph to retrieve more items.

3.1 Temporal filtering

Given a parameter $\alpha \in [0, 1)$, define the $n \times n$ *regularized Laplacian* function by

$$\mathcal{L}_\alpha(A) := (I_n - \alpha A)/(1 - \alpha) \tag{1}$$

for $n \times n$ real symmetric matrix A, where I_n is the $n \times n$ identity matrix. Iscen *et al*. [12] then define $\mathbf{x}$ as the unique solution of the linear system

$$\mathcal{L}_\alpha(\mathcal{W})\mathbf{x} = \mathbf{y}, \tag{2}$$

which they obtain approximately in practice by a few iterations of the *conjugate gradient* (CG) method, since $\mathcal{L}_\alpha(\mathcal{W})$ is positive-definite. At large scale, they truncate $\mathcal{W}$ by keeping only the rows and columns corresponding to a fixed number of neighbors of the query, and re-normalize. Then, (2) only performs re-ranking within this neighbor set.

We call this method *temporal*[2] *filtering* because if $\mathbf{x}$, $\mathbf{y}$ are seen as signals, then $\mathbf{x}$ is the result of applying a linear graph filter on $\mathbf{y}$, and CG iteratively applies a set of recurrence relations that determine the filter. While $\mathcal{W}$ is computed and stored off-line, (2) is solved online (at query time), and this is expensive.

3.2 Spectral filtering

Linear system (2) can be written as $\mathbf{x} = h_\alpha(\mathcal{W})\mathbf{y}$, where the *transfer function* h_α is defined by

$$h_\alpha(A) := \mathcal{L}_\alpha(A)^{-1} = (1 - \alpha)(I_n - \alpha A)^{-1} \tag{3}$$

for $n \times n$ real symmetric matrix A. Given the eigenvalue decomposition $\mathcal{W} = U\Lambda U^\top$ of the symmetric matrix $\mathcal{W}$, Iscen *et al*. [11] observe that $h_\alpha(\mathcal{W}) = U h_\alpha(\Lambda) U^\top$, so that (2) can be written as

$$\mathbf{x} = U h_\alpha(\Lambda) U^\top \mathbf{y}, \tag{4}$$

which they approximate by keeping only the largest r eigenvalues and the corresponding eigenvectors of $\mathcal{W}$. This defines a low-rank approximation $h_\alpha(\mathcal{W}) \approx U_1 h_\alpha(\Lambda_1) U_1^\top$ instead. This method is referred to as *fast spectral ranking* (FSR) in [11]. Crucially, Λ is a diagonal matrix, hence h_α applies element-wise, as a scalar function $h_\alpha(x) := (1 - \alpha)/(1 - \alpha x)$ for $x \in [-1, 1]$.

At the expense of off-line computing and storing the $n \times r$ matrix U_1 and the r eigenvalues in Λ_1, filtering is now computed as a sequence of low-rank matrix-vector multiplications, and the query is accelerated by orders of magnitude compared to [12]. However, the space overhead is significant. To deal with approximation errors when r is small, a heuristic is introduced that gradually falls back to the similarity in the original space for items that are poorly represented, referred to as FSRw [11].

[2] "Temporal" stems from conventional signal processing where signals are functions of "time"; while "spectral" is standardized also in graph signal processing.

We call this method *spectral filtering* because U represents the Fourier basis of the graph and the sequence of matrix-vector multiplications from right to left in the right-hand side of (4) represents the Fourier transform of $\mathbf{y}$ by $U^\top$, filtering in the frequency domain by $h_\alpha(\Lambda)$, and finally the inverse Fourier transform to obtain $\mathbf{x}$ in the time domain by U.

4 Hybrid spectral-temporal filtering

Temporal filtering (2) is performed once for every new query represented by $\mathbf{y}$, but $\mathcal{W}$ represents the dataset and is fixed. Could CG be accelerated if we had some very limited additional information on $\mathcal{W}$?

On the other extreme, spectral filtering (4) needs a large number of eigenvectors and eigenvalues of $\mathcal{W}$ to provide a high quality approximation, but always leaves some error. Could we reduce this space requirement by allocating some additional query time to recover the approximation error?

The answer is positive to both questions and in fact these are the two extreme cases of *hybrid spectral-temporal filtering*, which we formulate next.

4.1 Derivation

We begin with the eigenvalue decomposition $\mathcal{W} = U\Lambda U^\top$, which we partition as $\Lambda = \operatorname{diag}(\Lambda_1, \Lambda_2)$ and $U = (U_1\ U_2)$. Matrices Λ_1 and Λ_2 are diagonal $r \times r$ and $(n-r) \times (n-r)$, respectively. Matrices U_1 and U_2 are $n \times r$ and $n \times (n-r)$, respectively, and have the following orthogonality properties, all due to the orthogonality of U itself:

$$U_1^\top U_1 = I_r, \quad U_2^\top U_2 = I_{n-r}, \quad U_1^\top U_2 = \mathbf{O}, \quad U_1 U_1^\top + U_2 U_2^\top = I_n. \tag{5}$$

Then, $\mathcal{W}$ is decomposed as

$$\mathcal{W} = U_1 \Lambda_1 U_1^\top + U_2 \Lambda_2 U_2^\top. \tag{6}$$

Similarly, $h_\alpha(\mathcal{W})$ is decomposed as

$$\begin{aligned} h_\alpha(\mathcal{W}) &= U h_\alpha(\Lambda) U^\top && (7)\\ &= U_1 h_\alpha(\Lambda_1) U_1^\top + U_2 h_\alpha(\Lambda_2) U_2^\top, && (8) \end{aligned}$$

which is due to the fact that diagonal matrix $h_\alpha(\Lambda)$ is obtained element-wise, hence decomposed as $h_\alpha(\Lambda) = \operatorname{diag}(h_\alpha(\Lambda_1), h_\alpha(\Lambda_2))$. Here the first term is exactly the low-rank approximation that is used by spectral filtering, and the second is the approximation error

$$\begin{aligned} e_\alpha(\mathcal{W}) :&= U_2 h_\alpha(\Lambda_2) U_2^\top && (9)\\ &= (1-\alpha)\left(U_2 (I_{n-r} - \alpha\Lambda_2)^{-1} U_2^\top + U_1 U_1^\top - U_1 U_1^\top\right) && (10)\\ &= (1-\alpha)\left(\left(U_2 (I_{n-r} - \alpha\Lambda_2) U_2^\top + U_1 U_1^\top\right)^{-1} - U_1 U_1^\top\right) && (11)\\ &= (1-\alpha)\left(\left(I_n - \alpha U_2 \Lambda_2 U_2^\top\right)^{-1} - U_1 U_1^\top\right) && (12)\\ &= h_\alpha(U_2 \Lambda_2 U_2^\top) - (1-\alpha) U_1 U_1^\top. && (13) \end{aligned}$$

We have used the definition (3) of h_α in (10) and (13). Equation (12) is due to the orthogonality properties (5). Equation (11) follows from the fact that for any invertible matrices A, B of conformable sizes,

$$(U_1 A U_1 + U_2 B U_2)^{-1} = U_1 A^{-1} U_1 + U_2 B^{-1} U_2, \tag{14}$$

which can be verified by direct multiplication, and is also due to orthogonality.

Now, combining (8), (13) and (6), we have proved the following.

Theorem 1. *Assuming the definition (3) of transfer function h_α and the eigenvalue decomposition (6) of the symmetrically normalized adjacency matrix $\mathcal{W}$, $h_\alpha(\mathcal{W})$ is decomposed as*

$$h_\alpha(\mathcal{W}) = U_1 g_\alpha(\Lambda_1) U_1^\top + h_\alpha(\mathcal{W} - U_1 \Lambda_1 U_1^\top), \tag{15}$$

where

$$g_\alpha(A) := h_\alpha(A) - h_\alpha(\mathbf{O}) = (1-\alpha)\left((I_n - \alpha A)^{-1} - I_n\right) \tag{16}$$

for $n \times n$ real symmetric matrix A. For $x \in [-1, 1]$ in particular, $g_\alpha(x) := h_\alpha(x) - h_\alpha(0) = (1-\alpha)\alpha x/(1-\alpha x)$.

Observe that Λ_2, U_2 do not appear in (15) and indeed it is only the largest r eigenvalues Λ_1 and corresponding eigenvectors U_1 of $\mathcal{W}$ that we need to compute. The above derivation is generalized from h_α to a much larger class of functions in appendix A.

4.2 Algorithm

Why is decomposition (15) of $h_\alpha(\mathcal{W})$ important? Because given an observation vector $\mathbf{y}$ at query time, we can express the ranking vector $\mathbf{x}$ as

$$\mathbf{x} = \mathbf{x}^s + \mathbf{x}^t, \tag{17}$$

where the first, *spectral*, term $\mathbf{x}^s$ is obtained by spectral filtering

$$\mathbf{x}^s = U_1 g_\alpha(\Lambda_1) U_1^\top \mathbf{y}, \tag{18}$$

as in [11], where g_α applies element-wise, while the second, *temporal*, term $\mathbf{x}^t$ is obtained by temporal filtering, that is, solving the linear system

$$\mathcal{L}_\alpha(\mathcal{W} - U_1 \Lambda_1 U_1^\top)\mathbf{x}^t = \mathbf{y}, \tag{19}$$

which we do by a few iterations of CG as in [12]. The latter is possible because $\mathcal{L}_\alpha(\mathcal{W} - U_1 \Lambda_1 U_1^\top)$ is still positive-definite, like $\mathcal{L}_\alpha(\mathcal{W})$. It's also possible without an explicit dense representation of $U_1 \Lambda_1 U_1^\top$ because CG, like all Krylov subspace methods, only needs *black-box* access to the matrix A of the linear system, that is, a mapping $\mathbf{z} \mapsto A\mathbf{z}$ for $\mathbf{z} \in \mathbb{R}^n$. For system (19) in particular, according to the definition (1) of $\mathcal{L}_\alpha$, we use the mapping

$$\mathbf{z} \mapsto \left(\mathbf{z} - \alpha\left(\mathcal{W}\mathbf{z} - U_1 \Lambda_1 U_1^\top \mathbf{z}\right)\right)/(1-\alpha), \tag{20}$$

where product $\mathcal{W}\mathbf{z}$ is efficient because $\mathcal{W}$ is sparse as in [12], while $U_1 \Lambda_1 U_1^\top \mathbf{z}$ is efficient if computed right-to-left because U_1 is an $n \times r$ matrix with $r \ll n$ and Λ_1 is diagonal as in [11].

4.3 Discussion

What is there to gain from spectral-temporal decomposition (17) of $\mathbf{x}$*?*

First, since the temporal term (19) can recover the spectral approximation error, the rank r of U_1, Λ_1 in the spectral term (18) can be chosen as small as we like. In the extreme case $r = 0$, the spectral term vanishes and we recover temporal filtering [12]. This allows efficient computation of only a few eigenvectors/values, even with the Lanczos algorithm rather than the approximate method of [11]. Most importantly, it reduces significantly the space complexity, at the expense of query time. Like spectral filtering, it is possible to *sparsify* U_1 to compress the dataset embeddings and accelerate the queries online. In fact, we show in section 6 that sparsification is much more efficient in our case.

Second, the matrix $\mathcal{W} - U_1\Lambda_1 U_1^\top$ is effectively like $\mathcal{W}$ with the r largest eigenvalues removed. This improves significantly the condition number of matrix $\mathcal{L}_\alpha(\mathcal{W} - U_1\Lambda_1 U_1^\top)$ in the temporal term (19) compared to $\mathcal{L}_\alpha(\mathcal{W})$ in the linear system (2) of temporal filtering [12], on which the convergence rate depends. In the extreme case $r = n$, the temporal term vanishes and we recover spectral filtering [11]. In turn, even with small r, this reduces significantly the number of iterations needed for a given accuracy, at the expense of computing and storing U_1, Λ_1 off-line as in [11]. The improvement is a function of α and the spectrum of $\mathcal{W}$, and is quantified in section 5.

In summary, for a given desired accuracy, we can choose the rank r of the spectral term and a corresponding number of iterations of the temporal term, determining a trade-off between the space needed for the eigenvectors (and the off-line cost to obtain them) and the (online) query time. Such choice is not possible with either spectral or temporal filtering alone: at large scale, the former may need too much space and the latter may be too slow.

5 Analysis

How "easier" for CG is $\mathcal{W} - U_1\Lambda_1 U_1^\top$ *in (19) compared to* $\mathcal{W}$ *in (2)?*

In solving a linear system $A\mathbf{x} = \mathbf{b}$ where A is an $n \times n$ symmetric positive-definite matrix, CG generates a unique sequence of iterates $\mathbf{x}_i$, $i = 0, 1, \ldots$, such that the A-norm $\|\mathbf{e}_i\|_A$ of the error $\mathbf{e}_i := \mathbf{x}^* - \mathbf{x}_i$ is minimized over the Krylov subspace $\mathcal{K}_i := \langle \mathbf{b}, A\mathbf{b}, \ldots, A^i\mathbf{b} \rangle$ at each iteration i, where $\mathbf{x}^* := A^{-1}\mathbf{b}$ is the exact solution and the A-norm is defined by $\|\mathbf{x}\|_A := \sqrt{\mathbf{x}^\top A\mathbf{x}}$ for $\mathbf{x} \in \mathbb{R}^n$.

A well-known result on the rate of convergence of CG that assumes minimal knowledge of the eigenvalues of A states that the A-norm of the error at iteration i, relative to the A-norm of the initial error $\mathbf{e}_0 := \mathbf{x}^*$, is upper-bounded by [26]

$$\frac{\|\mathbf{e}_i\|_A}{\|\mathbf{e}_0\|_A} \leq \phi_i(A) := 2\left(\frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1}\right)^i, \tag{21}$$

where $\kappa(A) := \|A\| \left\|A^{-1}\right\| = \lambda_1(A)/\lambda_n(A)$ is the 2-norm *condition number* of A, and $\lambda_j(A)$ for $j = 1, \ldots, n$ are the eigenvalues of A in descending order.

In our case, matrix $\mathcal{L}_\alpha(\mathcal{W})$ of linear system (2) has condition number

$$\kappa(\mathcal{L}_\alpha(\mathcal{W})) = \frac{1-\alpha\lambda_n(\mathcal{W})}{1-\alpha\lambda_1(\mathcal{W})} = \frac{1-\alpha\lambda_n(\mathcal{W})}{1-\alpha}. \tag{22}$$

The first equality holds because for each eigenvalue λ of $\mathcal{W}$ there is a corresponding eigenvalue $(1-\alpha\lambda)/(1-\alpha)$ of $\mathcal{L}_\alpha(\mathcal{W})$, which is a decreasing function. The second holds because $\lambda_1(\mathcal{W}) = 1$ [6].

Now, let $\mathcal{W}_r := \mathcal{W} - U_1\Lambda_1 U_1^\top$ for $r = 0, 1, \ldots, n-1$, where Λ_1, U_1 represent the largest r eigenvalues and the corresponding eigenvectors of $\mathcal{W}$ respectively. Clearly, $\mathcal{W}_r$ has the same eigenvalues as $\mathcal{W}$ except for the largest r, which are replaced by zero. That is, $\lambda_1(\mathcal{W}_r) = \lambda_{r+1}(\mathcal{W})$ and $\lambda_n(\mathcal{W}_r) = \lambda_n(\mathcal{W})$. The latter is due to the fact that $\lambda_n(\mathcal{W}) \leq -1/(n-1) \leq 0$ [6], so the new zero eigenvalues do not affect the smallest one. Then,

$$\kappa(\mathcal{L}_\alpha(\mathcal{W}_r)) = \frac{1-\alpha\lambda_n(\mathcal{W})}{1-\alpha\lambda_{r+1}(\mathcal{W})} \leq \kappa(\mathcal{L}_\alpha(\mathcal{W})). \tag{23}$$

This last expression generalizes (22). Indeed, $\mathcal{W} = \mathcal{W}_0$. Then, our hybrid spectral-temporal filtering involves CG on $\mathcal{L}_\alpha(\mathcal{W}_r)$ for $r \geq 0$, compared to the baseline temporal filtering for $r = 0$. The inequality in (23) is due to the fact that $|\lambda_j(\mathcal{W})| \leq 1$ for $j = 1, \ldots, n$ [6]. Removing the largest r eigenvalues of $\mathcal{W}$ clearly improves (decreases) the condition number of $\mathcal{L}_\alpha(\mathcal{W}_r)$ relative to $\mathcal{L}_\alpha(\mathcal{W})$. The improvement is dramatic given that α is close to 1 in practice. For $\alpha = 0.99$ and $\lambda_{r+1}(\mathcal{W}) = 0.7$ for instance, $\kappa(\mathcal{L}_\alpha(\mathcal{W}_r))/\kappa(\mathcal{L}_\alpha(\mathcal{W})) = 0.0326$.

More generally, given the eigenvalues $\lambda_{r+1}(\mathcal{W})$ and $\lambda_n(\mathcal{W})$, the improvement can be estimated by measuring the upper bound $\phi_i(\mathcal{L}_\alpha(\mathcal{W}_r))$ for different i and r. A concrete example is shown in Figure 1, where we measure the eigenvalues of

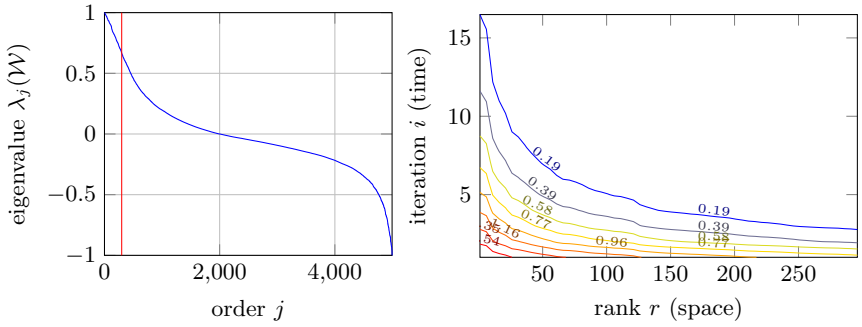


Fig. 1. (a) In descending order, eigenvalues of adjacency matrix $\mathcal{W}$ of Oxford5k dataset of $n = 5,063$ images with global GeM features by ResNet101 and $k = 50$ neighbors per point (see section 6). Eigenvalues on the left of vertical red line at $j = 300$ are the largest 300 ones, candidate for removal. (b) Contour plot of upper bound $\phi_i(\mathcal{L}_\alpha(\mathcal{W}_r))$ of CG's relative error as a function of rank r and iteration i for $\alpha = 0.99$, illustrating the space(r)-time(i) trade-off for constant relative error.

the adjacency matrix $\mathcal{W}$ of a real dataset, remove the largest r for $0 \leq r \leq 300$ and plot the upper bound $\phi_i(\mathcal{L}_\alpha(\mathcal{W}_r))$ of the relative error as a function of rank r and iteration i given by (21) and (23). Clearly, as more eigenvalues are removed, less CG iterations are needed to achieve the same relative error; the approximation error represented by the temporal term decreases and at the same time the linear system becomes easier to solve. Of course, iterations become more expensive as r increases; precise timings are given in section 6.

6 Experiments

In this section we evaluate our hybrid method on popular image retrieval benchmarks. We provide comparisons to baseline methods, analyze the trade-off between runtime complexity, memory footprint and search accuracy, and compare with the state of the art.

6.1 Experimental setup

Datasets. We use the revisited retrieval benchmark [19] of the popular Oxford buildings [16] and Paris [17] datasets, referred to as $\mathcal{R}$Oxford and $\mathcal{R}$Paris, respectively. Unless otherwise specified, we evaluate using the *Medium* setup and always report mean Average Precision (mAP). Large-scale experiments are conducted on $\mathcal{R}$Oxford $+\mathcal{R}$1M and $\mathcal{R}$Paris $+\mathcal{R}$1M by adding the new 1M challenging distractor set [19].

Image representation. We use GeM descriptors [20] to represent images. We extract GeM at 3 different image scales, aggregate the 3 descriptors, and perform whitening, exactly as in [20]. Finally, each image is represented by a single vector with $d = 2048$ dimensions, since ResNet-101 architecture is used.

Baseline methods. We consider the two baseline methods described in Section 3, namely temporal and spectral filtering. *Temporal filtering* corresponds to solving a linear system with CG [12] and is evaluated for different numbers of CG iterations. It is used with truncation at large scale to speed up the search [12] and is denoted by *Temporal*†. *Spectral filtering* corresponds to FSR and its FSRw variant [11]. Both FSR variants are parametrized by the rank r of the approximation, which is equal to the dimensionality of the spectral embedding.

Implementation details. Temporal ranking is performed with the implementation[3] provided by Iscen *et al.* [12]. The adjacency matrix is constructed by using top $k = 50$ reciprocal neighbors. Pairwise similarity between descriptors $\mathbf{v}$ and $\mathbf{z}$ is estimated by $(\mathbf{v}^\top \mathbf{z})_+^3$. Parameter α is set to 0.99, while the observation vector $\mathbf{y}$ includes the top 5 neighbors. The eigendecomposition is performed on the largest connected component, as in [11]. Its size is 933,412 and 934,809 for $\mathcal{R}$Oxford $+\mathcal{R}$1M and $\mathcal{R}$Paris $+\mathcal{R}$1M, respectively. Timings are measured with Matlab implementation on a 4-core Intel Xeon 2.60GHz CPU with 200 GB of RAM. We only report timings for the diffusion part of the ranking and exclude the initial nearest neighbor search used to construct the observation vector.

[3] https://github.com/ahmetius/diffusion-retrieval/

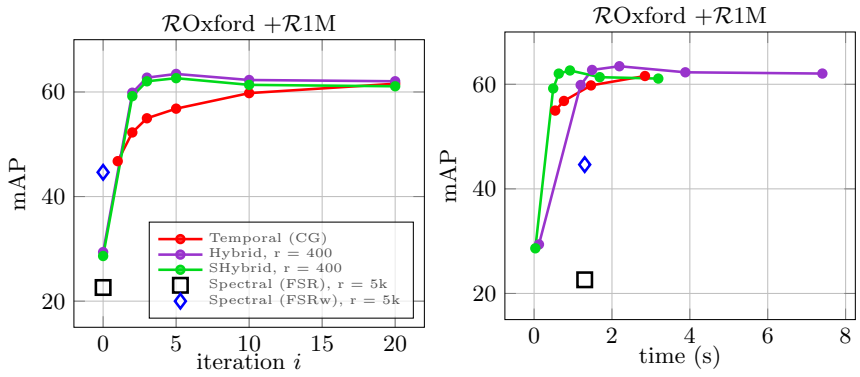


Fig. 2. mAP *vs*. CG iteration i and mAP *vs*. time for temporal, spectral, and hybrid filtering. Sparsified hybrid is used with sparsity 99%.

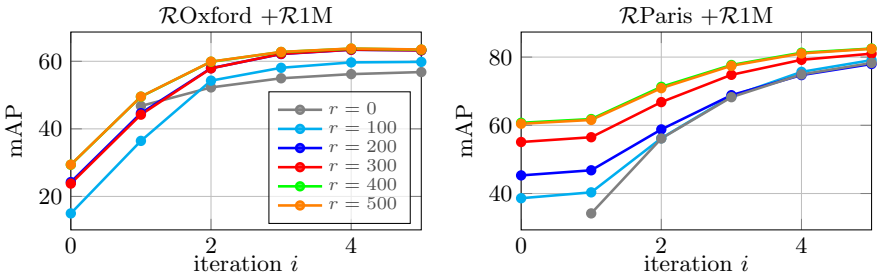


Fig. 3. mAP *vs*. CG iteration i for different rank r for our hybrid method, where $r = 0$ means temporal only.

6.2 Comparisons

Comparison with baseline methods. We first compare performance, query time and required memory for temporal, spectral, and hybrid ranking. With respect to the memory, all methods store the initial descriptors, *i.e*. one 2048-dimensional vector per image. Temporal ranking additionally stores the sparse regularized Laplacian. Spectral ranking stores for each vector an additional embedding of dimensionality equal to rank r, which is a parameter of the method. Our hybrid method stores both the Laplacian and the embedding, but with significantly lower rank r.

We evaluate on $\mathcal{R}$Oxford +$\mathcal{R}$1M and $\mathcal{R}$Paris +$\mathcal{R}$1M with global image descriptors, which is a challenging large scale problem, *i.e*. large adjacency matrix, where prior methods fail or have serious drawbacks. Results are shown in Figure 2. Temporal ranking with CG is reaching saturation near 20 iterations as in [12]. Spectral ranking is evaluated for a decomposition of rank r whose computation and required memory are reasonable and feasible on a single machine. Finally, the proposed hybrid method is evaluated for rank $r = 400$, which is a good compromise of speed and memory, as shown below.

Spectral ranking variants (FSR, FSRw) are not performing well despite requiring about 250% additional memory compared to nearest neighbor search in the original space. Compared to hybrid, more than one order of magnitude higher rank r is required for problems of this scale. Temporal ranking achieves good performance but at much more iterations and higher query times. Our hybrid solution provides a very reasonable space-time trade-off.

Runtime and memory trade-off. We report the trade-off between number of iterations and rank r, representing additional memory, in more detail in Figure 3. It is shown that the number of iterations to achieve the top mAP decreases as the rank increases. We achieve the optimal trade-off at $r = 400$ where we only need 5 or less iterations. Note that, the rank not only affects the memory and the query time of the spectral part in a linear manner, but the query time of the temporal part too (20).

Sparsification of spectral embeddings is exploited in prior work [11]. We sparsify the embeddings of our hybrid method by setting the smallest values of U_1 to zero until a desired level of sparsity is reached. We denote this method by *SHybrid*. This sparse variant provides memory savings and an additional speedup due to the computations with sparse matrices. Figure 4 shows that performance loss remains small even for extreme sparsity *e.g.* 99%, while the results in Figure 2 show that it offers a significant speedup in the global descriptor setup.

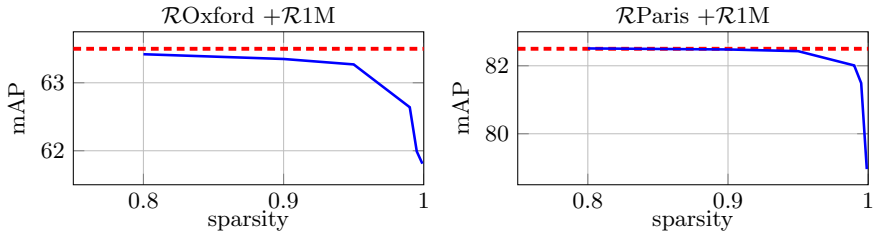


Fig. 4. mAP *vs.* level of sparsification on our hybrid method for $r = 400$. Dashed horizontal line indicates no sparsification.

Performance-memory-speed comparison with the baselines is shown in Table 1. Our hybrid approach enjoys query times lower than those of temporal with truncation or spectral with FSRw, while at the same time achieves higher performance and requires less memory than the spectral-only approach.

We summarize our achievement in terms of mAP, required memory, and query time in Figure 5. Temporal ranking achieves high performance at the cost of high query time and its truncated counterpart saves query time but sacrifices performance. Spectral ranking is not effective at this scale, while our hybrid solution achieves high performance at low query times.

Comparison with the state of the art. We present an extensive comparison with existing methods in the literature for global descriptors at small and large

	Temporal [12]	Temporal† [12]	Spectral (FSRw) [11]	SHybrid
mAP	61.6	59.0	42.1	62.6
Time (s)	2.8	1.0	1.3	0.9
Memory (MB)	205	205	35,606	264

Table 1. Performance, memory and query time comparison on $\mathcal{R}$Oxford +$\mathcal{R}$1M with GeM descriptors for temporal (20 iterations), truncated temporal (20 iterations, 75k images in shortlist), spectral ($r = 5k$), and hybrid ranking ($r = 400$, 5 iterations). Hybrid ranking is sparsified by setting the 99% smallest values to 0. Reported memory excludes the initial descriptors requiring 8.2 GB. U_1 is stored with double precision.

	$\mathcal{R}$Oxford		$\mathcal{R}$Oxf +$\mathcal{R}$1M		$\mathcal{R}$Paris		$\mathcal{R}$Paris +$\mathcal{R}$1M	
	Medium	Hard	Medium	Hard	Medium	Hard	Medium	Hard
NN-search	64.7	38.5	45.2	19.9	77.2	56.3	52.3	24.7
α-QE [19]	67.2	40.8	49.0	24.2	80.7	61.8	58.0	31.0
Temporal [12]	69.9	40.4	61.6	33.2	88.9	78.5	85.0	71.6
Temporal† [12]	-	-	59.0	31.4	-	-	79.7	65.2
FSR [11]	70.4	42.0	22.6	5.8	88.6	77.9	66.6	40.2
FSRw [11]	70.7	42.2	42.1	18.8	88.7	78.0	77.4	59.7
SHybrid (ours)	70.5	40.3	62.6	34.4	88.7	78.1	82.0	66.8

Table 2. mAP comparison with existing methods in the literature on small and large scale datasets, using Medium and Hard setup of the revisited benchmark.

scale (1M distractors). We choose $r = 400$ and 5 iterations for our hybrid method, 20 iterations for temporal ranking, $r = 2k$ and $r = 5k$ for spectral ranking at small and large scale, respectively. Temporal ranking is also performed with truncation on a shortlist size of $75k$ images at large scale. The comparison is presented in Table 2. Our hybrid approach performs the best or second best right after the temporal one, while providing much smaller query times at a small amount of additional required memory.

7 Conclusions

In this work we have tested the two most successful manifold ranking methods of temporal filtering [12] and spectral filtering [11] on the very challenging new benchmark of Oxford and Paris datasets [19]. It is the first time that such methods are evaluated at the scale of one million images. *Spectral filtering*, with both its FSR and FSRw variants, fails at this scale, despite the significant space required for additional vector embeddings. It is possible that a higher rank would work, but it wouldn't be practical in terms of space. In terms of query time, *temporal filtering* is only practical with its truncated variant at this scale. It works pretty well in terms of performance, but the query time is still high.

Our new *hybrid filtering* method allows for the first time to strike a reasonable balance between the two extremes. Without truncation, it outperforms temporal

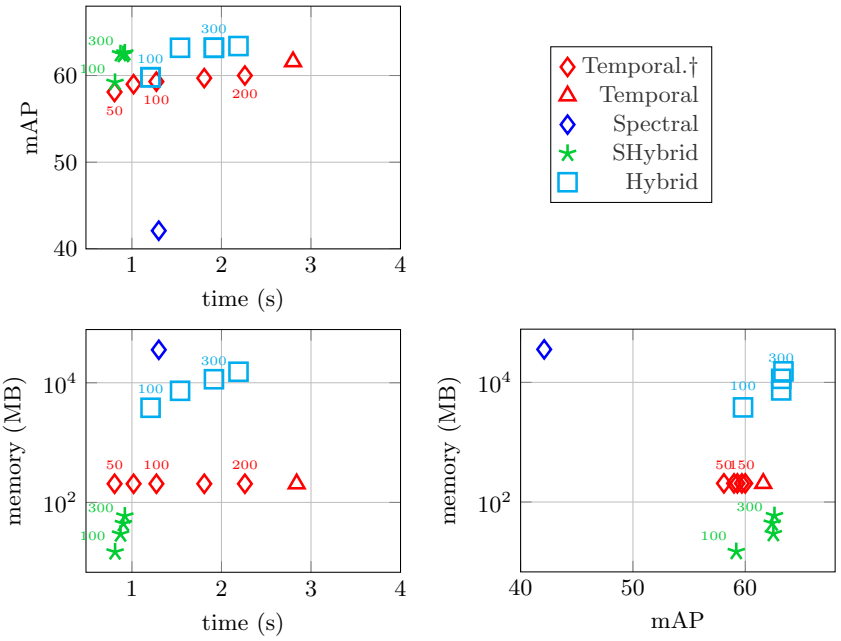


Fig. 5. Time (s) - memory (MB) - performance (mAP) for different methods. We show mAP *vs*. time, memory *vs*. time, and memory *vs*. mAP on $\mathcal{R}$Oxford +$\mathcal{R}$1M. Methods in the comparison: temporal for 20 iterations, truncated temporal for 20 iterations and shortlist of size 50k, 75k, 100k, 200k and 300k, spectral (FSRw) with $r = 5k$, hybrid with $r \in \{100, 200, 300, 400\}$ and 5 iterations, sparse hybrid with 99% sparsity, $r \in \{100, 200, 300, 400\}$ and 5 iterations. Text labels indicate the shortlist size (in thousands) for truncated temporal and rank for hybrid. Observe that the two plots on the left are aligned horizontally with respect to time, while the two at the bottom vertically with respect to memory.

filtering while being significantly faster, and its memory overhead is one order of magnitude less than that of spectral filtering. Unlike spectral filtering, it is possible to extremely sparsify the dataset embeddings with only negligible drop in performance. This, together with its very low rank, makes our hybrid method even faster than spectral, despite being iterative. More importantly, while previous methods were long known in other fields before being applied to image retrieval, to our knowledge our hybrid method is novel and can apply *e.g.* to any field where graph signal processing applies and beyond. Our theoretical analysis shows exactly why our method works and quantifies its space-time-accuracy trade-off using simple ideas from numerical linear algebra.

Acknowledgements: This work was supported by MSMT LL1303 ERC-CZ grant and the OP VVV funded project CZ.02.1.01/0.0/0.0/16_019/0000765 “Research Center for Informatics”.

A General derivation

The derivation of our algorithm in section 4.1 applies only to the particular function (filter) h_α (3). Here, as in [11], we generalize to a much larger class of functions, that is, any function h that has a series expansion

$$h(A) = \sum_{i=0}^{\infty} c_i A^i. \tag{24}$$

We begin with the same eigenvalue decomposition (6) of $\mathcal{W}$ and, assuming that $h(\mathcal{W})$ converges absolutely, its corresponding decomposition

$$h(\mathcal{W}) = U_1 h(\Lambda_1) U_1^\top + U_2 h(\Lambda_2) U_2^\top, \tag{25}$$

similar to (8), where U_1, U_2 have the same orthogonality properties (5).

Again, the first term is exactly the low-rank approximation that is used by spectral filtering, and the second is the approximation error

$$\begin{aligned} e_\alpha(\mathcal{W}) &:= U_2 h(\Lambda_2) U_2^\top && (26)\\ &= \sum_{i=0}^{\infty} c_i U_2 \Lambda_2^i U_2^\top && (27)\\ &= \sum_{i=0}^{\infty} c_i \left(U_2 \Lambda_2 U_2^\top \right)^i - c_0 U_1 U_1^\top && (28)\\ &= h(U_2 \Lambda_2 U_2^\top) - h(0) U_1 U_1^\top. && (29) \end{aligned}$$

Again, we have used the series expansion (24) of h in (27) and (29). Now, equation (28) is due to the fact that

$$(U_2 \Lambda_2 U_2^\top)^i = U_2 \Lambda_2^i U_2^\top \tag{30}$$

for $i \geq 1$, as can be verified by induction, while for $i = 0$,

$$U_2 \Lambda_2^0 U_2^\top = U_2 U_2^\top = I_n - U_1 U_1^\top = (U_2 \Lambda_2 U_2^\top)^0 - U_1 U_1^\top. \tag{31}$$

In both (30) and (31) we have used the orthogonality properties (5).

Finally, combining (25), (29) and (6), we have proved the following.

Theorem 2. *Assuming the series expansion (24) of transfer function h and the eigenvalue decomposition (6) of the symmetrically normalized adjacency matrix $\mathcal{W}$, and given that $h(\mathcal{W})$ converges absolutely, it is decomposed as*

$$h(\mathcal{W}) = U_1 g(\Lambda_1) U_1^\top + h(\mathcal{W} - U_1 \Lambda_1 U_1^\top), \tag{32}$$

where

$$g(A) := h(A) - h(\mathbf{O}) \tag{33}$$

for $n \times n$ real symmetric matrix A. For $h = h_\alpha$ and for $x \in [-1, 1]$ in particular, $g_\alpha(x) := h_\alpha(x) - h_\alpha(0) = (1 - \alpha)\alpha x / (1 - \alpha x)$.

This general derivation explains where the general definition of function g (33) is coming from in (16) corresponding to our treatment of h_α in section 4.1.

References

1. Arandjelovic, R., Zisserman, A.: Three things everyone should know to improve object retrieval. In: CVPR (June 2012)
2. Bai, S., Bai, X., Tian, Q., Latecki, L.J.: Regularized diffusion process on bidirectional context for object retrieval. IEEE Trans. PAMI (2018)
3. Bai, S., Zhou, Z., Wang, J., Bai, X., Jan Latecki, L., Tian, Q.: Ensemble diffusion for retrieval. In: ICCV (2017)
4. Chum, O., Mikulik, A., Perdoch, M., Matas, J.: Total recall II: Query expansion revisited. In: CVPR (June 2011)
5. Chum, O., Philbin, J., Sivic, J., Isard, M., Zisserman, A.: Total recall: Automatic query expansion with a generative feature model for object retrieval. In: ICCV (October 2007)
6. Chung, F.R.: Spectral graph theory, vol. 92. American Mathematical Soc. (1997)
7. Delvinioti, A., Jégou, H., Amsaleg, L., Houle, M.: Image retrieval with reciprocal and shared nearest neighbors. In: VISAPP (January 2014)
8. Donoser, M., Bischof, H.: Diffusion processes for retrieval revisited. In: CVPR (2013)
9. Gordo, A., Almazan, J., Revaud, J., Larlus, D.: End-to-end learning of deep visual representations for image retrieval. IJCV **124**(2) (2017)
10. Hackbusch, W.: Iterative solution of large sparse systems of equations. Springer Verlag (1994)
11. Iscen, A., Avrithis, Y., Tolias, G., Furon, T., Chum, O.: Fast spectral ranking for similarity search. In: CVPR (2018)
12. Iscen, A., Tolias, G., Avrithis, Y., Furon, T., Chum, O.: Efficient diffusion on region manifolds: Recovering small objects with compact cnn representations. In: CVPR (2017)
13. Jégou, H., Harzallah, H., Schmid, C.: A contextual dissimilarity measure for accurate and efficient image search. In: CVPR (June 2007)
14. Joly, A., Buisson, O.: Logo retrieval with a contrario visual query expansion. In: ACM Multimedia (October 2009)
15. Kalantidis, Y., Mellina, C., Osindero, S.: Cross-dimensional weighting for aggregated deep convolutional features. arXiv preprint arXiv:1512.04065 (2015)
16. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching. In: CVPR (June 2007)
17. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Lost in quantization: Improving particular object retrieval in large scale image databases. In: CVPR (June 2008)
18. Qin, D., Gammeter, S., Bossard, L., Quack, T., Van Gool, L.: Hello neighbor: Accurate object retrieval with k-reciprocal nearest neighbors. In: CVPR (2011)
19. Radenović, F., Iscen, A., Tolias, G., Avrithis, Y., Chum, O.: Revisiting oxford and paris: Large-scale image retrieval benchmarking. In: CVPR (2018)
20. Radenović, F., Tolias, G., Chum, O.: Fine-tuning CNN image retrieval with no human annotation. IEEE Trans. PAMI (2018)
21. Shen, X., Lin, Z., Brandt, J., Wu, Y.: Spatially-constrained similarity measure for large-scale object retrieval. IEEE Trans. PAMI **36**(6), 1229–1241 (2014)
22. Sivic, J., Zisserman, A.: Video Google: A text retrieval approach to object matching in videos. In: ICCV (2003)
23. Tolias, G., Jégou, H.: Visual query expansion with or without geometry: refining local descriptors by feature aggregation. Pattern recognition **47**(10), 3466–3476 (2014)

24. Tolias, G., Sicre, R., Jégou, H.: Particular object retrieval with integral max-pooling of cnn activations. ICLR (2016)
25. Tong, H., Faloutsos, C., Pan, J.Y.: Fast random walk with restart and its applications. In: Proceedings of the IEEE International Conference on Data Mining. pp. 613–622 (2006)
26. Trefethen, L.N., Bau III, D.: Numerical linear algebra. SIAM (1997)
27. Zhang, S., Yang, M., Cour, T., Yu, K., Metaxas, D.N.: Query specific fusion for image retrieval. In: ECCV (2012)
28. Zhou, D., Weston, J., Gretton, A., Bousquet, O., Schölkopf, B.: Ranking on data manifolds. In: NIPS (2003)

XX Visual query expansion with or without geometry: refining local descriptors by feature aggregation

Title: Visual query expansion with or without geometry: refining local descriptors by feature aggregation
Authors: G. Tolias, H. Jégou
Published at: Pattern Recognition 2014

Visual query expansion with or without geometry: refining local descriptors by feature aggregation

Giorgos Tolias, Hervé Jégou

Inria

Abstract

This paper proposes a query expansion technique for image search that is faster and more precise than the existing ones. An enriched representation of the query is obtained by exploiting the binary representation offered by the Hamming Embedding image matching approach: The initial local descriptors are refined by aggregating those of the database, while new descriptors are produced from the images that are deemed relevant.

The technique has two computational advantages over other query expansion techniques. First, the size of the enriched representation is comparable to that of the initial query. Second, the technique is effective even without using any geometry, in which case searching a database comprising 105k images typically takes 79 ms on a desktop machine. Overall, our technique significantly outperforms the visual query expansion state of the art on popular benchmarks. It is also the first query expansion technique shown effective on the UKB benchmark, which has few relevant images per query.

Keywords: image retrieval, query expansion, hamming embedding

1. Introduction

This paper considers the problem of image and object retrieval in image databases comprising up to millions of images. The goal is to retrieve the images describing the same visual object(s) as the query. In many applications, the query image is submitted by a user and must be processed in interactive time.

Most of the state-of-the-art approaches derive from the seminal Video-Google technique [1]. It describes an image by a bag-of-visual-words (BOVW) representation, in the spirit of the bag-of-words frequency histograms used in text information retrieval. This approach benefits from both the powerful local descriptors [2, 3] such as the SIFT, and from indexing techniques inherited from text information retrieval such as inverted files [4, 5]. Exploiting the sparsity of the representation, BOVW is especially effective for large visual vocabularies [6, 7].

This analogy with text representation is a long-lasting source of inspiration in visual matching systems, and many image search techniques based on BOVW have their counterparts in text retrieval. For instance, some statistical phenomenons such as burstiness or co-occurrences appear both in texts [8, 9] and images [10, 11, 12] and are addressed in similar ways.

One of the most successful techniques in information retrieval is the query expansion (QE) principle [13], which is a kind of automatic relevance feedback. The general idea is to exploit the reliable results returned by an initial query to produce an enriched representation, which is re-submitted in turn to the search engine. If the initial set of results is large and accurate enough, the new query retrieves some additional relevant elements that were not present in the first set of results, which dramatically increases the recall.

Query expansion has been introduced to the visual domain by Chum et al. [14], who proposed a technique implementing the QE principle and specifically adapted to visual search. Several extensions have been proposed to improve this initial QE scheme [15, 16, 17]. Although these variants have improved the accuracy, they suffer from two inherent drawbacks which severely affect the overall complexity and quality of the search:

- First, they require a costly geometrical verification step, which provides the automatic annotation of the relevant set and is typically performed on hundreds of images.
- Second, the augmented query representation contains significantly more non-zero components than the original one, which severely slows down the search. It is reported [17] that typically ten times more components are non-zeros. Since querying the inverted file has linear complexity in the number of features contained in the query vector, the second query is therefore one order of magnitude slower than the first.

Expansion methods that do not use any costly geometrical verification are typically based on an off-line stage with quadratic complexity in the number of database images [18, 19, 20]. They are thus limited to collections of small and fixed size.

In another line of research, several techniques address the loss in quantization underpinning BOVW, such as the use of multiple assignment [21] or soft quantization [22]. In a complementary manner, the Hamming Embedding (HE) technique [23] dramatically improves the matching quality by refining the descriptors with binary signatures. HE is not compatible with existing QE techniques because these assume a vector representa-

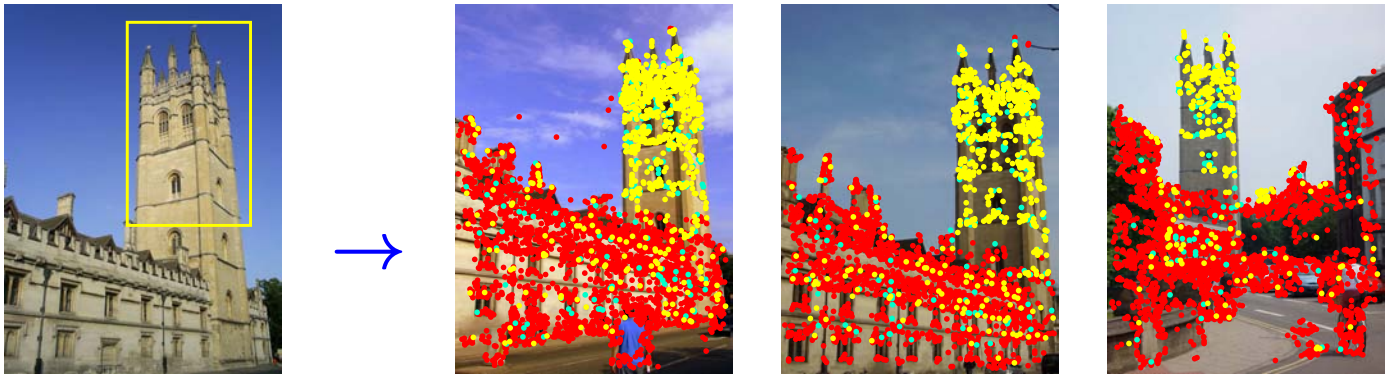


Figure 1: Query image (left) and the features selected (yellow+cyan) from the retrieved images to refine the original query. The features in red are discarded. Cyan features correspond to visual words that appear in the query image, and yellow ones to visual words that were not present in it. The selection of the depicted images and features has not involved any geometrical information.

tion of the images. A noticeable exception is the transitive QE, which does not explicitly exploit the underlying image representation. However, this variant is not satisfactory with respect to query time and performance.

This paper, for the first time, proposes a novel way to exploit the QE principle in a system that individually matches the local descriptors, namely the HE technique. The new query expansion technique is both efficient and precise, thanks to the following two contributions:

- First, we modify the selection rule for the set of relevant images so that it does not involve any spatial verification. The images deemed relevant provide additional descriptors that are employed to improve the original query representation. Unlike other QE methods, it is done *on a per-descriptor basis* and not on the global BOVW vector. Figure 1 depicts an example of images and features that are selected by our method to refine the original query.
- The second key property of our method is that the set of local features is aggregated to produce new binary vectors defining the new query image representation. This step drastically reduces the number of individual features to be considered when submitting the enriched query.

To our knowledge, it is the first time that a visual QE is successful without any geometrical information: The only visual QE technique [14] that we are aware of performs poorly compared with other variants such as the average query expansion (AQE). In contrast, our technique used without geometry reaches or outperforms the state of the art. Interestingly, it is effective even when a query has few corresponding images in the database, as shown by our results on the UKB image recognition benchmark [6]. Incorporating geometrical information in the pipeline further improves the accuracy. As a result, we report a large improvement compared to the state of the art. We further demonstrate the superiority of our method compared to a simple combination of HE with QE: The property of feature aggregation not only reduces the expanded query complexity, but further improves performance.

The paper is organized as follows. Section 3 introduces our core image system and Section 7 a post-processing technique for SIFT descriptors that is shown useful to improve the efficiency of the search. Section 4 introduces our Hamming Query Expansion (HQE) method and Section 5 describes our key aggregation strategy of local features. Section 6 describes how to exploit geometrical information with HQE. The experimental results presented in Section 8 demonstrate the superiority of our approach over concurrent visual QE approaches, with respect to both complexity and search quality, on the Oxford5k, Oxford105k and Paris benchmarks.

2. Related work

Chum and colleagues [14] were the first to translate the query expansion principle to the visual domain. Most of the variants they propose rely on a spatial verification method, which filters out the images that are not geometrically consistent with the query. The authors investigate several methods to build a new query from the images deemed relevant. The average query expansion is of particular interest and usually considered as a baseline, as it is the most efficient variant [14] and provides excellent results. It is conceptually simple: A new *term-frequency inverse document frequency* (TFIDF) vector is obtained as the average of the results assumed correct and spatially back-projected to the original image.

Following this first work, a number of QE variants and extensions have been proposed [15, 16, 17]. Using incremental spatial re-ranking, the query representation is updated by each spatially verified image and extended out of the initial query region [16]. Another extension is to learn, on-the-fly, a discriminative linear classifier [17] to define the new query instead of the average in AQE.

Other kinds of expansion have been proposed for fixed image collections [17, 24]. They rely on the off-line pairwise matching of all images pairs and aim at identifying the features coming from the same object using spatial verification, which is rather costly as the complexity is quadratic in the number of images. They also assume that the image collection is fixed: The selection depends on a given set of images. These methods are also related to other methods exploiting the neighborhood of the images within a given collection [21, 18], in particular

by updating the comparison metric or by employing reciprocal nearest neighbors as a filtering rule. For instance, Qin *et al.* [18] constructs a graph that links related images, and uses k-reciprocal nearest neighbors at query time to define a new similarity function that re-orders the images. Again, the cost of constructing and storing the graph in memory is impracticable for large datasets. In a similar spirit, Shen *et al.* [19] exploit the ranked lists of independent queries issued with top-ranked images. Query time increases significantly as it is linear to the number of those queries. A recent graph-based method combines multiple similarity measures to perform re-ranking [20]. The cost of such an offline procedure can be undertaken only for small collections. In the work of Chum and Matas [25], the quadratic cost is addressed by starting from seed query images, yet their method requires a costly spatial verification stage.

The query expansion method of Kuo *et al.* [26] is also related to our work. They also use a set of binary vectors for an image representation and try to identify database image regions which are similar to the query. The initial representation is not enhanced, but new independent queries are rather issued and a final fusion is performed on the ranked lists. Li *et al.* [27] straightforwardly use binary descriptors but only to select the relevant matches. As in other QE methods, their method relies on geometry and produces a larger set of features.

3. The core image system

This section describes the image search system based on Hamming Embedding upon which our query expansion techniques are built. This baseline method follows the guideline of the existing HE technique [23], which proceeds as follows. An image is represented by a set $\mathcal{P}$ of local SIFT descriptors [3] extracted with the Hessian-Affine detector [28].

BOVW and Hamming Embedding. The descriptors are quantized using a flat k-means quantizer, where k determines the visual vocabulary size. A descriptor $p \in \mathcal{P}$ is then represented by a quantization index, called a visual word $v(p)$. Computing and normalizing the histogram of visual words produces the BOVW representation. It can also be seen as a voting system in which all descriptors assigned to a specific visual word are considered as matching with a weight related to the inverse document frequency [1, 23].

In order to refine the quality of the matches and to provide more reliable weights to the votes, the HE technique [23] further refines each descriptor p by a binary signature $b(p)$, providing a better localization of the descriptor by subdividing the quantization cell $v(p)$. HE compares two local descriptors q and p that are assigned to the same visual word $v(p) = v(q)$ by computing the Hamming distance $h(q,p) = \|b(q) - b(p)\|_1$ between their binary signatures. If the Hamming distance is above a predefined threshold h_t, the descriptors are considered as non matching and zero similarity is attached. A significant benefit [23, 10] in accuracy is obtained by weighting the vote as a decreasing function of the Hamming distance. In this paper, we adopt the Gaussian function used in [10] with σ equal to one fourth of the binary signature size.

The burstiness phenomenon in images was first revealed and tackled by Jegou *et al.* [10]. It takes into account descriptors that individually trigger multiple matches between specific pairs of images, which is often the case because of repetitive structures, or features which are abnormally common across all database images. Several normalizations have been proposed, from which we adopt the one that down-weights a given match score by the square root of the number of matches associated with the corresponding query descriptor [10]. This strategy is similar to the successful component-wise power-law normalization later proposed for BOVW or Fisher Kernels [29], but here applied to a voting technique.

Multiple assignment (MA). BOVW and HE handles descriptors assigned to the same visual word. However quantization losses are introduced when truly matching descriptors are assigned to different visual words. This has been addressed by assigning multiple visual words to each descriptor [21, 22]. We apply MA on the query side only in order to keep memory requirements unchanged [23]. In the rest of the paper, the initial method that assigns a descriptor a single visual word is denoted by SA (single assignment) to distinguish it with MA.

Figure 2: Matching features using BOVW (top), HE with $h_t = 24$ (middle) and HE with $h_t = 16$ (bottom).

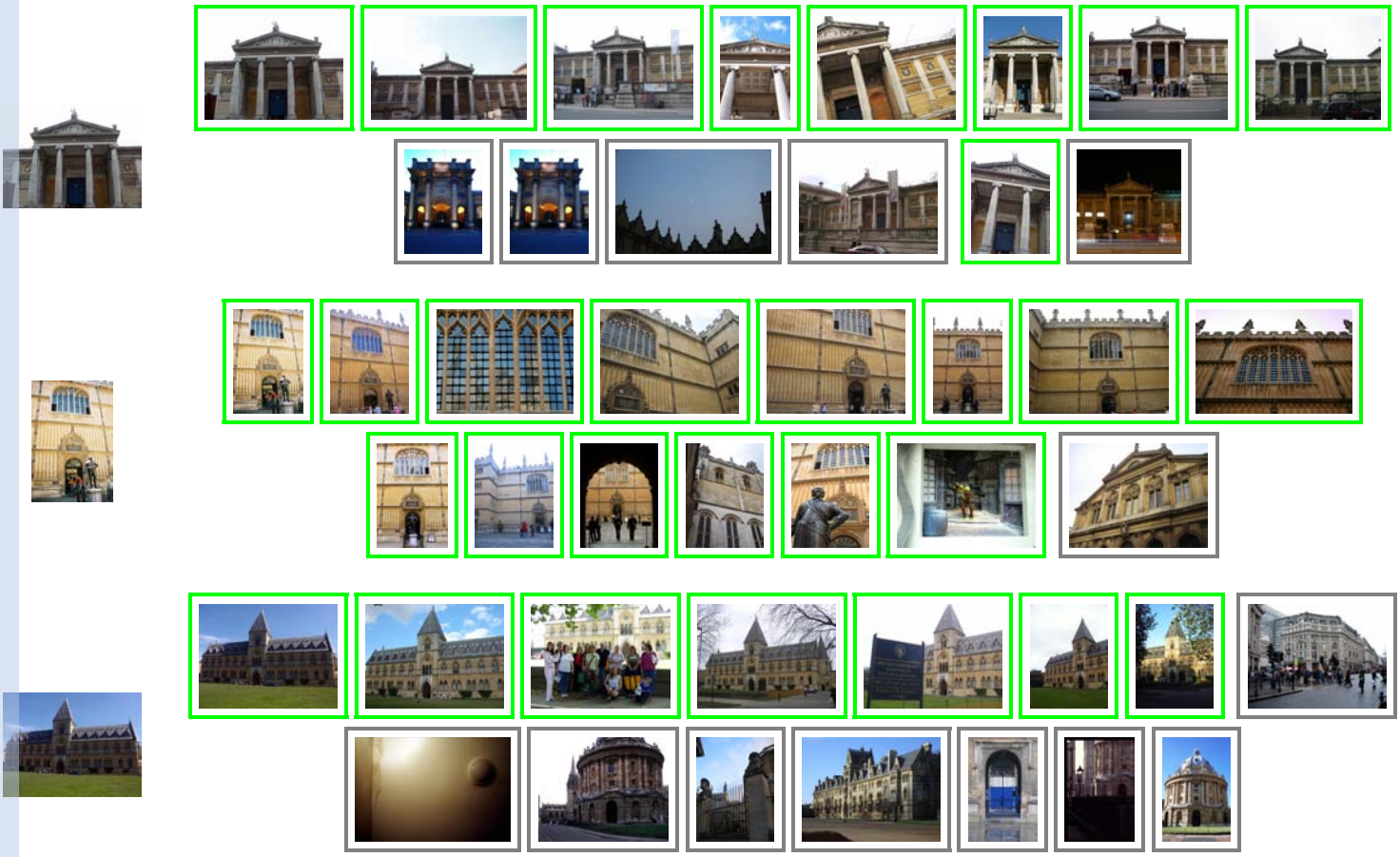


Figure 3: Examples of query images (left) and the corresponding top ranked lists by the baseline retrieval system. Images (not) selected as reliable are marked with (gray) green border.

4. HE with query expansion

This section defines a query expansion technique based on HE and not involving any geometrical information. We revisit the different stages involved in the QE principle. We first describe how reliable images are selected from the initial result set. Then we detail the way an enriched query is produced from the images deemed relevant. The key subsequent aggregation step and the use of geometry will be introduced later in Sections 5 and 6, respectively.

4.1. Selection of reliable images

As in all query expansion methods [14, 15, 16, 17], the core image search system processes an initial query. The resulting set is analyzed to identify a subset of reliable images that are likely to depict the query object, and therefore to provide additional features that will be subsequently exploited in the augmentation stage.

In the following, we will denote the local features of the query image by $\mathcal{Q}$, and those of a given database image by $\mathcal{P}$, respectively. As a criterion to determine the relevant images, we count the number $C(\mathcal{Q},\mathcal{P})$ of "strict" feature correspondences between the query and images in the short-list. It is given by

$$C(\mathcal{Q},\mathcal{P}) = \left|\{(q,p) \in \mathcal{Q}\times\mathcal{P} : h(q,p) \leq h_{\mathrm{t}}^{\star}\}\right|, \qquad (1)$$

where the threshold $h_{\mathrm{t}}^{\star}$ is lower than the Hamming embedding threshold h_{t} used for initial ranking. Such a lower threshold allows for a higher true positive to false positive ratio of matches [23]. It provides a strict way to count correspondences in a manner that resembles the number of RANSAC inliers commonly used to verify the images [7]. It is less precise than RANSAC, yet it has the advantage of not using any geometry. It is therefore much faster.

Figure 2 illustrates, for a pair of images, the matching features obtained using BOVW and HE. We consider two different thresholds for HE to show the impact of the strict threshold $h_{\mathrm{t}}^{\star} = 16$ on selected features. Observe that HE matching filters out many false matches compared to BOVW. With a lower threshold value, the filtering is not far in quality from that of a spatial matching method.

An image is declared reliable if at least c_{t} correspondences are satisfied, which formally leads to define the set of *reliable images* as

$$\mathcal{L}_{\mathcal{Q}} = \{\mathcal{P} : C(\mathcal{Q},\mathcal{P}) \geq c_{\mathrm{t}}\}. \qquad (2)$$

In practice, only the images short-listed in the initial search are considered as candidates for the set of reliable images. In our experiments, we count the number of correspondences with Equation 1 only for the top 100 images. Figure 3 shows examples of queries and the corresponding reliable images. Although

Figure 4: Sample reliable images and features assigned to reliable visual words, when geometry is not used. *Left:* Query image. *Top:* Features assigned to reliable visual words that appear in the query image. *Bottom:* Features in the set of augmented visual words. Note: we only show a subsample of the actual reliable visual words. Each color represents a distinct visual word.

some negative images are selected and some positive ones are not, the result is not far from what spatial verification would produce. This suggests that selecting reliable images with HE and a low threshold is sufficient for the purpose of QE, as proposed in this section. The proposed procedure for detecting reliable images gives a rate of 92.4% true positive instances in $\mathcal{L}_\mathcal{Q}$. Note, this is achieved without any geometry information.

4.2. *Feature set expansion*

First, let us recall that a feature descriptor is described by both a visual word and a binary signature. Our augmentation strategy, *i.e.*, how we introduce new local features in the representation, is partly based on the selection of visual words that are not present in the original query.

Since a large proportion of the reliable images depicts the same object, the visual words frequently occurring in the images of the reliable set $\mathcal{L}_\mathcal{Q}$ are likely to depict the query object rather than the background. Our selection strategy is simple and consists in selecting the most frequent visual words occurring in $\mathcal{L}_\mathcal{Q}$. More precisely, we sort the visual words contained in the images of $\mathcal{L}_\mathcal{Q}$ by the number of reliable images in which they appear. The top ranked words are selected and define the set of *reliable visual words* $\mathcal{V}$, which may include both visual words that are present or absent in the query image. The latter are referred to as the *augmented visual words*. Their count is controlled by a parameter α to ensure that the number of reliable visual words in the new query is proportional to that of the original query, as

$$|\mathcal{V} \setminus \mathcal{V}_\mathcal{Q}| = \alpha \cdot |\mathcal{V}_\mathcal{Q}|, \tag{3}$$

where $\mathcal{V}_\mathcal{Q}$ is the set of visual words occurring in the query. A typical value of parameter α is 0.5 (see Section 8).

The initial query set is enriched with the features of the reliable images assigned to the reliable visual words. Define as

$$\mathcal{G} = \{p \in \mathcal{P} : \mathcal{P} \in \mathcal{L}_\mathcal{Q} \wedge v(p) \in \mathcal{V}\} \tag{4}$$

the union of all features of reliable images assigned to some reliable words. It defines the set of database features used to augment the initial query. In other terms, this set is merged with the initial query feature set to construct the augmented query as

$$\mathcal{Q}_E = \mathcal{Q} \cup \mathcal{G}. \tag{5}$$

Figure 4 depicts some features from reliable images assigned to reliable visual words. Observe that, even without any spatial information, selected visual words are detected on the foreground object. Moreover, each visual word corresponds not only to similar image patches, but often to the exact same patch of the object, as if spatial matching was used. This appears to be the case for either visual words which appear (top) or miss (bottom) in the query.

A simple way to construct an enriched query is to use the expanded set of features as the new image representation. However, similar to existing QE strategies, such an approach leads to a high complexity because the number of features explodes. We observe that it is typically multiplied by a factor ranging from 10 to 20 for typical values of α, as analyzed in the experimental section 8. This drawback is shared by other effective techniques on query expansion [17], for which this problem leads to produce a BOVW vector having 10 times more non-zero components than the initial one. In the next section, we address this issue by proposing an aggregation strategy that drastically reduces the number of features.

5. QE with feature aggregation

The average query expansion technique [14] averages BOVW vectors to produce the new query. In this section, we explain how local descriptors are *individually* refined or created from binary signatures of the set of reliable features. At this stage, the augmented set contains multiple instances representing the same visual patches, either in the initial query or not. Descriptors associated to the same patch are expected to have similar binary signatures. The strategy presented below implicitly exploits this underlying property to produce the new set of query descriptors which is less redundant.

First, note that the selection strategies for images and features presented in the previous subsections introduce a few false positives in the augmented feature set. This is the cost to pay for not performing the selection with a stringent spatial matching technique: Our inliers are not selected as reliably as in other query expansion methods. The aggregation operation proposed hereafter comes as a complement on our selection method, as it is robust enough to false positives. In contrast, averaging over normalized TFIDF vectors of similar images [14], as done in AQE, is sensitive to background and noisy features.

Our aggregation scheme is inspired by methods [30, 31] such as the VLAD technique, which aggregates the descriptors per visual word to produce a vector representation of an image. In our method, we aggregate the features of $\mathcal{Q}_E$ that are assigned to the same visual word. Therefore, our technique produces exactly one binary signature per visual word occurring in $\mathcal{Q}_E$. Our motivation is that the true matching patches are likely to overrule the false positives. This actually happens in practice because the true correspondences are more numerous and are associated with more consistent binary signatures.

Our aggregation scheme is related to the recent work of Tolias *et al.* [32], where descriptors are aggregated per visual word for query and database images individually. A selectivity function is employed to appropriately weight the similarity scores. Our approach differs in that we rather aggregate descriptors collected from many images instead of a single one. In their work, aggregation consistently improves the performance in all cases. It is attributed to the way burstiness is handled. As a result, the voting scheme ensures that at most one correspondence is established for each visual word, and therefore at most one for each descriptor.

For each visual word v appearing in $\mathcal{Q}_E$, a new binary signature $b(v)$ is obtained by computing the median values over all the bit vectors occurring in $\mathcal{Q}_E$ and assigned to v. If the numbers of 0 and 1 are equal for a particular bit component, the tie is arbitrarily resolved by assigning either 0 or 1 with equal probabilities. This new set of descriptors comprises exactly one binary signature per visual word and serves as the new query, which is then submitted to the system.

In the remainder of this paper, we refer to the method described in this section as Hamming Query Expansion (HQE).

Remark. HQE differs from a simple combination of HE with QE. Firstly, our QE scheme is the first not to use any geometrical information in order to identify relevant images. Secondly, only the most frequent visual words appearing among relevant images are collected, avoiding the inclusion of false matches to the expanded query. Finally, the proposed feature aggregation, in addition to drastically reduce the expanded query size, further improves the performance. As shown in our experiments.

6. Geometrical information

This section proposes a variant of our method to further eliminate some incorrect matches by including some spatial information in the loop. For this reason and as shown later in the experimental section, it is not as fast as the HQE strategy proposed in Sections 4 and 5. However, this approach further improves the performance and is therefore interesting in some situations where one would trade an interactive time against any improvement in accuracy.

It proceeds as follows. The matches are collected with the regular HE technique, *i.e.*, they are returned by the first query. Instead of calculating the number of correspondences with Equation 1, we rely on the number $C_g(\mathcal{Q}, \mathcal{P})$ of inliers found with a spatial matching technique. For this purpose, we have used the spatial verification procedure proposed by Philbin *et al.* [7]. Similar to other QE techniques, this procedure is applied on the top ranked images only. An image is declared reliable if the number of inliers is above a pre-defined threshold. The estimation of the affine transformation is then further exploited to filter the expanded feature set. As first suggested by Chum *et al.* [14], the matching features associated with the reliable images are projected back to the query image plane. Those falling out of the query image borders are filtered out.

The remaining steps of this variant then become similar to the HQE method of Sections 4 and 5. The only difference is that the input set of reliable features is different. Therefore, we first select the reliable visual words and perform the feature set expansion. The aggregation is similarly applied to produce one binary vector per visual word. Note that, the reliable images, as detected by spatial matching, are ranked in top positions.

Figure 5 depicts the descriptors selected for the HQE expanded set with and without geometry. Notice that even without geometry, most of the selected features are localized on the target object. The geometry effectively filters out the remaining features that do not lie on the query object.

7. Implementation details

In this section we introduce a new post-processing stage for SIFT descriptors, which interest is evaluated in different setups.

Root-SIFT. It was recently shown [17, 33] that square rooting the components of SIFT descriptors improves the search performance. This is done either by L_1-normalizing the SIFT descriptor [17] prior to the square-rooting operation or, equivalently, by [33] square-rooting the components and normalizing the resulting vector in turn with respect to L_2. This operation amounts to computing the Hellinger distance instead of the Euclidean one. The impact of this scheme is evaluated in Table 1 on the Oxford5k building benchmark [7] for both BOVW and HE, without the burstiness processing. Following the standard

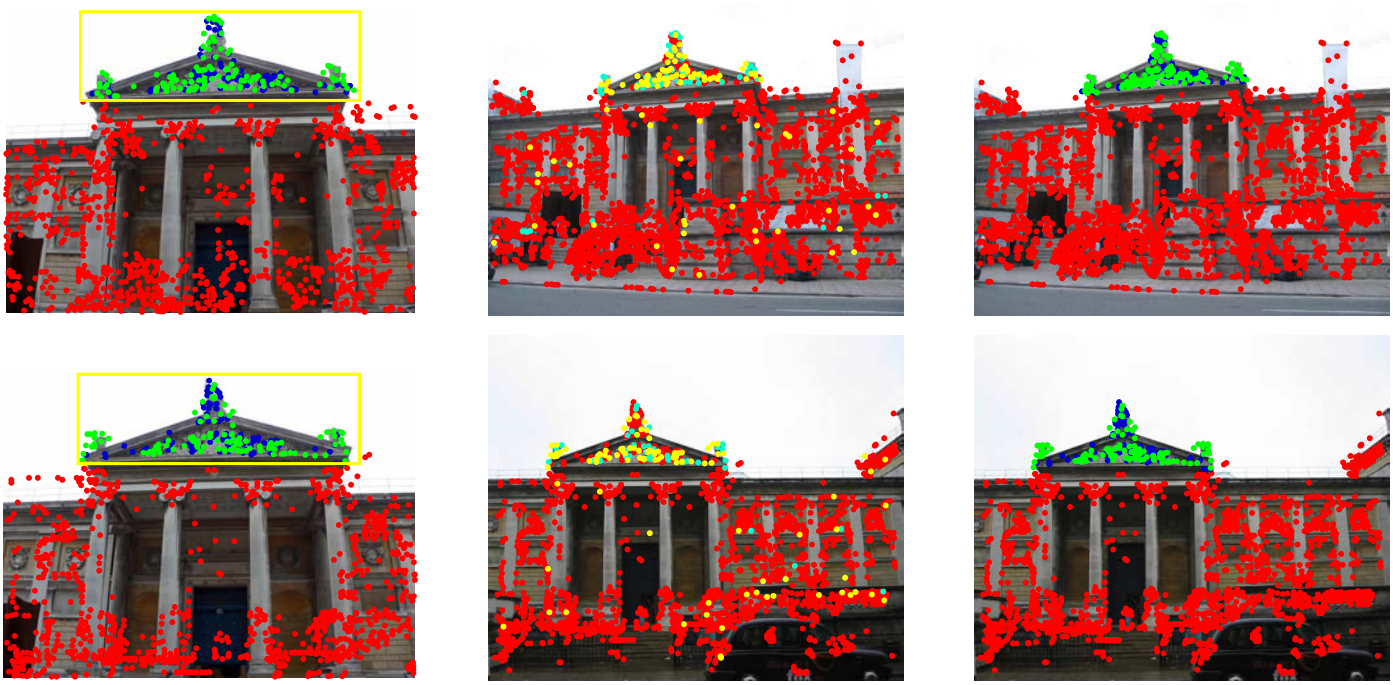


Figure 5: Features selected for the expanded set of a particular query image (left) without (middle) and with spatial matching (right). With spatial matching: features back-projected out of the bounding box are rejected (red), while the rest (blue and green) are kept. Those assigned to reliable visual words are shown in green. Without spatial matching: features assigned to reliable visual words are shown in cyan or yellow, with yellow being the ones assigned specifically to augmentation visual words and cyan the ones assigned to visual words that appear in the query. Rejected are shown in red. *Best viewed in color*.

$\sqrt{}$	$-\mu$	mAP/BOVW	mAP/HE	IF
		47.7 $\pm$*0.8*	67.1 $\pm$*0.6*	1.200 $\pm$*0.003*
x		47.7 $\pm$*0.5*	69.5 $\pm$*0.8*	1.290 $\pm$*0.003*
x	x	48.1 $\pm$*0.7*	69.6 $\pm$*0.8*	1.238 $\pm$*0.003*

Table 1: Evaluation with respect to mAP (performance) and IF (efficiency) of several post-processing procedures for SIFT: RootSIFT [17, 33] (denoted by $\sqrt{}$) and shifting (denoted by -μ). Post-processed descriptors are used to create the codebook, perform assignment to visual words and create the binary signatures used in HE. We have performed 10 independent experiments on Oxford5k with distinct vocabularies (k=16k) to report mean performance and standard deviations.

evaluation protocol, we measure the mean average precision (mAP). In order to cope with the variability of the results due to the sensitivity of k-means to the initial random seeds, we average the results over 10 runs with different vocabularies and report the standard deviation. We observe an improvement provided by square-rooting the components, which is statistically significant when used with HE.

However, as a side-effect of this processing, we observe that Root-SIFT introduces an unexpected complexity overhead, resulting from less balanced inverted lists. The undesirable impact of uneven inverted lists was first noticed by Nister et al. [6] and is commonly measured by the imbalance factor (IF) [23], which is a multiplicative factor reflecting the deviation from perfectly balanced lists. For instance, IF=2 means that, on average, two times more individual descriptor comparisons are performed compared to the case where the lists have equal lengths. Table 1 shows that this negative effect, which was not reported for this RootSIFT variant [17, 33], is statistically significant.

Shift-SIFT. In order to reduce this undesirable effect, we introduce another processing method for SIFT descriptors referred to as shift-SIFT. It is inspired by the approach proposed for BOVW vectors [34], which aims at handling "negative evidences" by centering the descriptors and L_2 normalizing them in turn. It gives more importance in the comparison to the components which are close to 0, and improves the performance in the case of BOVW vectors.

Table 1 shows the interest of this shifting strategy applied to SIFT descriptors. We have use SA and no burstiness normalization in this experiment (conclusions are similar in other setups). Performance is mainly unaffected. Yet, this approach provides more balanced lists and therefore reduces the search complexity by about 4% at no cost, as reflected by the IF measure. We use this shifting strategy combined with L_2 Root-SIFT [33] in all our experiments.

8. Experiments

This experimental section first introduces the datasets, gives details about the experimental setup, and evaluates the impact of the parameters and variants. Our technique is then compared with the state of the art on visual query expansion before a discussion on the complexity.

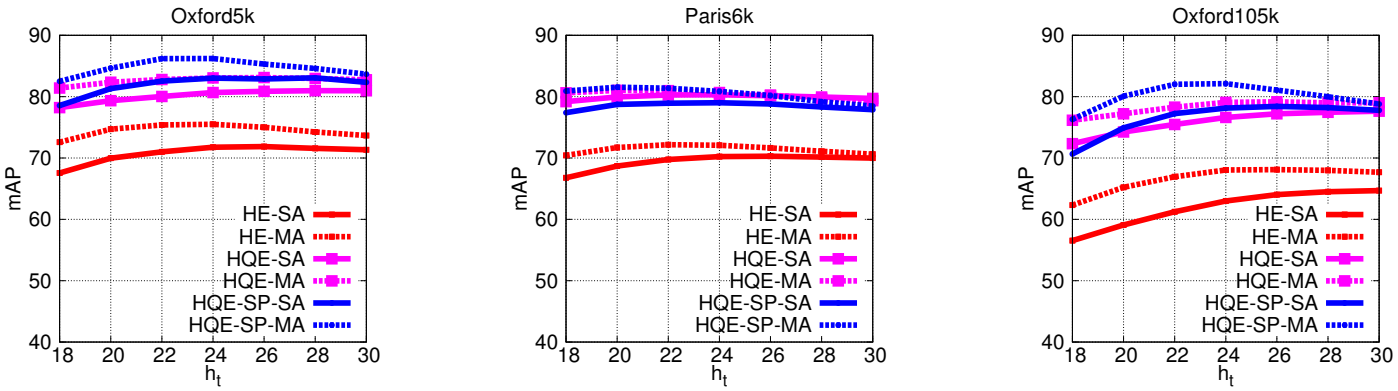


Figure 6: Impact of h_t on the performance of HE and HQE.

8.1. Datasets and experimental setup

Datasets and measures. Query expansion techniques are only effective if the dataset consists of several relevant images for a given query. We evaluate the proposed method on two publicly available datasets of this kind, namely Oxford5k Buildings [7] and Paris [22], but also on a dataset where queries have only few corresponding images, that is UKB [6]. Following the standard evaluation protocols, we report mean Average Precision (mAP) for the two first and use the score definition associated with UKB: the average number of correct images ranked in first 4 positions (from 0 to 4). As for other QE works, the large scale experiments are carried out on the Oxford105k dataset, which augments Oxford5k with 100k additional distractor images.

Features and experimental setup. For Oxford5k and Paris, we used the modified Hessian-Affine detector proposed by Perdoch *et al.* [35] to detect local features. The extracted SIFT descriptors have been subsequently post-processed by using the L_2 Root-SIFT and shift-SIFT procedure, as described in Section 3. For UKB, we have used the same features provided by the authors of the papers [21, 18]. We follow the more realistic, less biased approach, of learning the vocabulary on an independent dataset. That is, when we use Oxford5k for evaluation, the vocabulary is learned with features of Paris and *vice versa*. Similarly, learning the medians of Hamming Embedding is carried out on the independent dataset.

Unless otherwise stated, we use a visual vocabulary comprising $k = 65,536$ visual words, binary signatures of 64 dimensions, and apply HE with weights and burstiness normalization. In all our experiments, the reliable images for our approach, either without or with spatial matching, are selected among top 100 ones returned by the baseline system. When using MA, it is applied on the query side using the 3 nearest visual words to limit the computational overhead of using more.

MA produces more correspondences than single assignment (SA), therefore the probability of finding a false positive match is increased even with spatial matching and the matching parameters should be stricter [23]. We set the minimum number of correspondences to $c_t = 4$ with SA and to $c_t = 5$ with MA.

Two factors introduce some randomness in the measure with our approach: The random projection matrix (in HE) and the random decision used to resolve ties when aggregating binary signatures. Therfore, each experiment is performed 5 times using distinct and independent parameters. We report the average performance and standard deviation to assess the significance of our measurements.

8.2. Impact of the parameters

Thresholds. The strict threshold $h_t^\star$ is constant and set to 16 in all our experiments. Figure 6 shows the impact of the parameter h_t. The performance is not very sensitive around the optimal values attained at $h_t = 22$ or $h_t = 24$, depending on the setup. Note already that HQE gives a significant improvement compared to the HE baseline. In the rest of our experiments, we set $h_t = 24$ in all cases, similar to most works based on HE. We have fixed $\alpha = 0.5$ for this preliminary experiment, which implies that the size of the new query is at most 1.5 times larger than the initial one. In practice, it is much smaller thanks to the descriptors aggregation. See, for instance, Table 3 to compare the average number of descriptors used in the original and augmented queries.

The parameter α (see Section 4) controls the size of the augmented query. Figure 7 presents its impact on the performance. HQE without spatial matching rapidly attains its maximum in performance and then decreases. This suggests that not too many visual words should be selected because the additional ones will introduce many outliers compared to inliers. In contrast, spatial matching filters out most of the outliers: Using more descriptors is better because the added ones are inliers in their majority. As a compromise between performance and complexity, we set $\alpha = 0.5$ and 1.0 without and with spatial matching, respectively.

The vocabulary size k is critical in most methods based on pure BOVW. Figure 8 that it is not the case with HE and our techniques, which achieves excellent performance for all the sizes. This confirms observations in prior work [23, 36].

Weights, burstiness and HQE. Table 2 summarizes the respective contributions of the different elements of our search engine. First note the large gain already achieved by weighting the Hamming distances in HE, using MA and applying the

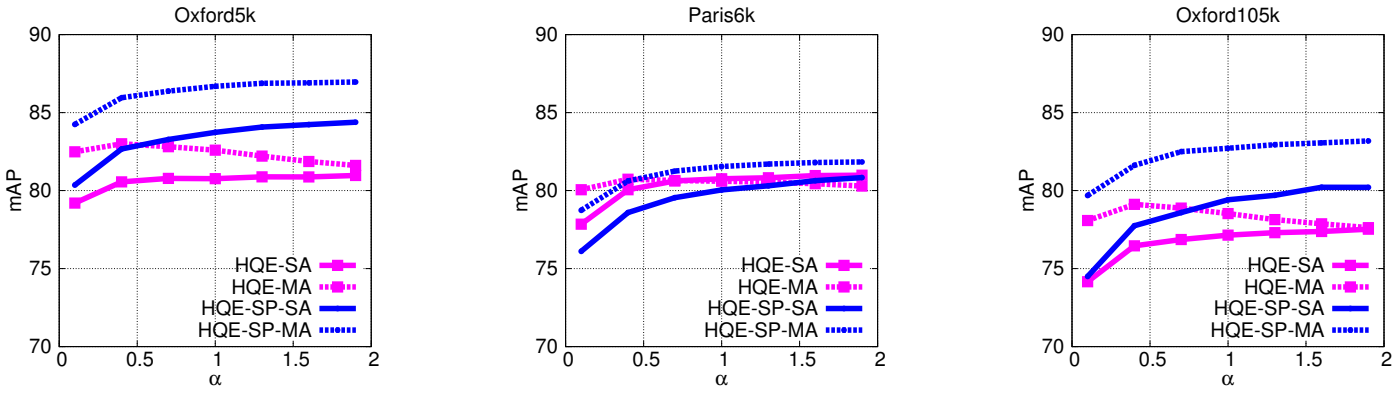


Figure 7: Impact of α. Performance of HQE when varying the number of new visual words in the expanded query.

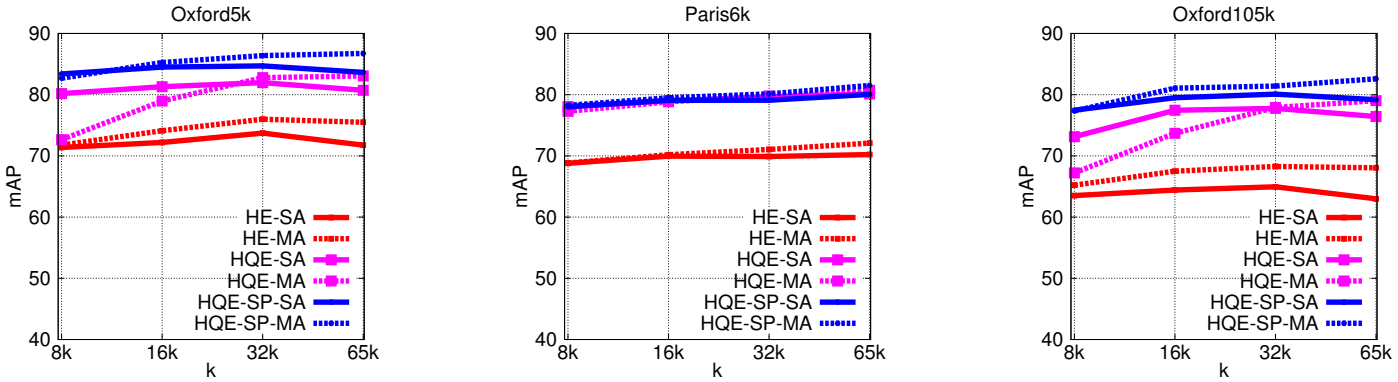


Figure 8: Impact of the vocabulary size on the performance of HE, HQE and HQE with spatial matching.

W	burst	MA	HQE	SP	Oxford5k	Paris6k	Oxford105k
					66.9	65.7	55.5
×					70.4	68.4	59.6
×	×				71.7	70.2	62.9
×	×	×			75.4	72.0	68.0
×	×	×	×		83.0	80.6	79.0
×	×	×	×	×	86.8	81.5	82.6
BOVW					53.3	54.8	44.2

Table 2: Mean average precision for separate components comprising the proposed method. Initial method is the original Hamming Embedding without weights. W=weighting similarities. MA=multiple assignment.

Dataset	Method	mAP		$\|Q\|$	
		SA	MA	SA	MA
Oxford5k	HE	71.7	75.4	1,362	4,088
	HQE/b.a.	79.0	82.0	11,937	27,345
	HQE	80.7	**83.0**	1,810	**5,030**
Paris6k	HE	70.2	72.0	1,460	4,382
	HQE/b.a.	76.6	77.3	35,982	66,665
	HQE	80.2	**80.6**	1,843	**5,045**
Oxford105k	HE	62.9	68.0	1,362	4,088
	HQE/b.a.	73.5	76.5	12,176	28,699
	HQE	75.6	**79.0**	1,810	**5,030**

Table 3: Performance and average query size $|Q|$ for the baseline HE, HQE and the use of the same expanded query before aggregation (HQE/b.a.). Note that the aggregation procedure is a key step: not only it significantly reduces the complexity (number of features), but it also improves the performance.

burstiness procedure [10]. Note also the even larger improvement obtained by using our HQE technique, either with or without spatial matching.

Aggregation. Table 3 reveals the double benefit of the local feature aggregation method proposed in Section 5 with respect to performance and query efficiency. Merging the binary signatures reduces the expanded query size and has a positive impact on complexity, as quantitatively measured in Table 3: The aggregation step reduces by about one order of magnitude the size of the enriched query, which becomes comparable to that of the initial query.

In addition, Table 3 also shows that this step significantly and consistently improves the performance. To demonstrate this, we have compared HQE (*i.e.*, with aggregation) to a method which issues the expanded query defined by Equation 5, *i.e.*, prior to aggregation. As already discussed in Section 5, our interpretation is that aggregating binary signatures filters out noisy features and removes the redundant features at the same time. Merging the features derived from the reliable images can also give rise to multiple matches per descriptor, yet those are effectively handled by aggregation [32].

Building	GT	HE	HQE		HQE-SP	
		mAP	$\lvert\mathcal{L}_\mathcal{Q}\rvert$	mAP	$\lvert\mathcal{L}_\mathcal{Q}\rvert$	mAP
All Souls	183	78.2	47.0	94.6	44.8	97.3
Ashmolean	56	63.7	10.9	76.1	9.9	80.8
Balliol	30	72.7	15.8	81.0	8.0	82.1
Bodleian	54	66.4	33.4	94.5	19.8	86.9
Christ Church	211	74.9	39.5	75.7	45.1	90.7
Cornmarket	22	69.5	9.6	64.9	6.4	71.4
Hertford	55	87.7	41.5	95.0	43.5	98.3
Keble	18	93.0	9.5	96.5	7.6	99.5
Magdalen	157	29.9	15.6	36.5	8.8	48.6
Pitt Rivers	16	100.0	9.7	99.7	7.0	100.0
Radcliffe	569	93.9	97.1	98.5	96.0	98.7

Table 4: Oxford5k dataset: Summary of the number of ground truth images, the number of reliable images and the performance for HE, HQE and HQE with spatial matching. We report the average value of $|\mathcal{L}_\mathcal{Q}|$ per building, *i.e.*, the number of automatically detected reliable images in the short-list of 100 top-ranked ones.

# features		12.53M	21.92M	27.59M
method	MA	mAP		
BOVW		54.9	58.7	55.2
HE		74.2	78.6	78.3
HQE		81.0	84.8	84.4
HQE-SP		85.3	88.1	88.5
HQE-SP	×	88.0	**89.4**	89.3

Table 5: More features: Performance comparison on Oxford5k using lower detector threshold values, *i.e.*, larger sets of local features. Binary signatures of 128 bits are used.

Detailed performance on Oxford5k. Table 4 presents some detailed performances and statistics we have collected on Oxford5k for HE and HQE. Our selection strategy for reliable images, even without spatial matching, does not suffer from the variability of the number of true similar images in the database, with an exception on *Cornmarket*, where HQE without spatial matching selects a few false positives as reliable images. Also observe that HQE notably outperforms HQE-SP for the *Bodleian* queries. It is because HQE-SP is stricter and does not select enough reliable images. This suggests that a weaker spatial matching model [37] could offer a good compromise to select these images.

More features. All our experiments are conducted with features extracted using the default threshold for the Hessian-Affine detector [35] to allow for a direct comparison with the literature. Using a lower threshold for the "cornerness" value produces a larger set of features. It might be useful for image matching but might also add noisy features and therefore arbitrary matches.

Table 5 investigates the impact of cornerness on both our methods and existing BOVW and HE baselines. With the default threshold, the software produces a total number of 12.53M features on Oxford5k. By using two smaller thresholds, we produced two other sets of features comprising 21.92M and 27.59M features, respectively. Table 5 shows that BOVW's performance increases with the medium-sized set, but its performance drops with the larger one. In contrast, HE benefits from having more features. The performance of the two larger sets is comparable, which suggests that HE better handle noisy matches in a better way and can use more features. As a consequence, HQE performs in a similar way. The performance increases up to **mAP=89.4** for HQE with geometry and MA, which is a large improvement over the state of the art.

Method	SP	MA	Oxford5k	Paris6k	Oxford105k
Perdoch [35]	×		78.4	N/A	72.8
Perdoch [35]	×	×	82.2	N/A	77.2
Mikulik [38]	×	×	**84.9**	**82.4**	**79.5**
Chum [16]	×		82.7	80.5	76.7
Arandjelovic [17]	×		80.9	76.5	72.2
HQE			80.7±*0.9*	80.2±*0.2*	76.6±*1.1*
HQE-SP	×		83.7±*0.7*	80.0±*0.2*	79.4±*0.6*
HQE		×	83.0±*0.9*	80.6±*0.2*	79.0±*1.0*
HQE-SP	×	×	86.8±*0.3*	81.5±*0.3*	82.6±*0.4*
HQE 128bits			81.0±*0.5*	81.5±*0.2*	76.9±*0.6*
HQE-SP 128bits	×		85.3±*0.4*	81.3±*0.3*	80.8±*0.5*
HQE 128bits		×	83.8±*0.3*	82.8±*0.1*	80.4±*0.5*
HQE-SP 128bits	×	×	**88.0**±*0.3*	**82.8**±*0.2*	**84.0**±*0.2*

Table 6: Performance comparison with state-of-the-art methods on Oxford5k, Paris6k and Oxford105k. The standard deviation is obtained from 5 measurements.

8.3. Comparison with the state of the art

Oxford5k, Paris6k and Oxford105k. Table 6 compares the QE proposed method with previously published results on the same datasets. For a fair comparison, we have included the scores of QE methods that use the same local feature detector [35] as input and learned the vocabulary on an independent dataset. In this table, we also include the scores for our method when using 128-bit signatures for HE, which are better at the cost of higher memory usage and a slightly larger complexity.

Interestingly, even without spatial matching, our method outperforms all methods in Oxford105k and Paris6k dataset. HQE-SP outperforms them in all three datasets. All of the compared methods rely on spatial matching to verify similar images and expand the initial query. Moreover, the work of Mikulik *et al.* [38] requires a costly off-line phase and assigns the descriptors with a very large vocabulary of 16M, thereby impacting the overall efficiency.

To our knowledge, the performance of our method is the best reported to date on Oxford5k, Paris6k and Oxford105k, when learning the vocabulary on an independent dataset (89.1 was reported [17] by learning it on the Oxford5k comprising the relevant images). In addition, all these techniques are likely to be complementary, as they consider orthogonal aspects to improve the performance.

UKB [6] is a dataset with few corresponding images per query (4, including the query image). QE techniques are therefore

Jégou [10]	Jégou [21]	Qin [18]	HE-MA	HQE-MA
3.64	**3.68**	**3.67**	3.59	**3.67**

Table 7: UKB: comparison with state-of-the-art methods.

Method	HE	HQE	HQE-SP
SA	30 ms	79 ms	731 ms
MA	76 ms	204 ms	955 ms

Table 8: Average query times for the HE baseline and our technique with and without spatial matching, measured on Oxford105k when using a single processor core. These timings do not include the description part (extracting and quantizing the SIFT descriptors), which does not depend on database size.

not expected to perform well, and accordingly we are not aware of any competitive result reported with a QE method on this dataset. For this set only, we reduce the short-list of images selected in the short-list to reflect the expected result set. Table 7 shows that HQE improves the performance significantly compared to the HE baseline and is therefore effective even with few relevant images. It performs similar to other state-of-the-art techniques that perform well on this dataset. Note that these best techniques all require to cross-match (off-line) the whole image collection with itself, which may be infeasible on a large scale (quadratic complexity).

8.4. Complexity: timings and query size

First, note that the initial query includes a binary signature per local feature and several features can be assigned to the same visual word for a given image, especially with MA. In addition, the expanded query set, as defined before aggregation, is much larger as several images contribute to it with their reliable features, as previously shown in Table 3. Thanks to HQE, only one binary signature per visual word is kept. This favorably impacts the complexity of the enriched query in terms of the number of signatures. On average, the total number of features increases only by a small factor after aggregation, to be compared with queries which are one order of magnitude larger for other QE techniques.

Table 8 reports the average search times when querying Oxford105k. They have been measured on a single core desktop machine (3.2 Ghz). The spatial matching has been estimated by an external software and is included in the query time, unlike the SIFT extraction and quantization times. As expected, the search times are competitive for HQE without geometry, even when MA is used. As a reference, best time reported for QE with spatial matching [35] is 509ms on Oxford105k on a 4×3.0Ghz machine. In addition, the cost of assignment to visual words is much smaller for our method with 65k visual words compared to the one of their method which needs up to 1M visual words to obtain optimal performance.

9. Conclusion

This paper makes several contributions related to visual query expansion. First, we introduce a query expansion method which is effective without using any geometry. While the general belief is that spatial verification is required to select the relevant images used to build the augmented query, exploiting the Hamming Embedding technique with a stringent selection rule and an aggregation strategy, we already achieve state-of-the-art performance. This method has a low complexity. We then show that combining our Hamming query expansion with geometry further improves the results and significantly outperform the state of the art.

In future work, we will investigate how to incorporate weak spatial matching models [37, 23] in our query expansion method, in order to find a compromise between a costly spatial verification or not using geometry at all.

10. Acknowledgments

This work was done in the context of the Project Fire-ID, supported by the Agence Nationale de la Recherche (ANR-12-CORD-0016). We also thank the reviewers for their valuable comments and effort to help us improving the manuscript.

References

[1] J. Sivic, A. Zisserman, Video Google: A text retrieval approach to object matching in videos, in: International Conference on Computer Vision, 2003. 1, 3

[2] C. Schmid, R. Mohr, Local grayvalue invariants for image retrieval, IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (5) (1997) 530–534. 1

[3] D. Lowe, Distinctive image features from scale-invariant keypoints, International Journal of Computer Vision 60 (2) (2004) 91–110. 1, 3

[4] G. Salton, C. Buckley, Term-weighting approaches in automatic text retrieval, Information Processing & Management 24 (5) (1988) 513–523. 1

[5] J. Zobel, A. Moffat, K. Ramamohanarao, Inverted files versus signature files for text indexing, ACM Trans. Database Systems 23 (4) (1998) 453–490. 1

[6] D. Nistér, H. Stewénius, Scalable recognition with a vocabulary tree, in: IEEE Conference on Computer Vision and Pattern Recognition, 2006, pp. 2161–2168. 1, 2, 7, 8, 10

[7] J. Philbin, O. Chum, M. Isard, J. Sivic, A. Zisserman, Object retrieval with large vocabularies and fast spatial matching, in: IEEE Conference on Computer Vision and Pattern Recognition, 2007. 1, 4, 6, 8

[8] S. M. Katz, Distribution of content words and phrases in text and language modeling, Natural Language Engineering 2 (1996) 15–59. 1

[9] K. Lund, C. Burgess, Producing high-dimensional semantic spaces from lexical co-occurrence, Behavior Research Methods, Instruments & Computers 28 (1996) 203–208. 1

[10] H. Jégou, M. Douze, C. Schmid, On the burstiness of visual elements, in: IEEE Conference on Computer Vision and Pattern Recognition, 2009. 1, 3, 9, 11

[11] O. Chum, J. Matas, Unsupervised discovery of co-occurrences in sparse high dimensional data, in: IEEE Conference on Computer Vision and Pattern Recognition, 2010. 1

[12] R. G. Cinbis, J. Verbeek, C. Schmid, Image categorization using fisher kernels of non-iid image models, in: IEEE Conference on Computer Vision and Pattern Recognition, 2012. 1

[13] C. D. Manning, P. Raghavan, H. Schütze, Introduction to Information Retrieval, Cambridge University Press, 2008, Ch. Relevance feedback & query expansion. 1

[14] O. Chum, J. Philbin, J. Sivic, M. Isard, A. Zisserman, Total recall: Automatic query expansion with a generative feature model for object retrieval, in: International Conference on Computer Vision, 2007. 1, 2, 4, 6

[15] A. Joly, O. Buisson, Logo retrieval with a contrario visual query expansion, in: ACM International conference on Multimedia, 2009. 1, 2, 4

[16] O. Chum, A. Mikulik, M. Perdoch, J. Matas, Total recall II: Query expansion revisited, in: IEEE Conference on Computer Vision and Pattern Recognition, 2011. 1, 2, 4, 10

[17] R. Arandjelovic, A. Zisserman, Three things everyone should know to improve object retrieval, in: IEEE Conference on Computer Vision and Pattern Recognition, 2012. 1, 2, 4, 5, 6, 7, 10

[18] D. Qin, S. Gammeter, L. Bossard, T. Quack, L. V. Gool, Hello neighbor: Accurate object retrieval with k-reciprocal nearest neighbors, in: IEEE Conference on Computer Vision and Pattern Recognition, 2011. 1, 2, 3, 8, 11

[19] X. Shen, Z. Lin, J. Brandt, S. Avidan, Y. Wu, Object retrieval and localization with spatially-constrained similarity measure and k-nn re-ranking, in: IEEE Conference on Computer Vision and Pattern Recognition, 2012. 1, 3

[20] S. Zhang, M. Yang, T. Cour, K. Yu, D. N. Metaxas, Query specific fusion for image retrieval, in: European Conference on Computer Vision, 2012. 1, 3

[21] H. Jégou, C. Schmid, H. Harzallah, J. Verbeek, Accurate image search using the contextual dissimilarity measure, IEEE Transactions on Pattern Analysis and Machine Intelligence 32 (1) (2010) 2–11. 1, 2, 3, 8, 11

[22] J. Philbin, O. Chum, M. Isard, J. Sivic, A. Zisserman, Lost in quantization: Improving particular object retrieval in large scale image databases, in: IEEE Conference on Computer Vision and Pattern Recognition, 2008. 1, 3, 8

[23] H. Jégou, M. Douze, C. Schmid, Improving bag-of-features for large scale image search, International Journal of Computer Vision 87 (3) (2010) 316–336. 1, 3, 4, 7, 8, 11

[24] P. Turcot, D. Lowe, Better matching with fewer features: the selection of useful features in large database recognition problems, in: International Conference on Computer Vision Workshop, 2009. 2

[25] O. Chum, J. Matas, Large-scale discovery of spatially related images, IEEE Transactions on Pattern Analysis and Machine Intelligence 32 (2) (2010) 371–377. 3

[26] Y.-H. Kuo, K.-T. Chen, C.-H. Chiang, W. H. Hsu, Query expansion for hash-based image object retrieval, in: ACM International conference on Multimedia, 2009. 3

[27] X. Li, W. Zhou, J. Tang, Q. Tian, Query expansion enhancement by fast binary matching, in: ACM International conference on Multimedia, 2012. 3

[28] K. Mikolajczyk, C. schmid, Scale and affine invariant interest point detectors, International Journal of Computer Vision 60 (1) (2004) 63–86. 3

[29] F. Perronnin, J.Sánchez, T. Mensink, Improving the Fisher kernel for large-scale image classification, in: European Conference on Computer Vision, 2010. 3

[30] F. Perronnin, C. R. Dance, Fisher kernels on visual vocabularies for image categorization, in: IEEE Conference on Computer Vision and Pattern Recognition, 2007. 6

[31] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, C. Schmid, Aggregating local descriptors into compact codes, in: IEEE Transactions on Pattern Analysis and Machine Intelligence, 2012. 6

[32] G. Tolias, Y. Avrithis, H. Jégou, To aggregate or not to aggregate: Selective match kernels for image search, in: International Conference on Computer Vision, 2013. 6, 9

[33] M. Jain, R. Benmokhtar, P. Gros, H. Jégou, Hamming embedding similarity-based image classification, in: ICMR, 2012. 6, 7

[34] H. Jégou, O. Chum, Negative evidences and co-occurences in image retrieval: The benefit of PCA and whitening, in: European Conference on Computer Vision, 2012. 7

[35] M. Perdoch, O. Chum, J. Matas, Efficient representation of local geometry for large scale object retrieval, in: IEEE Conference on Computer Vision and Pattern Recognition, 2009. 8, 10, 11

[36] T. Sattler, T. Weyand, B. Leibe, L. Kobbelt, Image retrieval for image-based localization revisited, in: IEEE Conference on Computer Vision and Pattern Recognition, 2012. 8

[37] G. Tolias, Y. Avrithis, Speeded-up, relaxed spatial matching, in: International Conference on Computer Vision, 2011. 10, 11

[38] A. Mikulík, M. Perdoch, O. Chum, J. Matas, Learning a fine vocabulary, in: European Conference on Computer Vision, 2010. 10

XXI Unsupervised object discovery for instance recognition

Title: Unsupervised object discovery for instance recognition
Authors: O. Siméoni, A. Iscen, G. Tolias, Y. Avrithis, O. Chum
Published at: WACV 2018

Unsupervised object discovery for instance recognition

Oriane Siméoni[1] Ahmet Iscen[2] Giorgos Tolias[2] Yannis Avrithis[1] Ondřej Chum[2]
[1]Inria Rennes [2]VRG, FEE, CTU in Prague
{oriane.simeoni,ioannis.avrithis}@inria.fr
{ahmet.iscen,giorgos.tolias,chum}@cmp.felk.cvut.cz

Abstract

Severe background clutter is challenging in many computer vision tasks, including large-scale image retrieval. Global descriptors, that are popular due to their memory and search efficiency, are especially prone to corruption by such a clutter. Eliminating the impact of the clutter on the image descriptor increases the chance of retrieving relevant images and prevents topic drift due to actually retrieving the clutter in the case of query expansion. In this work, we propose a novel salient region detection method. It captures, in an unsupervised manner, patterns that are both discriminative and common in the dataset. Saliency is based on a centrality measure of a nearest neighbor graph constructed from regional CNN representations of dataset images. The descriptors derived from the salient regions improve particular object retrieval, most noticeably in a large collections containing small objects.

1. Introduction

Particular object retrieval becomes very challenging when the object of interest is covering a small part of the image. In this case, the amount of relevant information is significantly reduced. Large objects might be partially occluded, while small objects are on a background that covers most of the image. A combination of both, occlusion and cluttered background, is not rare either. These conditions naturally arise from image acquisition and make naive approaches fail, including global template matching or semi-robust template matching [25].

Ideally, image descriptors should be extracted only from the relevant part of the image, suppressing the irrelevant clutter and occlusions. In this paper, we attempt to determine the regions containing the relevant information, as shown in Figure 1, in a fully unsupervised manner.

Methods based on robust matching of *hand-crafted local features* are naturally insensitive to occlusion and background clutter. The locality of the features allows to match small parts of images in regions containing the object of interest, while the incorrect matches are typically removed by robust geometric consistency check [28]. Methods based on efficient matching of vector-quantized local-feature descriptors were introduced in context of image retrieval by Sivic and Zisserman [35].

Figure 1. The saliency map (right) computed for an input image (left) based on common-structure analysis on *Instre* dataset. Background clutter and objects not relevant for this dataset are automatically removed. The image is represented only by the region detected on the saliency map.

Retrieval methods based on descriptors extracted by *convolutional neural networks* (CNNs) have become popular because they combine good precision and recall, efficiency of the search, and reasonable memory footprint [5, 30]. Deep neural networks are capable of learning, to some extent, what information in the image is relevant, which results in a good performance even with global descriptors [39, 4, 17]. However, if the signal to noise ratio is low, *e.g.* the object is relatively small, multiple objects are present, *etc.*, the global CNN descriptors fail [12, 11].

A class of methods inspired by *object detection* have recently emerged. Instead of attempting to match the whole image to the query, the problem is changed to finding a rectangular region in the image that best matches the query [39, 31]. An inefficient search by sliding window is intractable for large collections of images. The exhaustive enumeration is approximated by similarity evaluation on a number of pre-selected regions. The regions are either selected geometrically to cover the whole image at different scales, as in R-MAC [39], or by considering the content by object or region proposal methods [31, 36, 8].

Another direction of suppressing irrelevant content is saliency detection [17, 24]. For each image, a saliency map, that captures more general region shapes compared to (a small set of) rectangles, is first estimated. The contribution of each pixel (or region) is then proportional to the saliency of that location.

In this work we introduce a very simple pooling scheme that inherits the properties of both saliency detection and region based pooling and that, like all previous approaches, is applied to each image in the database *independently*. In addition, we investigate the use of the resulting regional representation for automatic, offline object discovery and suppression of background clutter, which considers the image collection *as a whole*. Unlike previous approaches, we do this in an unsupervised way. As a consequence, our representation takes two saliency detection steps into account. One that acts per image and depends solely on its content and another that considers the image collection as a whole and captures frequently appearing objects.

In both cases, we derive a *global* representation that outperforms comparable state-of-the-art methods in retrieving small objects on standard benchmarks, while the memory footprint and online cost is only a fraction compared to more powerful *regional* representations [30, 12]. Moreover, we show that our representation benefits significantly from *query expansion* methods.

Section 2 discusses our contributions against related work. Section 3 describes our methodology including our pooling scheme in Section 3.3 and our object discovery approach in Section 3.8. We present experimental results in Section 4 and draw conclusions in Section 5.

2. Related work

Local features and geometric matching offer an attractive way for retrieval systems to handle occlusions, clutter, and small objects [35, 28, 13]. One of their drawbacks is high query complexity and large storage cost; an image is typically represented by several thousands features. Many methods attempt to decrease the amount of indexed features by removing background clutter while maintaining the relevant information. The selection procedure is either applied independently per image or considers an image collection as a whole. Common examples of the former case are bursty feature detection [33], symmetry detection [38] or use of semantic segmentation [1, 26]. The methods of the second category, are scalable enough to jointly process the whole collection and perform feature selection by the following assumption. A feature that repeats over multiple instances of the same object in the dataset is likely to appear in novel views of the object too. Representative cases are common object discovery [40, 37], co-occurrence detection [6], or methods using GPS information [7, 19].

The work by Turcot and Lowe [40] performs pairwise spatial verification on hand-crafted local features across all images and only indexes verified features. With an additional off-line cost, the on-line stage is sped up and the memory footprint is reduced. However, unique views of objects are not verified and thus discarded. In this work, we address a similar selection problem based on more powerful CNN-based representation rather than local features.

Recent advances on deep learning [3, 39, 17, 9, 29] dispense with the large memory footprint by using global descriptors and cast the problem of instance search as Euclidean nearest neighbor search. Nevertheless, background clutter and occlusion are better handled by regional representation. Regional descriptors significantly increase the performance when they are indexed independently [30, 12] but this comes at a prohibited memory and computational cost for large scale scenarios. Region Proposal Networks (RPN) are applied either off-the-shelf [31] or after fine-tuning [36] for instance search. The RPNs reduce the number of regions per image only to the order of tens. Our work focuses on aggregating regional representation that keeps the complexity low but we rather detect regions around salient objects and objects that frequently appear in the dataset. Jimenez *et al*. [15] construct saliency maps and perform region detection to construct global image vectors, as we also do. However, they employ generic object detectors trained on ImageNet and this makes the method not applicable with fine-tuned networks which provide the best performance. The Hessian-affine detector is used on CNN activations to detect repeatable regions [14]. The major benefit in this work, though, comes from second order pooling and higher dimensional descriptors.

Saliency maps are another way to handle clutter and occlusions. Once more, there exist both examples of computation in an unsupervised manner [17, 20] or learned [24, 16] and applied per image afterwards. Our approach generates saliency maps in a fully unsupervised way that capture both salient objects on single images but also repeating objects appearing in a particular image collection.

3. Method

Like [40], our objective is to remove transient and non-distinctive objects as in Figure 1 and rather focus on objects appearing frequently in a dataset. Beginning with the activation map of a convolutional layer in a CNN, one would need access to a local representation to automatically discover such objects. On the other hand, knowing what these objects are would help forming a local representation by selecting regions depicting them, which appears to be a chicken-and-egg problem. Without an initial region selection, we risk "discovering" uninformative but frequently appearing "stuff"-like patches, for instance sky.

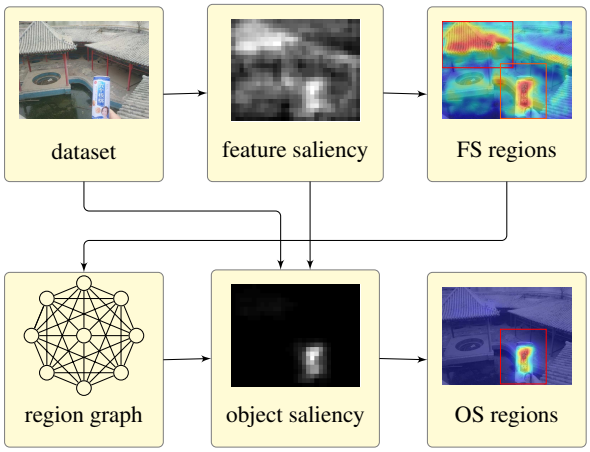


Figure 2. Overview of our offline unsupervised process. On the top row, CNN activations of dataset images are used to extract a *feature saliency* map, on which a set of regions is detected. On the bottom row, a *centrality* measure is obtained per region from a region k-NN graph. Using this measure, a dense *object saliency* map is formed from the original CNN activations and the feature saliency. This map is focusing on objects automatically discovered in the dataset, with background clutter removed. Finally, another set of regions is detected on the object saliency map to extract descriptors and represent the dataset for retrieval.

3.1. Overview

Fortunately, it is possible to make an initial selection based on CNN activations alone, without any training and without bounding box annotations. As described in Section 3.3, the mechanism is inspired by CroW [17] and Grad-CAM [32] and generates a *feature saliency* map. This initiates our offline analysis illustrated in Figure 2. A small set of rectangular regions is detected per image from this map as discussed in Section 3.4. This first round of detection is applied independently per image and depends only on its content.

Each region in the dataset is associated to a feature saliency score and a visual descriptor, pooled from the activation map of the corresponding image, as discussed in Section 3.5. It is now possible to compute a *centrality* score per region, representing the "significance" of each region in the dataset. This is based on a region k-NN graph and is discussed in Sections 3.6 and 3.7.

Now, given a new image, we can infer the "significance" of every region from its nearest neighbors in the graph, yielding a dense *object saliency* map as discussed in Section 3.8. This is a regression problem and we suggest a non-parametric k-NN solution. Finally, we detect a small set of rectangular regions on this saliency map and extract a global descriptor to represent dataset images for retrieval, as discussed in Section 3.9. This second detection procedure takes into account all salient and repeating objects appearing in the dataset.

The entire process is fully unsupervised and only assumes on the-shelf networks trained on a classification or retrieval task without bounding box annotations.

3.2. Representation

We represent the activation map of a convolutional layer as a non-negative 3d tensor $A \in \mathbb{R}^{h \times w \times c}$ where h, w are the spatial resolution (height, width) and c is the number of feature channels. The set of valid spatial positions is $P := [h] \times [w]$[1] and the set of all rectangles with vertices in P is denoted by $\mathcal{R}$. By A_{pj} we represent an element of A at position $p \in P$ and channel $j \in [c]$. By $A_{\bullet j} \in \mathbb{R}^{h \times w}$ we denote the 2d feature map of A corresponding to channel $j \in [c]$. By $A_{p\bullet} \in \mathbb{R}^c$ we denote the vector containing all feature channels at position $p \in P$.

3.3. Feature saliency

Inspired by *cross-dimensional weighting and pooling* (CroW) [17] and *class activation mapping* (CAM) [44], we construct a 2d saliency map of an image based on a convolution activation of that image alone. Following CroW, we compute an idf-like weight per channel $\mathbf{b} \in \mathbb{R}^c$ with elements

$$b_j = \log\left(\frac{(\mathbf{a}+\epsilon)^\top \mathbf{1}}{a_j + \epsilon}\right) \tag{1}$$

for $j \in [c]$, where $\mathbf{a} := \frac{1}{wh}\sum_{p \in P} \mathbb{1}[A_{p\bullet}] \in \mathbb{R}^c$ is the average number of nonzero elements per channel. We then compute a weighted sum over channels

$$F := \sum_{j \in [c]} b_j A_{\bullet j} \tag{2}$$

Finally, we obtain the 2d *feature saliency* (FS) map $\hat{F} \in \mathbb{R}^{h \times w}$ by normalizing F according to [17]. Contrary to CroW, we use the feature channel weights when computing the 2d spatial weights, amplifying channels with sparse activation. This order of summation is the same as in CAM. However, we are working with channel weights obtained by a sparsity property on any convolutional layer, without any assumption on the network topology. CAM on the other hand, assumes global average pooling followed by a fully connected layer mapping channels to classes and uses the parameters of this layer to obtain a saliency map per class.

3.4. Region detection

We are given a 2d saliency map S, which can be either the feature saliency described in section 3.3 or the object saliency described in Section 3.8. We use an *expanding Gaussian mixture* (EGM) model [2] to detect a number of

[1] Here, $[i]$ is the set $\{1, \ldots, i\}$ for $i \in \mathbb{N}$.

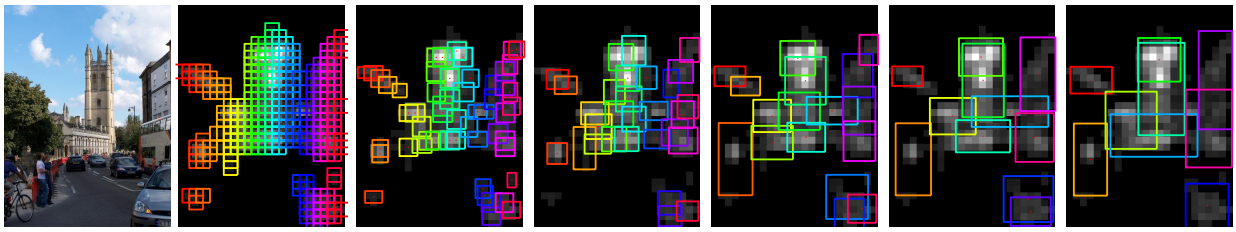


Figure 3. Evolution of regions during EGM iterations on the feature saliency map of an image of *Magdalen tower* from Oxford buildings dataset, shown on the left. Below each image we display the iteration i and the number of regions m.

salient rectangular regions. This is a variant of expectation-maximization (EM) that iteratively performs local averaging (E- and M-steps) interleaved with a selection process (P-step) similar to non-maximum suppression (NMS). In doing so, it dynamically estimates the number of regions.

The original algorithm applies to point sets and isotropic Gaussian components. Here we extend it to functions, considering that a saliency map is a function $S : P \to \mathbb{R}$. We use it to fit a number of components, each modeling a rectangular region in 2d coordinate space. We also extend it to a diagonal covariance model, so that a rectangle is modeled by an axis-aligned ellipse.

In particular, given 2d saliency map $S \in \mathbb{R}^{h \times w}$, we represent it as a set of Gaussian functions $s_i : \mathbb{R}^2 \to \mathbb{R}$ with

$$s_i(\mathbf{x}) := S_{p_i} \mathcal{N}(\mathbf{x} | p_i, \sigma I_2) \tag{3}$$

for $i \in [\ell]$, $\mathbf{x} \in \mathbb{R}^2$ where $\mathcal{N}$ is the normal density, $\ell = |P|$ is the number of positions and we represent P as $\{p_1, \ldots, p_\ell\}$. Here, σ is a *scale* parameter that determines how coarse or fine the region representation will be for the given saliency map. Similarly, we represent components as Gaussian functions $q_k : \mathbb{R}^2 \to \mathbb{R}$ with

$$q_k(\mathbf{x}) := \pi_k \mathcal{N}(\mathbf{x} | \mu_k, \Sigma_k) \tag{4}$$

for $k \in [m]$, $\mathbf{x} \in \mathbb{R}^2$, where m is the number of components and $\pi_k \in \mathbb{R}$, $\mu_k \in \mathbb{R}^2$ and $\Sigma_k \in \mathbb{R}^{2 \times 2}$ are the mixing coefficient, mean and diagonal covariance matrix respectively of component k. Means represent region centers, while the (inverse) eigenvalues of covariance matrices represent heights and widths. We initialize components as $q_k \leftarrow s_k$ for $k \in [m]$, with $m \leftarrow \ell$. In the *expectation* (E)-step, we compute the *responsibility*

$$\gamma_{ik} \leftarrow \frac{\langle s_i, q_k \rangle}{\sum_{j \in [m]} \langle s_i, q_j \rangle} \tag{5}$$

of component $k \in [m]$ for sample $i \in [\ell]$, where $\langle f, g \rangle$ is the L^2 inner product of square-integrable functions $f, g : \mathbb{R}^d \to \mathbb{R}$, computed in closed form for Gaussian functions [2]. In the *maximization* (M)-step, we update parameters as

$$\pi_k \leftarrow \frac{\ell_k}{\ell} \tag{6}$$

$$\mu_k \leftarrow \frac{1}{\ell_k} \sum_{i=1}^{n} \gamma_{ik} p_i \tag{7}$$

$$\Sigma_k \leftarrow \frac{1}{\ell_k} \sum_{i=1}^{n} \gamma_{ik} \operatorname{diag}(p_i - \mu_k)^{\circ 2} \tag{8}$$

where $\ell_k := \sum_{i=1}^{n} \gamma_{ik}$ is the effective number of points assigned to component k and $X^{\circ 2} := X \circ X$ is the Hadamard product power for a vector or matrix X.

Finally, in the *purge* (P)-step, similarly to NMS, we process components in descending order of mixing coefficient and we decide whether to keep a component or not depending on its overlap with the collection of previously kept components. Overlap is measured by a generalized responsibility function similar to (5), and again inner products are given in closed form [2]. This means that the number of components m is potentially reducing at each iteration.

Figure 3 shows how regions are formed during EGM iterations, starting from one small region centered on each spatial position. We get 4 clean regions on the ground truth building, as well as 6 regions on background objects, which, although less salient, cannot be removed based on the feature saliency alone.

3.5. Region pooling

Given a rectangular region $R \in \mathcal{R}$ of an image with feature saliency map $\hat{F} \in \mathbb{R}^{h \times w}$, we associate to it *feature saliency* $f := \mu_{\hat{F}}(R) \in \mathbb{R}$, where

$$\mu_{\hat{F}}(R) := \frac{1}{|R|} \sum_{p \in R} \hat{F}_p \tag{9}$$

is the average of 2d map $\hat{F}$ over R.

In addition, given the activation map $A \in \mathbb{R}^{h \times w \times c}$ of the same image, it is standard practice that a descriptor is obtained by pooling over R, for instance sum [4], weighted

sum [17] or max [3, 39] pooling. We adopt the latter choice to extract descriptor $\mathbf{z} := m_A(R) \in \mathbb{R}^c$, where

$$m_A(R) := \max_{q \in R} A_{q\bullet} \tag{10}$$

is the maximum of 3d tensor A over R along the spatial dimensions. This has been the basis of fine-tuning in [29, 8].

A particular set of regions, uniformly sampled on a grid at different scales, is referred to as *regional maximum activation of convolutions* (R-MAC) [39]. Global description, referred to as MAC, is a special case where there is a single region $R = P$. In contrast, we detect a set of regions based on saliency maps in this work.

Finally, we follow [29] in performing supervised whitening of the descriptors by simultaneous diagonalization [22]. In particular, given vector $\mathbf{z} \in \mathbb{R}^c$, we ℓ^2-normalize, center, whiten, PCA-project and renormalize to generate the *region descriptor* $\mathbf{v} := w(\mathbf{z}) \in \mathbb{R}^d$ for region R. Function $w : \mathbb{R}^c \to \mathbb{R}^d$ represents this pipeline entirely.

3.6. Graph construction

Given an image dataset, we assume here a set of regions $\{R_1, \ldots, R_n\}$ are detected from the saliency maps (Section 3.4), a *feature saliency* vector $\mathbf{f} := (f_1, \ldots, f_n) \in \mathbb{R}^n$ is computed with the corresponding average saliency per region in (9), and a set of descriptors $V := \{\mathbf{v}_1, \ldots, \mathbf{v}_n\} \subset \mathbb{R}^d$ are extracted from the activation maps, whitened and normalized per region (Section 3.5).

Based on the above information, we construct a k-NN graph on those regions in order to compute a global *centrality* score per region as discussed in Section 3.7, which enables us to form an *object saliency* map on a new image, described in Section 3.8. Approximate techniques for k-NN graph construction [**?**] can be used to handle large-scale databases.

We construct a weighted undirected graph having the set of descriptors V as vertices. Following [12], the edge weights are defined according to *mutual k-nearest neighbors* (NN) in the descriptor space. In particular, given descriptors $\mathbf{v}, \mathbf{u} \in \mathbb{R}^d$, we measure their *similarity* by $s(\mathbf{v}, \mathbf{u}) = (\mathbf{v}^\top \mathbf{u})^\beta$, where exponent $\beta > 0$ is a parameter. We define the sparse symmetric nonnegative *adjacency matrix* $W \in \mathbb{R}^{n \times n}$ with elements w_{ij} being $s(\mathbf{v}_i, \mathbf{v}_j)$ if $\mathbf{v}_i, \mathbf{v}_j$ are mutual k-NN in V and zero otherwise.

We define the $n \times n$ *degree matrix* $D := \mathrm{diag}(W\mathbf{1})$ where $\mathbf{1} \in \mathbb{R}^n$ is the all-ones vector, and the *symmetrically normalized adjacency matrix*

$$\mathcal{W} := D^{-1/2} W D^{-1/2}, \tag{11}$$

with the convention $0/0 = 0$. Following [12, 11], we define the $n \times n$ matrices $L_\alpha := (D - \alpha W)/(1 - \alpha)$ and

$$\mathcal{L}_\alpha := D^{-1/2} L_\alpha D^{-1/2} = (I - \alpha \mathcal{W})/(1 - \alpha), \tag{12}$$

where $\alpha \in [0, 1)$. Both are positive-definite [12, 11].

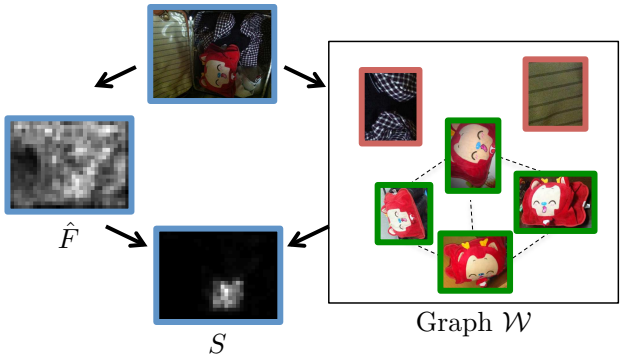


Figure 4. Computing the *object saliency* map S of an image from Instre dataset (top), as defined in (14). For each patch, its neighbors in the graph (right) are found. Common patterns with high centrality in green outline, outliers with low centrality in red. S (bottom) then focuses on patches similar to common patterns and combines with feature saliency $\hat{F}$ (left).

3.7. Graph centrality

With the above definitions in place, the objective is to compute a vector $\mathbf{g} \in \mathbb{R}^n$ where each element g_i represents the significance of vertex $\mathbf{v}_i$ in the graph, for $i \in [n]$. We define this *centrality vector* as the solution $\mathbf{g}^* \in \mathbb{R}^n$ of the linear system

$$\mathcal{L}_\alpha \mathbf{g} = \mathbf{1}. \tag{13}$$

As in [12], we solve this system by the *conjugate gradients* (CG) [23] method. Any method would be equally appropriate because this is computed just once offline.

The solution $\mathbf{g}^*$ is a *graph centrality* measure [21], and in particular, *Katz centrality* [18]. Centrality is a global measure of significance of vertices in a graph, and PageRank [27] is maybe the most well-known. In fact, Katz centrality was introduced as such a global measure before being adapted by *boundary condition* $\mathbf{y}$ to measure relevance to individual vertices by Hubbell [10]. This work has a long history before being rediscovered *e.g.* by [27, 45], as summarized in the study of *spectral ranking* [41].

3.8. Saliency map construction

Given the region descriptor set V, the region saliency vector $\mathbf{f}$ and the associated centrality vector $\mathbf{g}^*$ of an entire dataset, the problem is to construct a new saliency map $S \in \mathbb{R}^{h \times w}$ for an image in the dataset. The image is represented by its activation map $A \in \mathbb{R}^{h \times w \times c}$. Since this saliency is based on regions or patterns appearing frequently in the dataset, which are commonly associated to repeating objects, we call it *object saliency* (OS).

We compute S by a sliding window iteration over each position $p \in P$. The saliency value S_p at p is found as a linear combination of the centrality values of the nearest neighbors in V of a patch centered at p. In particular, we consider a square patch R_p of side a centered at p. We compute the vector $\mathbf{u}_p := w(m_A(R_p)) \in \mathbb{R}^d$ by max-pooling

over R_p, whitening and normalizing as discussed in Section 3.5. If N_p is the set of indices of the k-NN of $\mathbf{u}_p$ in V, we compute S_p as

$$S_p := \hat{F}_p^{\Theta} \sum_{i \in N_p} s(\mathbf{v}_i, \mathbf{u}_p) f_i^{\theta} g_i^*. \quad (14)$$

That is, each neighboring region descriptor $\mathbf{v}_i$ is weighted by its similarity to patch descriptor $\mathbf{u}_p$, its feature saliency f_i and its centrality g_i^*, while the entire sum is scaled by the feature saliency $\hat{F}_p$ at the current position p of the image being considered. Exponents Θ and θ control the relative importance of feature saliency of the current image and neighbors, respectively, compared to centrality. The object saliency computation is illustrated in Figure 4. Looking at the input image and is feature saliency map $\hat{F}$ alone, it is not evident which is the object of interest and which is clutter. This is only found by discovering other instances of the same object in the dataset, as represented by the graph.

3.9. Representation

The object saliency map S highlights patterns that appear frequently in the dataset, with the background clutter removed. It is only natural then to apply the same method described in Section 3.4 to this map in order to detect a small number of regions per image. Unlike the regions detected from the feature saliency map $\hat{F}$, these new regions are more likely to appear in a new image. For the purpose of evaluation, we investigate both saliency maps.

For each region R detected from a saliency map ($\hat{F}$ or S) in a dataset image with activation map A, we apply max pooling and ℓ^2-normalization. All descriptors are then summed and the resulting descriptor is whitened with $w : \mathbb{R}^c \to \mathbb{R}^d$ as described in Section 3.5. The difference here is that we apply whitening on the aggregated vector and not separately per region. This is the same representation as R-MAC evaluated in [29] and both yield a global image representation in $\mathbb{R}^d$, but here the regions are detected in the saliency map rather than being uniformly distributed.

Pooling based on saliency is in fact the idea explored in CroW [17], but here we follow the nonlinear two-level pooling of R-MAC (max followed by sum) rather than the one-level sum of CroW. This is more powerful and has also been the basis of fine-tuning in [8].

4. Experiments

We apply the proposed representation on image retrieval. In particular, we have two variants of our method that both use the region detection described in Section 3.4. The saliency map which the detection is performed on is different in each case. FS.EGM uses the feature saliency map described in Section 3.3, and OS.EGM uses the object saliency map described in Section 3.4. The former is image specific, while the latter both image and database specific.

4.1. Experimental setup

Test sets. We evaluate on Oxford Buildings [28] and the more recently introduced Instre [42] dataset. Instre contains around 27k images of small objects in cluttered scenes while objects appear with different variations, such as rotation, occlusion and scale changes, making it a challenging case. We use the evaluation protocol introduced in [12] for Instre. We add 100k distractors from Flickr [28] to Oxford5k to perform experiments at larger scale. We refer to it as Oxford105k. Search performance in all datasets is measured with mAP.

Image Representation. We represent each image by global image representation as described in Section 3.9. This reduces image similarity to cosine, which is common practice [39]. Feature extraction is performed with the VGG network [34] that is fine-tuned specifically for image retrieval [29]. Supervised whitening [29] is used for post-processing. The same network is additionally used to compare against two baselines. First, MAC global descriptor, which is obtained by global max pooling and the descriptor that the network is directly optimized for [29]. Second, the baseline approach (*Uniform*), which refers to regional max pooling for regions that are uniformly sampled at 3 scales, as in R-MAC [39]. Our variants are different in that regions are detected from salient and repeating objects, while aggregation and whitening is identical. Detection is applied to dataset images only, while we use the provided bounding boxes on the query side.

Implementation Details. To simplify region detection, each saliency map is masked above threshold τ and element-wise raised to exponent ρ before detection, which removes the weakest regions and increases the contrast between foreground and background objects. We set $\rho = 1$, $\tau = 0.2$ and scale parameter $\sigma = 1$ before any parameter tuning is performed. We determine OS parameters Θ, θ in (14) by visual inspection of OS and set $\Theta = 2$, $\theta = 3$ throughout our experiments. We perform our experiments on a 16-core Intel Xeon 2.00GHz CPU. It takes 36s to create the graph on Instre, while centrality computation takes negligible amount of time. It takes 0.02s for FS computation and detection per image, while 0.23s in the case of OS.

4.2. Parameter tuning

In this section, we show the impact of FS.EGM and OS.EGM detection parameters on the retrieval performance. We tune the parameters on Oxford5k when using diffusion [12]. The remaining experiments evaluate the proposed representation with the chosen parameters on Instre and Oxford105k as well.

Feature saliency detection is evaluated first by FS.EGM, while we do not compute object saliency and OS.EGM yet.

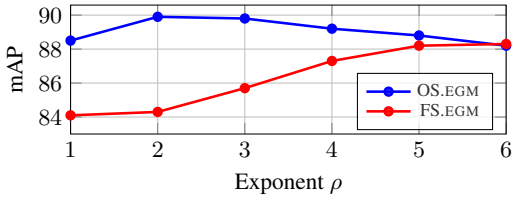


Figure 5. mAP on Oxford5k versus saliency exponent ρ for FS.EGM and OS.EGM.

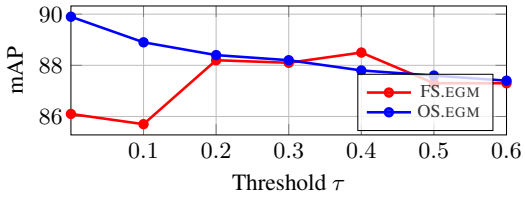


Figure 6. mAP on Oxford5k versus threshold τ for FS.EGM and OS.EGM.

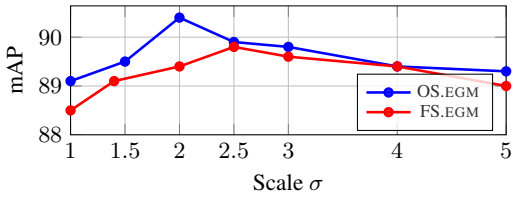


Figure 7. mAP on Oxford5k versus EGM scale parameter σ for FS.EGM and OS.EGM.

Figure 5 shows the effect of ρ, which controls the contrast of the the saliency map. We observe that large ρ is needed to remove as much clutter as possible from the noisy FS activations. We set $\rho = 5$ for the rest of our experiments. Figure 6 shows the effect of threshold τ, which is another selectivity parameter. We set $\tau = 0.4$. Scale σ is used during EGM sampling as explained in Section 3.4. Its impact in performance is shown in Figure 7. Setting $\sigma = 2.5$ results in good performance and regions that are large enough for FS.EGM.

Object saliency detection is then evaluated once the feature saliency parameters are fixed, and EGM detection is applied on the new saliency map. We observe that OS behaves quite differently to FS, because foreground objects are much cleaner. The impact of parameters σ and ρ is shown in Figures 5 and 7 respectively. It is remarkable that a much lower exponent is needed in this case. We choose $\rho = 2$ and $\sigma = 2$. Finally, we fix $\tau = 0$ for OS, as the saliency maps obtained with OS are exactly zero at background regions. The effect is shown in Figure 6.

4.3. Evaluation of saliency maps

We exploit the fact that Instre dataset comes with bounding box annotation for all database images. We use the ground truth information to quantitatively evaluate the saliency maps. We define *precision* as the sum of saliency over ground truth regions, normalized by the sum over the entire image, and we measure it for FS and OS as shown in Figure 8. High precision means that a saliency map is well aligned to the ground truth bounding boxes. Given that these bounding boxes are not used anywhere, the improvement that OS offers is impressive. Visual examples for saliency maps and detections for FS.EGM and OS.EGM are shown in Figure 9. In all cases, OS is cleaner and focuses on objects that FS cannot discriminate.

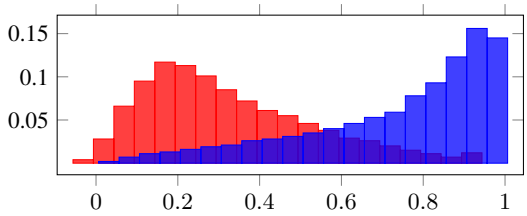


Figure 8. Histogram of saliency precision for FS and OS maps measured on all images of Instre.

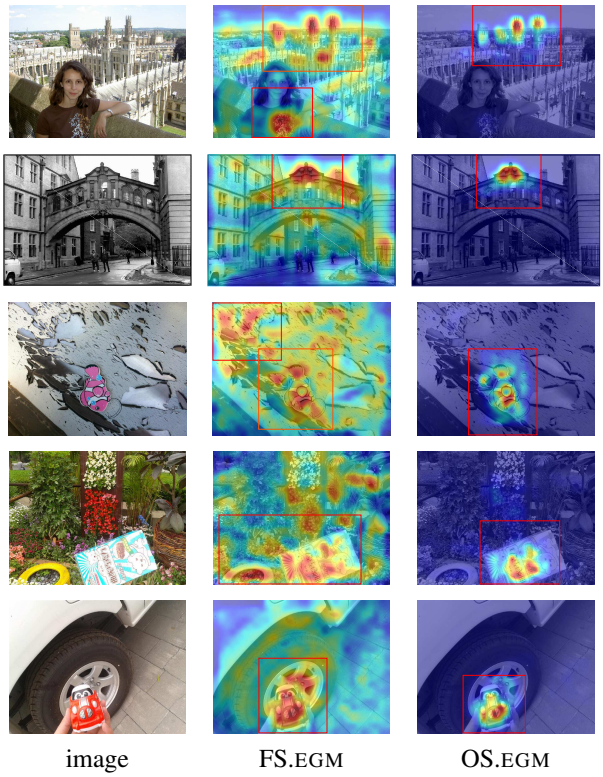


Figure 9. Examples of images from Oxford5k (first 2 rows) and Instre (last 3 rows) datasets, along with smoothed FS and OS maps superimposed on the images and regions detected by EGM, in red.

4.4. Comparison to other methods

We compare our methods to the standard practice of uniform region sampling (Uniform) as in R-MAC and global max pooling (MAC). We additionally propose a variant of OS.EGM, where further uniform region sampling at 2 scales

Method	QE	Instre	Oxford	Oxford105k
MAC	-	48.5	79.7	73.9
Uniform [39]	-	47.7	77.7	70.1
FS.EGM *	-	48.4	77.5	70.2
OS.EGM *	-	50.1	79.6	71.8
OS.EGM-$\triangle^\star$	-	53.7	79.8	71.4
MAC	✓	71.8	87.4	86.0
Uniform [39]	✓	70.3	85.7	82.7
FS.EGM *	✓	71.2	89.8	87.9
OS.EGM *	✓	72.7	**90.4**	**88.0**
OS.EGM-$\triangle^\star$	✓	**75.4**	90.1	84.3

Table 1. mAP comparison of our methods marked with * against baselines on all tested datasets. QE refers to query expansion by diffusion [12].

is performed within each detected region. We refer to this as OS.EGM-$\triangle$. All methods are tested with k-NN search and global diffusion [12], which is a method for query expansion or manifold search and is known to significantly improve performance. Results are given in Table 1.

FS.EGM improves performance compared to uniform sampling by focusing on salient objects. However, salient objects are not necessarily relevant for the particular dataset. This is what OS.EGM captures and boosts the search performance, especially on Instre. On all datasets, MAC is better than uniform sampling (R-MAC). This is because the network used [29] is directly fine-tuned to optimize MAC. However, when using diffusion, we outperform it on all datasets. This can be explained by the fact that diffusion boosts any items that are similar to the top-ranking ones according to the original similarity [12], so it is essential that these items are reliable. A global descriptor is affected by clutter in general. By contrast, our representation is global yet clutter-free. Our improvements are larger on Instre, which is more challenging due to small objects and severe background clutter. This is exactly where our detection is essential. Most Instre images are also quite different than the building images which the network is fine-tuned on. This is probably why our representations outperform MAC even without diffusion on this dataset.

There are several other previous approaches that deal with region detection or saliency masks, which are not directly comparable, so they are not included in Table 1. Nevertheless, we outperform their reported results. Salvador *et al*. [31] use the off-the-shelf VGG and fine-tune RPN in the test set. Without using query expansion, they obtain 71.0 in Oxford5k. Similarly, Jimenez *et al*. [15] learn class weights and apply them on the activation maps of off-the-shelf VGG and achieve 73.6 in Oxford5k. Song *et al*. [36] train on different datasets, and achieve 78.3 in Oxford5k. The results obtained by learning a saliency mask are not comparable since spatial verification with local features is always applied in the end [24]. Finally, Zheng *et al*. [43] achieve 83.4

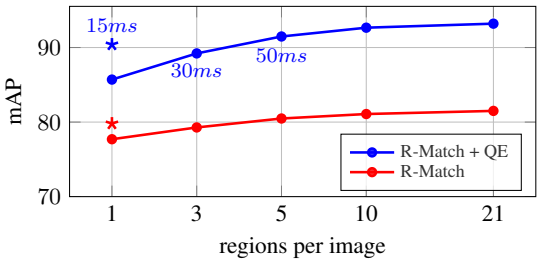


Figure 10. mAP comparison of our global OS.EGM ($\star$) to R-Match with uniformly sampled regional descriptors, with and without diffusion on Oxford5k. Text labels refer to query time.

with regional representation on Oxford5k. They employ both CNN and local features, while we only rely on CNN and much more compact representation. Finally, no work other than [12] evaluates on Instre which is rather challenging due to small objects.

Region cross-matching methods [30] represent an image with multiple vectors, sacrificing memory footprint and complexity for accuracy. In particular, the memory is linear in the number of regions, while the complexity is quadratic. We compare our global representation with region cross-matching (R-Match) and regional diffusion [12] in Figure 10. Different numbers of regions are obtained by GMM reduction, exactly as in [12].

Compared to regional descriptors, we require about 4 times less memory to achieve the same performance. The runtime complexity gain is in the order of 4^2, which holds for the case of R-Match and also for the first part of diffusion where Euclidean nearest neighbors are found. The diffusion complexity is O(m), where m is the number of non-zero entries of the graph. We found that m is 3.7 times smaller in our case and our measurements of actual query timings agree with this ratio.

5. Conclusions

We propose a region detection approach that is dataset specific but requires no supervision. It captures not only salient objects by considering each image individually but also frequently appearing ones by considering the dataset as a whole. As a result, we avoid separate indexing of regional descriptors and construct a global descriptor by pooling over data-dependent regions, which performs well under background clutter and severe occlusions. We demonstrate that this approach is effective in particular object retrieval where background clutter is a common problem.

Acknowledgments The authors were supported by the MSMT LL1303 ERC-CZ grant. The Tesla K40 used for this research was donated by the NVIDIA Corporation.

References

[1] R. Arandjelović and A. Zisserman. Visual vocabulary with a semantic twist. In *ACCV*, 2014. 2

[2] Y. Avrithis and Y. Kalantidis. Approximate gaussian mixtures for large scale vocabularies. In *ECCV*, pages 15–28. Springer, 2012. 3, 4

[3] H. Azizpour, A. S. Razavian, J. Sullivan, A. Maki, and S. Carlsson. From generic to specific deep representations for visual recognition. *arXiv preprint arXiv:1406.5774*, 2014. 2, 5

[4] A. Babenko and V. Lempitsky. Aggregating deep convolutional features for image retrieval. In *ICCV*, 2015. 1, 4

[5] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky. Neural codes for image retrieval. In *ECCV*, 2014. 1

[6] O. Chum and J. Matas. Unsupervised discovery of co-occurrence in sparse high dimensional data. In *CVPR*, June 2010. 2

[7] S. Gammeter, L. Bossard, T. Quack, and L. V. Gool. I know what you did last summer: Object-level auto-annotation of holiday snaps. In *ICCV*, 2009. 2

[8] A. Gordo, J. Almazan, J. Revaud, and D. Larlus. Deep image retrieval: Learning global representations for image search. *ECCV*, 2016. 1, 5, 6

[9] A. Gordo, J. Almazan, J. Revaud, and D. Larlus. End-to-end learning of deep visual representations for image retrieval. *arXiv preprint arXiv:1610.07940*, 2016. 2

[10] C. H. Hubbell. An input-output approach to clique identification. *Sociometry*, 1965. 5

[11] A. Iscen, Y. Avrithis, G. Tolias, T. Furon, and O. Chum. Fast spectral ranking for similarity search. *arXiv*, 2017. 1, 5

[12] A. Iscen, G. Tolias, Y. Avrithis, T. Furon, and O. Chum. Efficient diffusion on region manifolds: Recovering small objects with compact cnn representations. In *CVPR*, 2017. 1, 2, 5, 6, 8

[13] H. Jégou, M. Douze, and C. Schmid. Improving bag-of-features for large scale image search. *IJCV*, 87(3):316–336, February 2010. 2

[14] D.-j. Jeong, S. Choo, W. Seo, and N. I. Cho. Regional deep feature aggregation for image retrieval. In *ICASSP*, 2017. 2

[15] A. Jimenez, J. M. Alvarez, and X. Giro-i Nieto. Class-weighted convolutional features for visual instance search. *BMVC*, 2017. 2, 8

[16] H. Jin Kim, E. Dunn, and J.-M. Frahm. Learned contextual feature reweighting for image geo-localization. In *CVPR*, 2017. 2

[17] Y. Kalantidis, C. Mellina, and S. Osindero. Cross-dimensional weighting for aggregated deep convolutional features. In *arXiv*, 2015. 1, 2, 3, 5, 6

[18] L. Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, 1953. 5

[19] J. Knopp, J. Sivic, and T. Pajdla. Avoiding confusing features in place recognition. In *ECCV*, 2010. 2

[20] Z. Laskar and J. Kannala. Context aware query image representation for particular object retrieval. In *Scandinavian Conference on Image Analysis*, 2017. 2

[21] N. MEJ. *Networks: an introduction*. Oxford University Press, Oxford, 2010. 5

[22] K. Mikolajczyk and J. Matas. Improving descriptors for fast tree matching by optimal linear projection. In *CVPR*, 2007. 5

[23] J. Nocedal and S. Wright. *Numerical optimization*. Springer, 2006. 5

[24] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han. Large-scale image retrieval with attentive deep local features. In *arXiv*, 2016. 2, 8

[25] A. Oliva and A. Torralba. Building the gist of a scene: The role of global image features in recognition. *Progress in brain research*, 155:23–36, 2006. 1

[26] D. Omercevic, R. Perko, A. T. Targhi, J.-O. Eklundh, and A. Leonardis. Vegetation segmentation for boosting performance of mser feature detector. In *Computer Vision Winter Workshop*, 2008. 2

[27] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: bringing order to the web. 1999. 5

[28] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, June 2007. 1, 2, 6

[29] F. Radenović, G. Tolias, and O. Chum. CNN image retrieval learns from bow: Unsupervised fine-tuning with hard examples. *ECCV*, 2016. 2, 5, 6, 8

[30] A. S. Razavian, J. Sullivan, S. Carlsson, and A. Maki. Visual instance retrieval with deep convolutional networks. *ITE Transactions on Media Technology and Applications*, 4:251–258, 2016. 1, 2, 8

[31] A. Salvador, X. Giró-i Nieto, F. Marqués, and S. Satoh. Faster r-cnn features for instance search. In *CVPRW*, 2016. 1, 2, 8

[32] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra. Grad-CAM: Why did you say that? visual explanations from deep networks via gradient-based localization. *arXiv preprint arXiv:1610.02391*, 2016. 3

[33] M. Shi, Y. Avrithis, and H. Jegou. Early burst detection for memory-efficient image retrieval. In *CVPR*, 2015. 2

[34] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2014. 6

[35] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *ICCV*, 2003. 1, 2

[36] J. Song, T. He, L. Gao, X. Xu, and H. T. Shen. Deep region hashing for efficient large-scale instance search from images. In *arXiv*, 2017. 1, 2, 8

[37] G. Tolias, Y. Avrithis, and H. Jégou. Image search with selective match kernels: aggregation across single and multiple images. *IJCV*, 2016. 2

[38] G. Tolias, Y. Kalantidis, and Y. Avrithis. Symcity: Feature selection by symmetry for large scale image retrieval. In *ACM Multimedia*, 2012. 2

[39] G. Tolias, R. Sicre, and H. Jégou. Particular object retrieval with integral max-pooling of cnn activations. *ICLR*, 2016. 1, 2, 5, 6, 8

[40] P. Turcot and D. G. Lowe. Better matching with fewer features: The selection of useful features in large database recognition problems. In *ICCVW*, 2009. 2

[41] S. Vigna. Spectral ranking. *arXiv preprint arXiv:0912.0238*, 2009. 5

[42] S. Wang and S. Jiang. Instre: a new benchmark for instance-level object retrieval and recognition. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 11:37, 2015. 6

[43] L. Zheng, S. Wang, J. Wang, and Q. Tian. Accurate image search with multi-scale contextual evidences. *IJCV*, 120(1):1–13, 2016. 8

[44] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *cvpr*, June 2016. 3

[45] D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Schölkopf. Ranking on data manifolds. In *NIPS*, 2003. 5

XXII Graph-based particular object discovery

Title: Graph-based particular object discovery
Authors: O. Siméoni, A. Iscen, G. Tolias, Y. Avrithis, O. Chum
Published at: Machine Vision and Applications 2019

Machine Vision and Applications (2019) 30:243–254
https://doi.org/10.1007/s00138-019-01005-z

SPECIAL ISSUE PAPER

Graph-based particular object discovery

Oriane Siméoni[1] · Ahmet Iscen[2] · Giorgos Tolias[2] · Yannis Avrithis[1] · Ondřej Chum[2]

Received: 1 September 2018 / Accepted: 2 January 2019 / Published online: 8 February 2019

Abstract
Severe background clutter is challenging in many computer vision tasks, including large-scale image retrieval. Global descriptors, which are popular due to their memory and search efficiency, are especially prone to corruption by such a clutter. Eliminating the impact of the clutter on the image descriptor increases the chance of retrieving relevant images and prevents topic drift due to actually retrieving the clutter in the case of query expansion. In this work, we propose a novel salient region detection method. It captures, in an unsupervised manner, patterns that are both discriminative and common in the dataset. Saliency is based on a centrality measure of a nearest neighbor graph constructed from regional CNN representations of dataset images. The proposed method exploits recent CNN architectures trained for object retrieval to construct the image representation from the salient regions. We improve particular object retrieval on challenging datasets containing small objects.

Keywords Image retrieval · Unsupervised object discovery · Image saliency

1 Introduction

Particular object retrieval becomes very challenging when the object of interest is covering a small part of the image. In this case, the amount of relevant information is significantly reduced. Large objects might be partially occluded, while small objects are on a background that covers most of the image. A combination of both, occlusion and cluttered background, is not rare either. These conditions naturally arise from image acquisition and make naive approaches fail, including global template matching or semi-robust template matching [31].

Ahmet Iscen
ahmet.iscen@cmp.felk.cvut.cz

Oriane Siméoni
oriane.simeni@inria.fr

Giorgos Tolias
giorgos.tolias@cmp.felk.cvut.cz

Yannis Avrithis
ioannis.avrithis@inria.fr

Ondřej Chum
chum@cmp.felk.cvut.cz

[1] Inria, Univ Rennes, CNRS, IRISA, Rennes, France

[2] VRG, FEE, CTU in Prague, Prague, Czech Republic

Ideally, image descriptors should be extracted only from the relevant part of the image, suppressing the irrelevant clutter and occlusions. In this paper, we attempt to determine the regions containing the relevant information, as shown in Fig. 1, in a fully unsupervised manner.

Methods based on robust matching of *handcrafted local features* are naturally insensitive to occlusion and background clutter. The locality of the features allows to match small parts of images in regions containing the object of interest, while the incorrect matches are typically removed by robust geometric consistency check [35]. Methods based on efficient matching of vector-quantized local-feature descriptors were introduced in context of image retrieval by Sivic and Zisserman [47].

Retrieval methods based on descriptors extracted by *convolutional neural networks* (CNNs) have become popular because they combine good precision and recall, efficiency of the search and reasonable memory footprint [5,40]. Deep neural networks are capable of learning, to some extent, what information in the image is relevant, which results in a good performance even with global descriptors [4,19,51]. However, if the signal-to-noise ratio is low, e.g., the object is relatively small, multiple objects are present, etc., the global CNN descriptors fail [14,15].

A class of methods inspired by *object detection* have recently emerged. Instead of attempting to match the whole image to the query, the problem is changed to finding a

Fig. 1 The saliency map (right) computed for an input image (left) based on common-structure analysis on *Instre* dataset. Background clutter and objects not relevant for this dataset are automatically removed. The image is represented only by the region detected on the saliency map

rectangular region in the image that best matches the query [42,51]. An inefficient search by sliding window is intractable for large collections of images. The exhaustive enumeration is approximated by similarity evaluation on a number of pre-selected regions. The regions are either selected geometrically to cover the whole image at different scales, as in R-MAC [51], or by considering the content by object or region proposal methods [11,42,48].

Another direction of suppressing irrelevant content is saliency detection [19,30]. For each image, a saliency map, which captures more general region shapes compared to (a small set of) rectangles, is first estimated. The contribution of each pixel (or region) is then proportional to the saliency of that location.

In this work, we introduce a very simple pooling scheme that inherits the properties of both saliency detection and region-based pooling and that, like all previous approaches, is applied to each image in the database *independently*. In addition, we investigate the use of the resulting regional representation for automatic, off-line object discovery and suppression of background clutter, which considers the image collection *as a whole*. Unlike previous approaches, we do this in an unsupervised way. As a consequence, our representation takes two saliency detection steps into account. One that acts per image and depends solely on its content and another that considers the image collection as a whole and captures frequently appearing objects.

In both cases, we derive a *global* representation that outperforms comparable state-of-the-art methods in retrieving small objects on standard benchmarks, while the memory footprint and online cost is only a fraction of more powerful *regional* representations [15,40]. Moreover, we show that our representation benefits significantly from *query expansion* methods.

We make the following contributions:

1. We show that it is possible to select a set of candidate image regions based on CNN activations of off-the-shelf networks trained on retrieval tasks without bounding box annotations.
2. We obtain a global dataset-dependent "significance" of such regions via a neighborhood graph.
3. We represent the dataset by sampling and pooling CNN activations according to a dense saliency map of automatically discovered objects.
4. We thereby improve the state of the art on particular object retrieval, especially in a large-scale dataset containing small objects.

Compared to the prior version of this work [45], we make the following improvements. We apply our method to the recent GeM representation [39]. We use a multi-scale representation in saliency computation. We analyze multiple graph centrality measures and validate their impact on detection quality using bounding box annotations. Finally, we evaluate using the recently revisited and challenging $\mathcal{R}$Oxford and $\mathcal{R}$Paris benchmarks [37].

Section 2 discusses our contributions against related work. Section 3 describes our methodology including our pooling scheme in Sect. 3.3 and our object discovery approach in Sect. 3.8. We present experimental results in Sect. 4 and draw conclusions in Sect. 5.

2 Related work

Local features and geometric matching offer an attractive way for retrieval systems to handle occlusions, clutter and small objects [16,35,47]. One of their drawbacks is high query complexity and large storage cost; an image is typically represented by several thousands features. Many methods attempt to decrease the amount of indexed features by removing background clutter while maintaining the relevant information. The selection procedure is either applied independently per image or considers an image collection as a whole. Common examples of the former case are bursty feature detection [44], symmetry detection [50] or use of semantic segmentation [1,32]. The methods of the second category are scalable enough to jointly process the whole collection and perform feature selection by the following assumption. A feature that repeats over multiple instances of the same object in the dataset is likely to appear in novel views of the object too. Representative cases are common object discovery [49,52], co-occurrence detection [8] or methods using GPS information [10,23].

The work by Turcot and Lowe [52] performs pairwise spatial verification on handcrafted local features across all images and only indexes verified features. With an additional off-line cost, the online stage is speed up and the memory footprint is reduced. However, unique views of objects are not verified and thus discarded. In this work, we address

a similar selection problem based on more powerful CNN-based representation rather than local features.

Recent advances on deep learning [3,12,19,38,51] dispense with the large memory footprint by using global descriptors and cast the problem of instance search as Euclidean nearest neighbor search. Nevertheless, background clutter and occlusion are better handled by regional representation. Regional descriptors significantly increase the performance when they are indexed independently [15, 40], but this comes at a prohibited memory and computational cost for large-scale scenarios. Region proposal networks (RPNs) are applied either off-the-shelf [42] or after fine-tuning [48] for instance search. The RPNs reduce the number of regions per image only to the order of tens. Our work focuses on aggregating regional representation that keeps the complexity low, but we rather detect regions around salient objects and objects that frequently appear in the dataset.

Recent work uses CNN activation statistics to construct a saliency map in an unsupervised manner. These methods consider each dataset image independently. CRoW [19] computes spatial and channel weights based on activation maps and weighs each feature accordingly. Similarly, Laskar and Kannala [25] weighed each R-MAC region. Traditional methods are applied on top of CNN activations for the same purpose. Jeong et al. [17] used the Hessian-affine detector on activation maps to obtain repeatable regions. Pang et al. [34] used heat diffusion within an image to eliminate bursty features and keep the discriminative ones.

Another line of research trains a network to estimate the saliency map or find regions of interest. Zhu et al. [58] trained an attention layer applied over multiple scales and used it for visual place recognition. This way, the network learns to discard background clutter. Similarly, Kim and Yoon [22] trained a regional attention network to focus on important parts of an image. Jimenez et al. [18] constructed saliency maps and performed region detection to construct global image vectors, which is similar to our goal. However, they employ object detectors trained on ImageNet classes, which does not apply to networks fine-tuned for retrieval on new classes. Mohedano et al. [28] evaluated deep and non-deep saliency models in order to detect regions of interest from an image.

All aforementioned methods either act per image without supervision or learn from a collection with ground truth. By contrast, our method operates on the entire dataset jointly and at the same time is fully unsupervised.

The problem that this work is dealing with has been addressed previously in the literature but on different tasks. Common objects or regions in an image collection have been addressed in several different ways. Kim and Torralba [21] started from local features and detected region candidates using PageRank [33]. Bagon et al. [6] used *local self-similarity descriptors*, while Rubinstein et al. [41] used SIFT flow. More recently, Cho et al. [7] started from region proposals to discover dominant objects, while Kwak et al. [24] extended such a discovery process to videos.

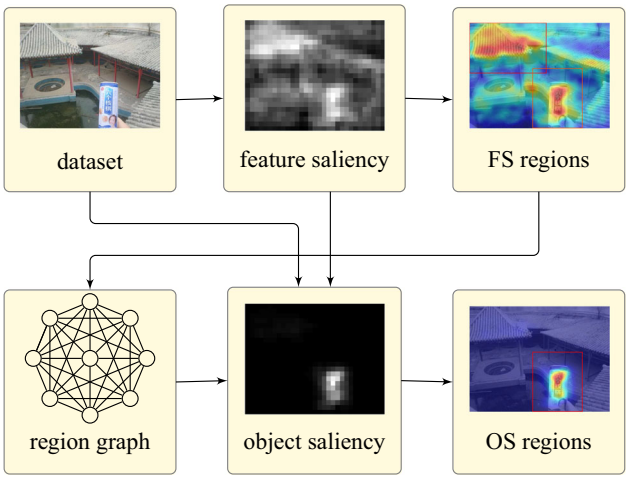


Fig. 2 Overview of our off-line unsupervised process. On the top row, CNN activations of dataset images are used to extract a *feature saliency* map, on which a set of regions is detected. On the bottom row, a *centrality* measure is obtained per region from a region k-NN graph. Using this measure, a dense *object saliency* map is formed from the original CNN activations and the feature saliency. This map is focusing on objects automatically discovered in the dataset, with background clutter removed. Finally, another set of regions is detected on the object saliency map to extract descriptors and represent the dataset for retrieval

3 Method

Like [52], our objective is to remove transient and nondistinctive objects as in Fig. 1 and rather focus on objects appearing frequently in a dataset. Beginning with the activation map of a convolutional layer in a CNN, one would need access to a local representation to automatically discover such objects. On the other hand, knowing what these objects are would help in forming a local representation by selecting regions depicting them, which appears to be a chicken-and-egg problem. Without an initial region selection, we risk "discovering" uninformative but frequently appearing "stuff"-like patches, for instance sky.

3.1 Overview

Fortunately, it is possible to make an initial selection based on CNN activations alone, without any training and without bounding box annotations. As described in Sect. 3.3, the mechanism is inspired by CroW [19] and Grad-CAM [43] and generates a *feature saliency* map. This initiates our off-line analysis illustrated in Fig. 2. A small set of rectangular regions is detected per image from this map as discussed in

Sect. 3.4. This first round of detection is applied independently per image and depends only on its content.

Each region in the dataset is associated with a feature saliency score and a visual descriptor, pooled from the activation map of the corresponding image, as discussed in Sect. 3.5. It is now possible to compute a *centrality* score per region, representing the "significance" of each region in the dataset. This is based on a region k-NN graph and is discussed in Sects. 3.6 and 3.7.

Now, given a new image, we can infer the "significance" of every region from its nearest neighbors in the graph, yielding a dense *object saliency* map as discussed in Sect. 3.8. This is a regression problem, and we suggest a nonparametric k-NN solution. Finally, we detect a small set of rectangular regions on this saliency map and extract a global descriptor to represent dataset images for retrieval, as discussed in Sect. 3.9. This second detection procedure takes into account all salient and repeating objects appearing in the dataset.

The entire process is fully unsupervised and only assumes on the-shelf networks trained on a classification or retrieval task without bounding box annotations.

3.2 Notation

We represent the activation map of a convolutional layer as a nonnegative 3d tensor $A \in \mathbb{R}^{h\times w\times c}$ where h, w are the spatial resolution (height, width) and c is the number of feature channels. The set of valid spatial positions is $P := [h] \times [w]$[1], and the set of all rectangles with vertices in P is denoted by $\mathcal{R}$. By A_{pj}, we represent an element of A at position $p \in P$ and channel $j \in [c]$. By $A_{\bullet j} \in \mathbb{R}^{h\times w}$, we denote the 2d feature map of A corresponding to channel $j \in [c]$. By $A_{p\bullet} \in \mathbb{R}^c$, we denote the vector containing all feature channels at position $p \in P$. By $\nu(\mathbf{x})$, we denote the ℓ^2-normalized vector $\mathbf{x}/\|\mathbf{x}\|_2$.

3.3 Feature saliency

Inspired by *cross-dimensional weighting and pooling* (CroW) [19] and *class activation mapping* (CAM) [56], we construct a 2d saliency map of an image based on a convolution activation of that image alone. Following CroW, we compute an idf-like weight per channel $\mathbf{b} \in \mathbb{R}^c$ with elements

$$b_j = \log\left(\frac{(\mathbf{a}+\epsilon)^\top \mathbf{1}}{a_j + \epsilon}\right) \tag{1}$$

for $j \in [c]$, where $\mathbf{a} := \frac{1}{wh}\sum_{p\in P} \mathbb{1}[A_{p\bullet}] \in \mathbb{R}^c$ is the average number of nonzero elements per channel. We then compute a weighted sum over channels

$$F := \sum_{j\in[c]} b_j A_{\bullet j} \tag{2}$$

Finally, we obtain the 2d *feature saliency* (FS) map $\hat{F} \in \mathbb{R}^{h\times w}$ by normalizing F according to [19]. Examples of feature saliency maps are presented in Sect. 4. Despite its simplicity, this kind of saliency can focus on objects of interest when the background is simple enough. It fails, however, in the presence of clutter. Contrary to CroW, we use the feature channel weights when computing the 2d spatial weights, amplifying channels with sparse activation. This order of summation is the same as in CAM. However, we are working with channel weights obtained by a sparsity property on any convolutional layer, without any assumption on the network topology. CAM, on the other hand, assumes global average pooling followed by a fully connected layer mapping channels to classes and uses the parameters of this layer to obtain a saliency map per class.

3.4 Region detection

We are given a 2d saliency map S, which can be either the feature saliency described in Sect. 3.3 or the object saliency described in Sect. 3.8. We use an *expanding Gaussian mixture* (EGM) model [2] to detect a number of salient rectangular regions. This is a variant of expectation–maximization (EM) that iteratively performs local averaging (E- and M-steps) interleaved with a selection process (P-step) similar to non-maximum suppression (NMS). In doing so, it dynamically estimates the number of regions.

The original algorithm applies to point sets and isotropic Gaussian components. Here we extend it to functions, considering that a saliency map is a function $S : P \to \mathbb{R}$. We use it to fit a number of components, each modeling a rectangular region in 2d coordinate space. We also extend it to a diagonal covariance model, so that a rectangle is modeled by an axis-aligned ellipse.

In particular, given 2d saliency map $S \in \mathbb{R}^{h\times w}$, we represent it as a set of Gaussian functions $s_i : \mathbb{R}^2 \to \mathbb{R}$ with

$$s_i(\mathbf{x}) := S_{p_i}\mathcal{N}(\mathbf{x}|p_i, \sigma I_2) \tag{3}$$

for $i \in [\ell], \mathbf{x} \in \mathbb{R}^2$ where $\mathcal{N}$ is the normal density, $\ell = |P|$ is the number of positions and we represent P as $\{p_1, \ldots, p_\ell\}$. Here, σ is a *scale* parameter that determines how coarse or fine the region representation will be for the given saliency map. Similarly, we represent components as Gaussian functions $q_k : \mathbb{R}^2 \to \mathbb{R}$ with

$$q_k(\mathbf{x}) := \pi_k\mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k) \tag{4}$$

[1] Here, $[i]$ is the set $\{1, \ldots, i\}$ for $i \in \mathbb{N}$.

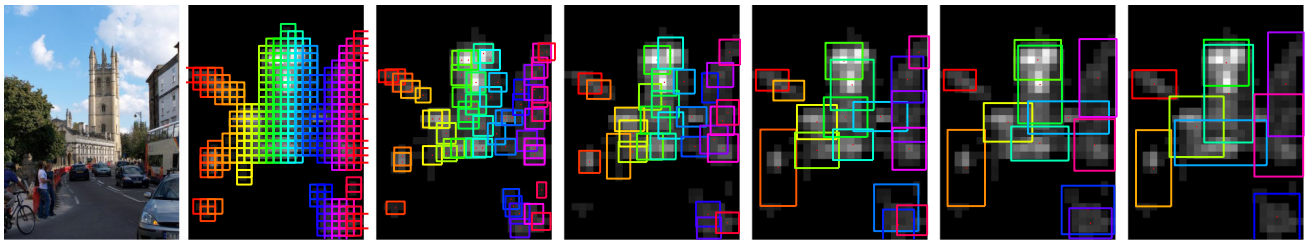


Fig. 3 Evolution of regions during EGM iterations on the feature saliency map of an image of *Magdalen Tower* from Oxford buildings dataset, shown on the left. Below each image, we display the iteration i and the number of regions m

for $k \in [m]$, $\mathbf{x} \in \mathbb{R}^2$, where m is the number of components and $\pi_k \in \mathbb{R}$, $\mu_k \in \mathbb{R}^2$ and $\Sigma_k \in \mathbb{R}^{2\times 2}$ are the mixing coefficient, mean and diagonal covariance matrix, respectively, of component k. Means represent region centers, while the (inverse) eigenvalues of covariance matrices represent heights and widths. We initialize components as $q_k \leftarrow s_k$ for $k \in [m]$, with $m \leftarrow \ell$. In the *expectation* (E)-step, we compute the *responsibility*

$$\gamma_{ik} \leftarrow \frac{\langle s_i, q_k \rangle}{\sum_{j\in[m]} \langle s_i, q_j \rangle} \tag{5}$$

of component $k \in [m]$ for sample $i \in [\ell]$, where $\langle f, g \rangle$ is the L^2 inner product of square-integrable functions $f, g : \mathbb{R}^d \to \mathbb{R}$, computed in closed form for Gaussian functions [2]. In the *maximization* (M)-step, we update parameters as

$$\pi_k \leftarrow \frac{\ell_k}{\ell} \tag{6}$$

$$\mu_k \leftarrow \frac{1}{\ell_k} \sum_{i=1}^{n} \gamma_{ik} p_i \tag{7}$$

$$\Sigma_k \leftarrow \frac{1}{\ell_k} \sum_{i=1}^{n} \gamma_{ik} \operatorname{diag}(p_i - \mu_k)^{\circ 2} \tag{8}$$

where $\ell_k := \sum_{i=1}^{n} \gamma_{ik}$ is the effective number of points assigned to component k and $X^{\circ 2} := X \circ X$ is the Hadamard product power for a vector or matrix X.

Finally, in the *purge* (P)-step, similarly to NMS, we process components in descending order of mixing coefficient and we decide whether to keep a component or not depending on its overlap with the collection of previously kept components. Overlap is measured by a generalized responsibility function similar to (5), and again inner products are given in closed form [2]. This means that the number of components m is potentially reducing at each iteration.

Figure 3 shows how regions are formed during EGM iterations, starting from one small region centered on each spatial position. We get four clean regions on the ground truth building, as well as six regions on background objects, which, although less salient, cannot be removed based on the feature saliency alone.

3.5 Region pooling and whitening

Given a rectangular region $R \in \mathcal{R}$ of an image with feature saliency map $\hat{F} \in \mathbb{R}^{h\times w}$, we associate with it *feature saliency* $f := \mu_{\hat{F}}(R) \in \mathbb{R}$, where

$$\mu_{\hat{F}}(R) := \frac{1}{|R|} \sum_{p\in R} \hat{F}_p \tag{9}$$

is the average of 2d map $\hat{F}$ over R.

In addition, given the activation map $A \in \mathbb{R}^{h\times w\times c}$ of the same image, it is standard practice that a descriptor is obtained by pooling over R, for instance sum [4], weighted sum [19], max [3,51] or generalized-mean (GeM) [39] pooling. We adopt the latter choice to extract descriptor $\mathbf{z} := m_A(R) \in \mathbb{R}^c$, where

$$m_A(R) := \left(\frac{1}{|A_{q\bullet}|} \sum_{q\in R} (A_{q\bullet})^{\omega} \right)^{\frac{1}{\omega}} \tag{10}$$

is the generalized-mean of 3d tensor A over R along the spatial dimensions and ω is a pooling parameter that is learned. This has been the basis of fine-tuning in [39] and produces a global description, referred to as GeM, in the special case where there is a single region $R = P$. In contrast, we detect a set of regions based on saliency maps in this work.

It is also standard practice to perform a sequence of post-processing steps including normalization, PCA and whitening [4,11,19,38,51]. We follow [39] in performing supervised whitening by simultaneous diagonalization [27]. In particular, given a descriptor $\mathbf{z} \in \mathbb{R}^c$, we ℓ^2-normalize, center, whiten, PCA project and re-normalize by function $w : \mathbb{R}^c \to \mathbb{R}^d$ to generate a d-dimensional descriptor:

$$w(\mathbf{z}) := \nu(U_{\mathcal{D}}^{\top} T_{\mathcal{D}}(\nu(\mathbf{z}) - \mu_{\mathcal{D}})). \tag{11}$$

Parameters $T_{\mathcal{D}}$, $U_{\mathcal{D}}$, $\mu_{\mathcal{D}}$ are trained on an independent labeled dataset $\mathcal{D} = \{Z, y\}$ where $Z = \{\mathbf{z}_1, \ldots, \mathbf{z}_n\} \subset \mathbb{R}^c$ and $y : [n]^2 \to \{0, 1\}$ maps a pair of descriptors to 1 if "positive" (similar) or 0 if "negative" (dissimilar). In particular, given set $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, define covariance matrix

$$C_l(X) := \frac{1}{|Y_l|} \sum_{(i,j)\in Y_l} (\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^\top \tag{12}$$

where $Y_l := \{(i, j) \in [n]^2 : y(i, j) = l\}$. Then, the $c \times c$ *whitening matrix* $T_\mathcal{D} := C_1(\hat{Z})^{-\frac{1}{2}}$ is the inverse square root of the intra-class covariance matrix of $\hat{Z} = \nu(Z)$. The *PCA matrix* $U_\mathcal{D}$ is the $c \times d$ matrix having as columns the top d eigenvectors of $C_0(T_\mathcal{D}\hat{Z})$, that is, the inter-class covariance matrix of the whitened counterpart of $\hat{Z}$. Finally, the *mean vector* is $\mu_\mathcal{D} := \frac{1}{n}\sum_{i\in[n]} \nu(\mathbf{z}_i)$.

3.6 Graph construction

Given an image dataset, we assume here a set of regions $\{R_1, \ldots, R_n\}$ are detected from the saliency maps (Sect. 3.4), a *feature saliency* vector $\mathbf{f} := (f_1, \ldots, f_n) \in \mathbb{R}^n$ is computed with the corresponding average saliency per region in (9), and a set of descriptors $V := \{\mathbf{v}_1, \ldots, \mathbf{v}_n\} \subset \mathbb{R}^d$ are extracted from the activation maps, whitened and normalized per region (Sect. 3.5).

Based on the above information, we construct a k-NN graph on those regions in order to compute a global *centrality* score per region as discussed in Sect. 3.7, which enables us to form an *object saliency* map on a new image, described in Sect. 3.8. Approximate techniques for k-NN graph construction [9] can be used to handle large-scale databases.

We construct a weighted undirected graph having the set of descriptors V as vertices. Following [15], the edge weights are defined according to *mutual k-nearest neighbors* (NN) in the descriptor space. In particular, given descriptors $\mathbf{v}, \mathbf{u} \in \mathbb{R}^d$, we measure their *similarity* by $s(\mathbf{v}, \mathbf{u}) = \max(0, \mathbf{v}^\top \mathbf{u})^\beta$, where exponent $\beta > 0$ is a parameter. We define the sparse symmetric nonnegative *adjacency matrix* $W \in \mathbb{R}^{n\times n}$ with elements w_{ij} being $s(\mathbf{v}_i, \mathbf{v}_j)$ if $\mathbf{v}_i, \mathbf{v}_j$ are mutual k-NN in V and zero otherwise.

We define the $n \times n$ *degree matrix* $D := \text{diag}(W\mathbf{1})$ where $\mathbf{1} \in \mathbb{R}^n$ is the all-ones vector, and the *symmetrically normalized adjacency matrix*

$$\mathcal{W} := D^{-1/2} W D^{-1/2}, \tag{13}$$

with the convention $0/0 = 0$. Following [14,15], we define the $n \times n$ matrices $L_\alpha := (D - \alpha W)/(1 - \alpha)$ and

$$\mathcal{L}_\alpha := D^{-1/2} L_\alpha D^{-1/2} = (I - \alpha\mathcal{W})/(1 - \alpha), \tag{14}$$

where $\alpha \in [0, 1)$. Both are positive definite [14,15].

3.7 Graph centrality

With the above definitions in place, the objective is to compute a vector $\mathbf{g} \in \mathbb{R}^n$ where each element g_i represents the significance of vertex $\mathbf{v}_i$ in the graph, for $i \in [n]$. We choose *graph centrality* [26] to estimate $\mathbf{g}$. Centrality is a global measure of significance of vertices in a graph. Different definitions exist, each one reflecting a different kind of vertex importance. Herein, we consider a number of centrality measures on adjacency matrix W, which we then evaluate in the experimental section.

Degree centrality is the simplest one to define and to compute. It is defined as the degree of each vertex, i.e., the diagonal of degree matrix D. Large value means that the vertex is connected to many other vertices with edges of large weight (similarity).

Eigenvector centrality, also known as eigencentrality, corresponds to the dominant eigenvector of W. Centrality value of vertex $\mathbf{v}_i$ is given by the ith element of this eigenvector. Large value means that the vertex is connected to many vertices which themselves have high centrality.

PageRank centrality is maybe the most well-known [33] centrality measure and is a variant of the eigencentrality. It is given by the dominant left eigenvector of the transition matrix $\alpha D^{-1} W + (1 - \alpha) J/n$, where J is an $n \times n$ matrix of ones. It is efficiently computed with power iteration. It models a random walk on the graph, and its score represents the probability of visiting a vertex.

Betweenness centrality reflects the number of times a vertex is part of the shortest path between any two other vertices of the graph. In contrast to all previously mentioned measures that are computed based on edge importance (similarity w_{ij}), it is computed based on an edge cost. In this case, we are using Euclidean distance $||\mathbf{v}_i - \mathbf{v}_j||$.

Closeness centrality is inversely proportional to the average length of the shortest path to all other nodes. Closeness centrality for vertex $\mathbf{v}_i$ is equal to $\sum_{\mathbf{v}_j \neq \mathbf{v}_i} \frac{n-1}{|\mathbf{v}_i \to \mathbf{v}_j|}$, where $|\mathbf{v}_i \to \mathbf{v}_j|$ is the length of the shortest path between $\mathbf{v}_i$ and $\mathbf{v}_j$. Similarly to betweenness, closeness applies to edge cost, and we use the Euclidean distance.

Katz centrality [20] is given by the solution $\mathbf{g}^* \in \mathbb{R}^n$ of the linear system

$$\mathcal{L}_\alpha \mathbf{g} = \mathbf{1}. \tag{15}$$

As in [15], we solve this system by the *conjugate gradient method* (CG) [29]. Any method would be equally appropriate because this is computed just once off-line.

It is interesting to observe that in [57], given a vertex $\mathbf{v}_i \in V$ as a *query*, the linear system $\mathcal{L}_\alpha \mathbf{x} = \mathbf{e}_i$ is considered instead, where $\mathbf{e}_i$ is the ith canonical basis vector. A random walk iteration is applied, which (slowly) converges to $\mathbf{x}^*$, where each element x^*_j represents the "similarity" of vertex $\mathbf{v}_j$ to the query $\mathbf{v}_i$. In [15], the same linear system is rather solved directly and more efficiently with CG. One may then interpret (15) as follows. If we denote by $\mathbf{x}^*_i$ the solution to $\mathcal{L}_\alpha \mathbf{x}_i = \mathbf{e}_i$ for $i \in [n]$, then the solution of (15) is $\mathbf{g}^* = \sum_{i\in[n]} \mathbf{x}^*_i$. It follows that each element g^*_j measures

the "expected similarity" of $\mathbf{v}_j$ to a query vertex of the graph for $j \in [n]$, averaged over all query vertices.

In fact, Katz centrality was introduced as such a global measure before being adapted by a *boundary condition* to measure relevance to individual vertices by Hubbell [13]. This work has a long history before being rediscovered, e.g., by [33,57], as summarized in the study of *spectral ranking* [53].

3.8 Saliency map construction

Given the region descriptor set V, the region saliency vector $\mathbf{f}$ and the associated centrality vector $\mathbf{g}^*$ of an entire dataset, the problem is to construct a new saliency map $S \in \mathbb{R}^{h \times w}$ for an image in the dataset. The image is represented by its activation map $A \in \mathbb{R}^{h \times w \times c}$. Since this saliency is based on regions or patterns appearing frequently in the dataset, which are commonly associated with repeating objects, we call it *object saliency* (OS).

We compute S by a sliding window iteration over each position $p \in P$. The saliency value S_p at p is found as a linear combination of the centrality values of the nearest neighbors in V of a patch centered at p. In particular, we consider a square patch R_p of side a centered at p. We compute the vector $\mathbf{u}_p := w(m_A(R_p)) \in \mathbb{R}^d$ by max-pooling over R_p, whitening and normalizing as discussed in Sect. 3.5. If N_p is the set of indices of the k-NN of $\mathbf{u}_p$ in V, we compute S_p as

$$S_p := \hat{F}_p^\theta \sum_{i \in N_p} s(\mathbf{v}_i, \mathbf{u}_p) f_i^\theta g_i^*. \tag{16}$$

That is, each neighboring region descriptor $\mathbf{v}_i$ is weighted by its similarity to patch descriptor $\mathbf{u}_p$, its feature saliency f_i and its centrality g_i^*, while the entire sum is scaled by the feature saliency $\hat{F}_p$ at the current position p of the image being considered. Exponent θ controls the relative importance of feature saliency of the current image and neighbors compared to centrality. The object saliency computation is illustrated in Fig. 4. Looking at the input image and is feature saliency map $\hat{F}$ alone, it is not evident which is the object of interest and which is clutter. This is only found by discovering other instances of the same object in the dataset, as represented by the graph.

3.9 Saliency-based representation

The object saliency map S highlights patterns that appear frequently in the dataset, with the background clutter removed. It is only natural then to apply the same method described in Sect. 3.4 to this map in order to detect a small number of regions per image. Unlike the regions detected from the feature saliency map $\hat{F}$, these new regions are more likely to appear in a new image. For the purpose of evaluation, we investigate both saliency maps.

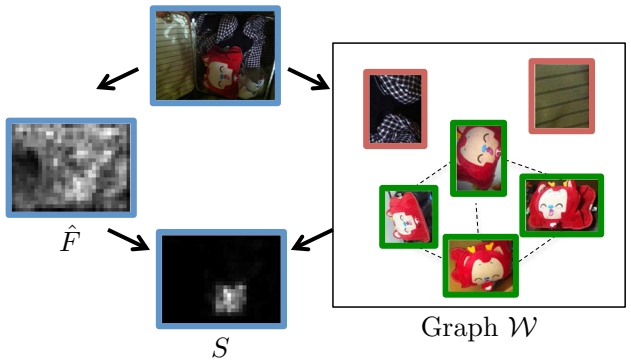


Fig. 4 Computing the *object saliency* map S of an image from Instre dataset (top), as defined in (16). For each patch, its neighbors in the graph (right) are found. Common patterns with high centrality in green outline, outliers with low centrality in red. S (bottom) then focuses on patches similar to common patterns and combines with feature saliency $\hat{F}$ (left) (color figure online)

For each region R detected from a saliency map ($\hat{F}$ or S) in a dataset image with activation map A, we apply GeM pooling and ℓ^2-normalization. All descriptors are then summed and the resulting descriptor is whitened with $w : \mathbb{R}^c \to \mathbb{R}^d$ as described in Sect. 3.5. We apply whitening on the aggregated vector and not separately per region. This follows the spirit of R-MAC [51], but performs GeM pooling instead of max, and the regions are detected in the saliency map rather than being uniformly distributed. This yields a global image representation in $\mathbb{R}^d$.

Pooling based on saliency is in fact the idea explored in CroW [19], but here we follow the nonlinear two-level pooling of R-MAC (first within a region and then over regions) rather than the one-level sum of CroW. This is more powerful and has also been the basis of fine-tuning in [12,39].

3.10 Multi-scale representation

Multi-scale descriptor extraction with CNNs is becoming standard practice [12,39]. It consists of the following steps. Re-sample the image to multiple scales, feed each scale separately to the network, obtain an ℓ_2-normalized vector per scale, sum-pool over scales, re-normalize, whiten and re-normalize. We follow the same principle, but use the representation of Sect. 3.9 per scale. Saliency maps are simply constructed independently per scale. Finally, pooling over scales is done with the generalized mean as in [39].

We additionally adopt the multi-scale concept for graph construction. Regions are detected independently per scale, and the corresponding descriptors form new vertices.

4 Experiments

We apply the proposed representation on image retrieval. In particular, we have two variants of our method that both use the region detection described in Sect. 3.4. The saliency map which the detection is performed on is different in each case. FS.EGM uses the feature saliency map described in Sect. 3.3, and OS.EGM uses the object saliency map described in Sect. 3.4. The former is image specific, while the latter is both image and database specific.

4.1 Experimental setup

Test sets. We use three image retrieval benchmarks for our experiments. We evaluate on the revisited benchmark [37] of the Oxford Buildings [35] and Paris [36] datasets. We refer to the revisited datasets as $\mathcal{R}$Oxford and $\mathcal{R}$Paris, respectively. We also use the more recently introduced Instre [54] dataset. Instre contains around 27k images of small objects in cluttered scenes, while objects appear with different variations, such as rotation, occlusion and scale changes, making it a challenging case. We use the evaluation protocol introduced in [15] for Instre. Search performance in all datasets is measured with mean average precision (mAP).

Image representation. We use the global image representation as described in Sect. 3.9. This reduces image similarity to cosine, which is common practice [51]. Feature extraction is performed with the VGG network [46] with GeM pooling that is fine-tuned for image retrieval [39]. We use supervised whitening [38,39] as described in Sect. 3.5. Compared to global GeM pooling, our variants are different in that regions are detected from salient and repeating objects, while aggregation and whitening is identical. Detection is applied to dataset images only, while we use the provided bounding boxes on the query side.

Implementation details. To simplify region detection, each saliency map is masked above threshold τ and element-wise raised to exponent ρ before detection, which removes the weakest regions and increases the contrast between foreground and background objects. We fix threshold $\tau = 0.01$ in order to remove noise from saliency maps. We set exponent $\rho = 1$ and scale parameter $\sigma = 1$ before any parameter tuning is performed. We determine OS parameter θ in (16) by visual inspection of OS and set $\theta = 3$ throughout our experiments. We perform our experiments on a 16-core Intel Xeon 2.00 GHz CPU. It takes 36 s to create the graph on Instre, while centrality computation takes negligible amount of time. Saliency computation and detection per image takes 0.02 s for FS and 0.23 s for OS.

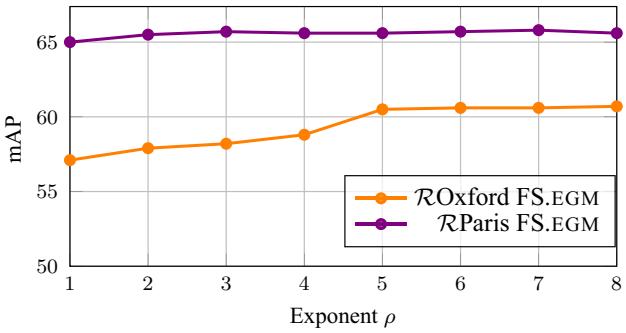


Fig. 5 mAP on $\mathcal{R}$Oxford and $\mathcal{R}$Paris versus saliency exponent ρ in FS.EGM

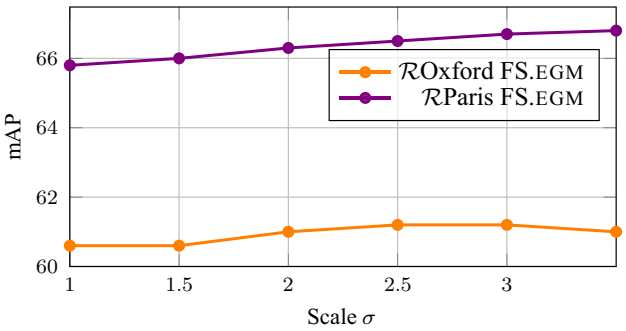


Fig. 6 mAP on $\mathcal{R}$Oxford and $\mathcal{R}$Paris versus EGM scale parameter σ in FS.EGM

4.2 Parameter tuning

In this section, we show the impact of FS.EGM and OS.EGM detection parameters on the retrieval performance. We tune the parameters on $\mathcal{R}$Oxford and $\mathcal{R}$Paris, while showing that their impact is similar on both datasets.

Feature saliency detection is evaluated first by FS.EGM, while we do not compute object saliency and OS.EGM yet. Figure 5 shows the effect of ρ on FS, which controls the contrast of the saliency map. We observe that large ρ is needed to remove as much clutter as possible from the noisy FS activations. We set $\rho = 7$ for the rest of our experiments. Scale σ is used during EGM sampling as explained in Sect. 3.4. Its impact in performance on FS is shown in Fig. 6. Setting $\sigma = 2.5$ results in good performance and regions that are large enough for FS.EGM.

Centrality measure. We use the different centrality measurements presented in Sect. 3.7, compute the OS map and evaluate the quality of saliency maps. We use Instre's bounding box annotation as ground truth and measure *saliency precision* as the sum of saliency map values inside the bounding box normalized by the total sum over the entire image. High precision means that a saliency map captures the repeating object and discards the background.

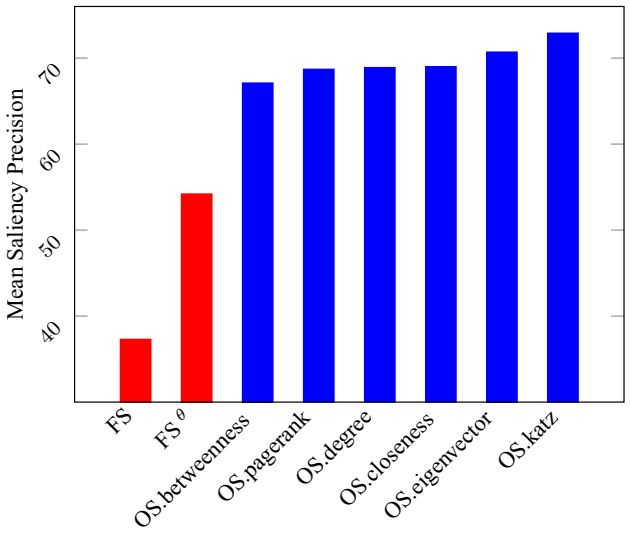


Fig. 7 Saliency precision evaluation of FS and OS created with several centralities maps measured on all images of Instre

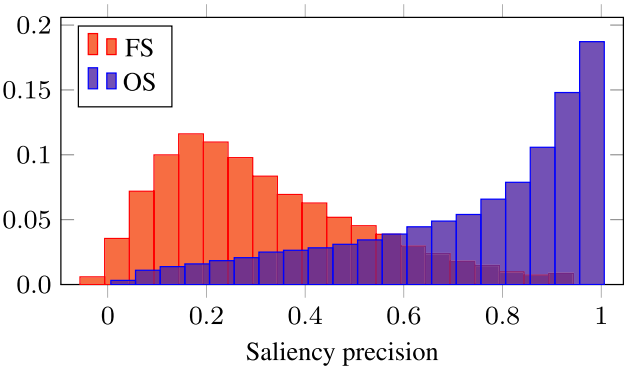


Fig. 8 Histogram of saliency precision for FS and OS maps measured on all images of Instre

We present the average saliency precision over the entire Instre dataset in Fig. 7. The comparison includes saliency maps created with FS and FS $^{\theta}$, since the latter is used in (16). Improvements brought by OS compared to FS are significant regardless of the centrality measurement, while Katz centrality gives the best precision. Figure 8 shows the distribution of precision over all Instre images for FS and OS. We use Katz centrality measurement for the rest of our experiments and simply refer to it as OS.

Object saliency detection is then evaluated for retrieval. Now, the feature saliency parameters are fixed, Katz centrality is selected, and EGM detection is applied on the new saliency map. We observe that OS behaves quite differently compared to FS, because foreground objects are much cleaner. The impact of parameters σ and ρ is shown in Figs. 9 and 10, respectively. It is remarkable that a much lower exponent is needed in this case. We choose $\rho = 2$ and $\sigma = 2.5$.

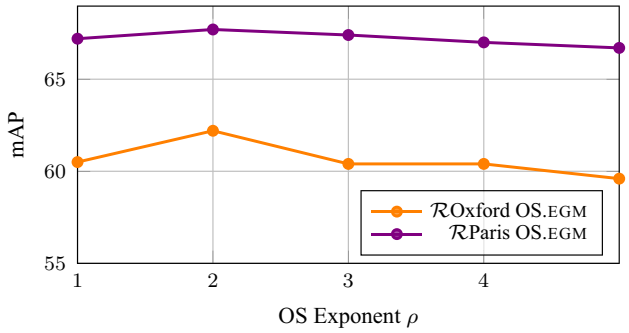


Fig. 9 mAP on $\mathcal{R}$Oxford and $\mathcal{R}$Paris versus saliency exponent ρ in OS.EGM

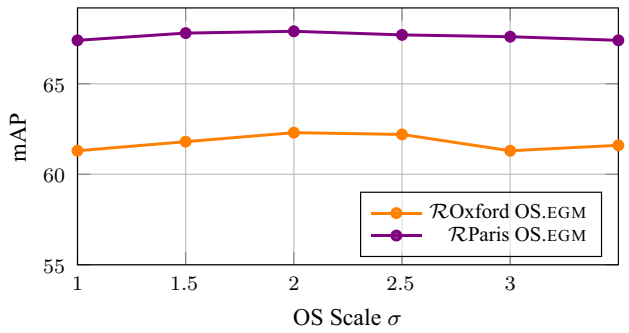


Fig. 10 mAP on $\mathcal{R}$Oxford and $\mathcal{R}$Paris versus EGM scale parameter σ in OS.EGM

Visual examples for saliency maps and detections for FS.EGM and OS.EGM are shown in Fig. 11. In all cases, OS is cleaner and focuses on objects that FS cannot discriminate.

4.3 Comparison with other methods

We compare our method against GeM descriptor [39]. We additionally evaluate the multi-scale variants for GeM, FS.EGM and OS.EGM, as described in Sect. 3.10. All methods are tested with k-NN search and global diffusion [15], which is a method for query expansion (QE) and is known to significantly improve performance. Results are given in Table 1. We report results for both medium and hard settings on $\mathcal{R}$Oxford and $\mathcal{R}$Paris.

FS.EGM does not always improve the performance, since objects captured are not necessarily relevant for the particular dataset. This is what OS.EGM captures and boosts the search performance, especially on Instre. We outperform GeM pooling in all datasets and scenarios, except for $\mathcal{R}$Paris—hard with query expansion. Note that we use the default diffusion parameters which are tuned on GeM-like descriptors. OS.EGM provides larger improvements on Instre, which is more challenging due to small objects and severe background clutter. This is exactly where our detection is essential.

There are several other approaches that deal with region detection or saliency masks. Their results are not directly

Fig. 11 Examples of images from $\mathcal{R}$Oxford (first 2 rows) and Instre (last 3 rows) datasets, along with smoothed FS and OS maps superimposed on the images and regions detected by EGM, in red (color figure online)

Table 1 mAP comparison of our method against baselines on all tested datasets. QE refers to query expansion by diffusion [15]

Method	Instre	Medium		Hard	
		$\mathcal{R}$Oxford	$\mathcal{R}$Paris	$\mathcal{R}$Oxford	$\mathcal{R}$Paris
Single scale					
GeM [39]	54.2	60.8	67.0	33.3	42.1
FS.EGM	54.6	61.2	66.5	33.4	41.7
OS.EGM	58.3	62.2	67.7	34.9	43.4
Single scale + QE					
GeM [39]	73.3	69.3	83.9	42.3	71.8
FS.EGM	72.8	71.0	84.1	42.3	71.4
OS.EGM	76.5	**72.2**	83.8	**46.5**	69.9
Multi-scale					
GeM [39]	57.0	62.0	69.3	33.7	44.3
FS.EGM	57.7	63.0	68.7	34.5	43.9
OS.EGM	61.3	64.2	69.9	35.9	46.1
Multi-scale + QE					
GeM [39]	75.0	69.3	83.9	41.1	**73.9**
FS.EGM	74.6	71.0	84.1	40.6	72.5
OS.EGM	**77.4**	69.0	**85.4**	41.9	72.3

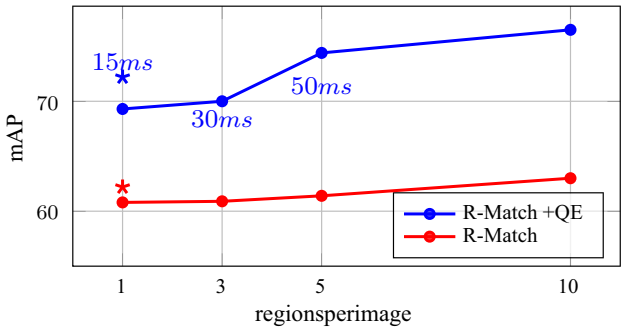


Fig. 12 mAP comparison of our global OS.EGM ($\star$) to R-Match with uniformly sampled regional descriptors, with and without diffusion on $\mathcal{R}$Oxford (medium setup). Text labels refer to query time

comparable, because they are not evaluated on $\mathcal{R}$Oxford, but Oxford5k. Therefore, they are not included in Table 1. Nevertheless, we outperform their reported results. We achieve 86.6 on Oxford5k with OS.EGM and no query expansion. Salvador et al. [42] use the off-the-shelf VGG and fine-tune RPN on the test set. Without using query expansion, they obtain 71.0 on Oxford5k. Similarly, Jimenez et al. [18] learned class weights and applied them on the activation maps of off-the-shelf VGG and achieved 73.6 on Oxford5k. Song et al. [48] trained on different datasets, and achieve 78.3 on Oxford5k. The results obtained by learning a saliency mask in [30] are not comparable since spatial verification with local features is always applied in the end. Zheng et al. [55] achieved 83.4 with regional representation on Oxford5k. They employ both CNN and local features, while we only rely on CNN and much more compact representation. Finally, according to our knowledge, [28] is the only work that evaluates on Instre, which is rather challenging due to small objects. However, their results are not directly comparable as they use a different CNN model (ResNet101).

Region cross-matching methods [40] represent an image with multiple vectors, sacrificing memory footprint and complexity for accuracy. In particular, the memory is linear in the number of regions, while the complexity is quadratic. We compare our global representation with region cross-matching (R-Match) and regional diffusion [15] in Fig. 12. We sample regions uniformly at three scales as in R-MAC [51] and apply GeM pooling separately for each region. Different numbers of regions are obtained by GMM reduction, exactly as in [15].

Compared to regional descriptors, we require about four times less memory to achieve the same performance. The runtime complexity gain is in the order of 4^2, which holds for the case of R-Match and also for the first part of diffusion where Euclidean nearest neighbors are found. The diffusion

complexity is $O(m)$, where m is the number of nonzero entries of the graph. We found that m is 3.7 times smaller in our case and our measurements of actual query timings agree with this ratio.

5 Conclusions

We propose a region detection approach that is dataset specific but requires no supervision. It captures not only salient objects by considering each image individually but also frequently appearing ones by considering the dataset as a whole. As a result, we avoid separate indexing of regional descriptors and construct a global descriptor by pooling over data-dependent regions, which performs well under background clutter and severe occlusions. We demonstrate that this approach is effective in particular object retrieval where background clutter is a common problem.

Acknowledgements This work was supported by the OP VVV funded project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics." The Tesla K40 used for this research was donated by the NVIDIA Corporation.

References

1. Arandjelović, R., Zisserman, A.: Visual vocabulary with a semantic twist. In: ACCV (2014)
2. Avrithis, Y., Kalantidis, Y.: Approximate gaussian mixtures for large scale vocabularies. In: ECCV, pp. 15–28. Springer (2012)
3. Azizpour, H., Razavian, A.S., Sullivan, J., Maki, A., Carlsson, S.: From generic to specific deep representations for visual recognition. arXiv preprint arXiv:1406.5774 (2014)
4. Babenko, A., Lempitsky, V.: Aggregating deep convolutional features for image retrieval. In: ICCV (2015)
5. Babenko, A., Slesarev, A., Chigorin, A., Lempitsky, V.: Neural codes for image retrieval. In: ECCV (2014)
6. Bagon, S., Brostovski, O., Galun, M., Irani, M.: Detecting and sketching the common. In: CVPR (2010)
7. Cho, M., Kwak, S., Schmid, C., Ponce, J.: Unsupervised object discovery and localization in the wild: part-based matching with bottom-up region proposals. In: CVPR (2015)
8. Chum, O., Matas, J.: Unsupervised discovery of co-occurrence in sparse high dimensional data. In: CVPR (2010)
9. Dong, W., Charikar, M., Li, K.: Efficient k-nearest neighbor graph construction for generic similarity measures. In: WWW (2011)
10. Gammeter, S., Bossard, L., Quack, T., Gool, L.V.: I know what you did last summer: Object-level auto-annotation of holiday snaps. In: ICCV (2009)
11. Gordo, A., Almazan, J., Revaud, J., Larlus, D.: Deep image retrieval: Learning global representations for image search. In: ECCV (2016)
12. Gordo, A., Almazan, J., Revaud, J., Larlus, D.: End-to-end learning of deep visual representations for image retrieval. arXiv preprint arXiv:1610.07940 (2016)
13. Hubbell, C.H.: An input-output approach to clique identification. Sociometry (1965)
14. Iscen, A., Avrithis, Y., Tolias, G., Furon, T., Chum, O.: Fast spectral ranking for similarity search. In: CVPR (2018)
15. Iscen, A., Tolias, G., Avrithis, Y., Furon, T., Chum, O.: Efficient diffusion on region manifolds: recovering small objects with compact cnn representations. In: CVPR (2017)
16. Jégou, H., Douze, M., Schmid, C.: Improving bag-of-features for large scale image search. IJCV **87**(3), 316–336 (2010)
17. Jeong, D.-J., Choo, S., Seo, W., Cho, N.I.: Regional deep feature aggregation for image retrieval. In: ICASSP (2017)
18. Jimenez, A., Alvarez, J.M., Giro-i Nieto, X.: Class-weighted convolutional features for visual instance search. In: BMVC (2017)
19. Kalantidis, Y., Mellina, C., Osindero, S.: Cross-dimensional weighting for aggregated deep convolutional features. In: arXiv (2015)
20. Katz, L.: A new status index derived from sociometric analysis. Psychometrika **18**(1), 39–43 (1953)
21. Kim, G., Torralba, A.: Unsupervised detection of regions of interest using iterative link analysis. In: NIPS (2009)
22. Kim, J., Yoon, S.-E.: Regional attention based deep feature for image retrieval. In: BMVC (2018)
23. Knopp, J., Sivic, J., Pajdla, T.: Avoiding confusing features in place recognition. In: ECCV (2010)
24. Kwak, S., Cho, M., Laptev, I., Ponce, J., Schmid, C.: Unsupervised object discovery and tracking in video collections. In: CVPR (2015)
25. Laskar, Z., Kannala, J.: Context aware query image representation for particular object retrieval. In: Scandinavian Conference on Image Analysis (2017)
26. Mej, N.: Networks: An Introduction. Oxford University Press, Oxford (2010)
27. Mikolajczyk, K., Matas, J.: Improving descriptors for fast tree matching by optimal linear projection. In: CVPR (2007)
28. Mohedano, E., McGuinness, K., Giro-i Nieto, X., O'Connor, N.E.: Saliency weighted convolutional features for instance search. arXiv preprint arXiv:1711.10795 (2017)
29. Nocedal, J., Wright, S.: Numerical Optimization. Springer, Berlin (2006)
30. Noh, H., Araujo, A., Sim, J., Weyand, T., Han, B.: Large-scale image retrieval with attentive deep local features. In: arXiv (2016)
31. Oliva, A., Torralba, A.: Building the gist of a scene: the role of global image features in recognition. Prog. Brain Res. **155**, 23–36 (2006)
32. Omercevic, D., Perko, R., Targhi, A.T., Eklundh, J.-O., Leonardis, A.: Vegetation segmentation for boosting performance of mser feature detector. In: Computer Vision Winter Workshop (2008)
33. Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank citation ranking: bringing order to the web (1999)
34. Pang, S., Ma, J., Xue, J., Zhu, J., Ordonez, V.: Image retrieval using heat diffusion for deep feature aggregation. arXiv preprint arXiv:1805.08587 (2018)
35. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching. In: CVPR (2007)
36. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Lost in quantization: Improving particular object retrieval in large scale image databases. In: CVPR (2008)
37. Radenović, F., Iscen, A., Tolias, G., Avrithis, Y., Chum, O.: Revisiting oxford and paris: large-scale image retrieval benchmarking. In: CVPR (2018)
38. Radenović, F., Tolias, G., Chum, O.: CNN image retrieval learns from bow: unsupervised fine-tuning with hard examples. In: ECCV (2016)
39. Radenović, F., Tolias, G., Chum, O.: Fine-tuning cnn image retrieval with no human annotation. IEEE Trans. PAMI (2018)
40. Razavian, A.S., Sullivan, J., Carlsson, S., Maki, A.: Visual instance retrieval with deep convolutional networks. ITE Trans. Media. Technol. Appl. **4**, 251–258 (2016)

41. Rubinstein, M., Joulin, A., Kopf, J., Liu, C.: Unsupervised joint object discovery and segmentation in internet images. In: CVPR (2013)
42. Salvador, A., Giró-i Nieto, X., Marqués, F., Satoh, S.: Faster r-cnn features for instance search. In: CVPRW (2016)
43. Selvaraju, R.R., Das, A., Vedantam, R., Cogswell, M., Parikh, D., Batra, D.: Grad-CAM: Why did you say that? visual explanations from deep networks via gradient-based localization. arXiv preprint arXiv:1610.02391 (2016)
44. Shi, M., Avrithis, Y., Jegou, H.: Early burst detection for memory-efficient image retrieval. In: CVPR (2015)
45. Simeoni, O., Iscen, A., Tolias, G., Avrithis, Y., Chum, O.: Unsupervised object discovery for instance recognition. In: WACV (2018)
46. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. ICLR (2014)
47. Sivic, J., Zisserman, A.: Video Google: A text retrieval approach to object matching in videos. In: ICCV (2003)
48. Song, J., He, T., Gao, L., Xu, X., Shen, H.T.: Deep region hashing for efficient large-scale instance search from images. In: arXiv (2017)
49. Tolias, G., Avrithis, Y., Jégou, H.: Image search with selective match kernels: aggregation across single and multiple images. IJCV (2016)
50. Tolias, G., Kalantidis, Y., Avrithis, Y.: Symcity: Feature selection by symmetry for large scale image retrieval. In: ACM Multimedia (2012)
51. Tolias, G., Sicre, R., Jégou, H.: Particular object retrieval with integral max-pooling of cnn activations. In: ICLR (2016)
52. Turcot, P., Lowe, D.G.: Better matching with fewer features: The selection of useful features in large database recognition problems. In: ICCVW (2009)
53. Vigna, S.: Spectral ranking. arXiv preprint arXiv:0912.0238 (2009)
54. Wang, S., Jiang, S.: Instre: a new benchmark for instance-level object retrieval and recognition. ACM Trans. Multimed. Comput. Commun. Appl. (TOMM) **11**, 37 (2015)
55. Zheng, L., Wang, S., Wang, J., Tian, Q.: Accurate image search with multi-scale contextual evidences. IJCV **120**(1), 1–13 (2016)
56. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., Torralba, A.: Learning deep features for discriminative localization. In: CVPR (2016)
57. Zhou, D., Weston, J., Gretton, A., Bousquet, O., Schölkopf, B.: Ranking on data manifolds. In: NIPS (2003)
58. Zhu, Y., Wang, J., Xie, L., Zheng, L.: Attention-based pyramid aggregation network for visual place recognition. arXiv preprint arXiv:1808.00288 (2018)

Oriane Siméoni is a PhD student at Inria Rennes Bretagne–Atlantique in the LinkMedia team. She focuses her research on computer vision tasks and in particular study methods to go towards less supervision in visual learning. She received her engineering diploma from Enseirb-Matmeca and a MS degree in Statistics from Paul Sabatier University.

Ahmet Iscen received his B.Sc. degree from The State University of New York at Binghamton, and M.Sc. from Bilkent University. He received his PhD student at Inria Rennes and University of Rennes. Currently, he is a post doctoral researcher at the CTU in Prague working on large-scale image retrieval.

Giorgos Tolias obtained his PhD from NTU of Athens and then moved as a post-doctoral researcher at Inria Rennes. Currently, he is a post-doctoral researcher at the Visual Recognition Group of CTU in Prague. He enjoys working on large-scale visual recognition problems.

Dr. Yannis Avrithis is a research scientist in LinkMedia team of Inria Rennes–Bretagne Atlantique, carrying out research on computer vision and machine learning. His research interests include visual feature detection, representation of visual appearance and geometry, image matching and registration, image indexing and retrieval, clustering, nearest neighbor search, object detection and recognition, scene classification, image/video segmentation and tracking, and video summarization. He has been involved in 15 European and 9 National research projects, he has co-supervised 10 Ph.D. theses and 12 Diploma theses, and he has published 3 theses, 3 edited volumes, 26 articles in journals, 108 in conferences and workshops, 8 book chapters and 16 technical reports in the above fields. He has contributed to the organization of 21 conferences and workshops, and is a reviewer in 16 scientific journals and 15 conferences.

Ondřej Chum is leading a team within the Visual Recognition Group at the Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague. He received the MSc degree in computer science from Charles University, Prague, in 2001 and the PhD degree from the Czech Technical University in Prague, in 2005. From 2006 to 2007, he was a postdoctoral researcher at the Visual Geometry Group, University of Oxford, United Kingdom. His research interests include large-scale image and particular object retrieval, object recognition, and robust estimation of geometric models. He is a member of Image and Vision Computing editorial board, and he has served in various roles at major international conferences. He co-organizes Computer Vision and Sports Summers School in Prague. He was the recipient of the Best Paper Prize at the British Machine Vision Conference in 2002. He was awarded the 2012 Outstanding Young Researcher in Image and Vision Computing runner up for researchers within seven years of their PhD. In 2017, he was the recipient of the Longuet-Higgins Prize.

XXIII CNN Image Retrieval Learns from BoW: Unsupervised Fine-Tuning with Hard Examples

Title: CNN Image Retrieval Learns from BoW: Unsupervised Fine-Tuning with Hard Examples
Authors: F. Radenović, G. Tolias, O. Chum
Published at: ECCV 2016

CNN Image Retrieval Learns from BoW: Unsupervised Fine-Tuning with Hard Examples

Filip Radenović Giorgos Tolias Ondřej Chum

CMP, Faculty of Electrical Engineering, Czech Technical University in Prague
{filip.radenovic,giorgos.tolias,chum}@cmp.felk.cvut.cz

Abstract. Convolutional Neural Networks (CNNs) achieve state-of-the-art performance in many computer vision tasks. However, this achievement is preceded by extreme manual annotation in order to perform either training from scratch or fine-tuning for the target task. In this work, we propose to fine-tune CNN for image retrieval from a large collection of unordered images in a fully automated manner. We employ state-of-the-art retrieval and Structure-from-Motion (SfM) methods to obtain 3D models, which are used to guide the selection of the training data for CNN fine-tuning. We show that both hard positive and hard negative examples enhance the final performance in particular object retrieval with compact codes.

Keywords: CNN fine-tuning, unsupervised learning, image retrieval

1 Introduction

Image retrieval has received a lot of attention since the advent of invariant local features, such as SIFT [1], and since the seminal work of Sivic and Zisserman [2] based on Bag-of-Words (BoW). Retrieval systems have reached a higher level of maturity by incorporating large visual codebooks [3,4], spatial verification [3,5] and query expansion [6,7,8]. These ingredients constitute the state of the art on particular object retrieval. Another line of research focuses on compact image representations in order to decrease memory requirements and increase the search efficiency. Representative approaches are Fisher vectors [9], VLAD [10] and alternatives [11,12,13]. Recent advances [14,15] show that Convolutional Neural Networks (CNN) offer an attractive alternative for image search representations with small memory footprint.

CNNs attracted a lot of attention after the work of Krizhevsky *et al.* [16]. Their success is mainly due to the computational power of GPUs and the use of very large annotated datasets [17]. Generation of the latter comes at the expense of costly manual annotation. Using CNN layer activations as off-the-shelf image descriptors [18,19] appears very effective and is adopted in many tasks [20,21,22]. In particular for image retrieval, Babenko *et al.* [14] and Gong *et al.* [22] concurrently propose the use of Fully Connected (FC) layer activations as descriptors, while convolutional layer activations are later shown to have superior performance [15,23,24,25].

Generalization to other tasks [26] is attained by CNN activations, at least up to some extent. However, initialization by a pre-trained network and re-training for another task, a process called *fine-tuning*, significantly improves the adaptation ability [27,28]. Fine-tuning by training with classes of particular objects, *e.g.* building classes in the work of Babenko *et al.* [14], is known to improve retrieval accuracy. This formulation is much closer to classification than to the desired properties of instance retrieval. Typical architectures for metric learning, such as siamese [29,30,31] or triplet networks [32,33,34] employ *matching* and *non-matching* pairs to perform the training and better suit to this task. In this fashion, Arandjelovic *et al.* [35] perform fine-tuning based on geo-tagged databases and, similar to our work, they directly optimize the the similarity measure to be used in the final task. In contrast to them, we dispense with the need of annotated data or any assumptions on the training dataset. A concurrent work [36] bears resemblance to ours but their focus is on boosting performance through end-to-end learning of a more sophisticated representation, while we target to reveal the importance of hard examples and of training data variation.

A number of image clustering methods based on local features have been introduced [37,38,39]. Due to the spatial verification, the *clusters* discovered by these methods are reliable. In fact, the methods provide not only clusters, but also a matching graph or sub-graph on the cluster images. These graphs are further used as an input to a Structure-from-Motion (SfM) pipeline to build a 3D model [40]. The SfM filters out virtually all mismatched images, and also provides camera positions for all matched images in the cluster. The whole process from unordered collection of images to 3D reconstructions is fully automatic.

In this paper, we address an unsupervised fine-tuning of CNN for image retrieval. We propose to exploit 3D reconstructions to select the training data for CNN. We show that compared to previous supervised approaches, the variability in the training data from 3D reconstructions delivers superior performance in the image retrieval task. During the training process the CNN is trained to learn what a state-of-the-art retrieval system based on local features and spatial verification would match. Such a system has large memory requirements and high query times, while our goal is to mimic this via CNN-based representation. We derive a short image representation and achieve similar performance to such state-of-the-art systems.

In particular we make the following contributions. (1) We exploit SfM information and enforce not only hard non-matching (*negative*) but also hard matching (*positive*) examples to be learned by the CNN. This is shown to enhance the derived image representation. (2) We show that the whitening traditionally performed on short representations [41] is, in some cases, unstable and we rather propose to learn the whitening through the same training data. Its effect is complementary to fine-tuning and it further boosts performance. (3) Finally, we set a new state-of-the-art based on compact representations for Oxford Buildings and Paris datasets by re-training well known CNNs, such as AlexNet [16] and VGG [42]. Remarkably, we are on par with existing 256D compact representations even by using 32D image vectors.

2 Related work

A variety of previous methods apply CNN activations on the task of image retrieval [22,15,23,24,25,43]. The achieved accuracy on retrieval is evidence for the generalization properties of CNNs. The employed networks were trained for image classification using ImageNet dataset, optimizing classification error. Babenko *et al.* [14] go one step further and re-train such networks with a dataset that is closer to the target task. They perform training with object classes that correspond to particular landmarks/buildings. Performance is improved on standard retrieval benchmarks. Despite the achievement, still, the final metric and utilized layers are different to the ones actually optimized during learning.

Constructing such training datasets requires manual effort. The same stands for attempts on different tasks [19,25] that perform fine-tuning and achieve increase of performance. In a recent work, geo-tagged datasets with timestamps offer the ground for weakly supervised fine-tuning of a triplet network [35]. Two images taken far from each other can be easily considered as non-matching, while matching examples are picked by the most similar nearby images. In the latter case, similarity is defined by the current representation of the CNN. This is the first approach that performs end-to-end fine-tuning for image retrieval and in particular for the task of geo-localization. The employed training data are now much closer to the final task. We differentiate by discovering matching and non-matching image pairs in an unsupervised way. Moreover, we derive matching examples based on 3D reconstruction which allows for harder examples, compared to the ones that the current network identifies. Even though hard negative mining is a standard process [20,35], this is not the case with hard positive examples. Large intra-class variation in classification tasks requires the positive pairs to be sampled carefully; forcing the model to learn extremely hard positives may result in over-fitting. Another exception is the work Simo-Serra *et al.* [44] where they mine hard positive patches for descriptor learning. They are also guided by 3D reconstruction but only at patch level.

Despite the fact that one of the recent advances is the triplet loss [32,33,34], note that also Arandjelovic *et al.* [35] use it, there are no extenstive and direct comparisons to siamese networks and the contrastive loss. One exception is the work of Hoffer and Ailon [34], where triplet loss is shown to be marginally better only on MNIST dataset. We rather employ a siamese architecture with the contrastive loss and find it to generalize better and to converge at higher performance than the triplet loss.

3 Network architecture and image representation

In this section we describe the derived image representation that is based on CNN and we present the network architecture used to perform the end-to-end learning in a siamese fashion. Finally, we describe how, after fine-tuning, we use the same training data to learn projections that appear to be an effective post-processing step.

3.1 Image representation

We adopt a compact representation that is derived from activations of convolutional layers and is shown to be effective for particular object retrieval [26,25]. We assume that a network is fully convolutional [45] or that all fully connected layers are discarded. Now, given an input image, the output is a 3D tensor $\mathcal{X}$ of $W \times H \times K$ dimensions, where K is the number of feature maps in the last layer. Let $\mathcal{X}_k$ be the set of all $W \times H$ activations for feature map $k \in \{1 \dots K\}$. The network output consists of K such sets of activations. The image representation, called Maximum Activations of Convolutions (MAC) [15,25], is simply constructed by max-pooling over all dimensions per feature map and is given by

$$\mathbf{f} = [\mathrm{f}_1 \dots \mathrm{f}_k \dots \mathrm{f}_K]^\top, \text{ with } \mathrm{f}_k = \max_{x \in \mathcal{X}_k} x \cdot \mathbb{1}(x > 0). \quad (1)$$

The indicator function $\mathbb{1}$ takes care that the feature vector $\mathbf{f}$ is non-negative, as if the last network layer was a Rectified Linear Unit (ReLU). The feature vector finally consists of the maximum activation per feature map and its dimensionality is equal to K. For many popular networks this is equal to 256 or 512, which makes it a compact image representation. MAC vectors are subsequently ℓ_2-normalized and similarity between two images is evaluated with inner product. The contribution of a feature map to the image similarity is measured by the product of the corresponding MAC vector components. In Figure 1 we show the image patches in correspondence that contribute most to the similarity. Such implicit correspondences are improved after fine-tuning. Moreover, the CNN fires less to ImageNet classes, *e.g.* cars and bicycles.

3.2 Network and siamese learning

The proposed approach is applicable to any CNN that consists of only convolutional layers. In this paper, we focus on re-training (*i.e.* fine-tuning) state-of-the-art CNNs for classification, in particular AlexNet and VGG. Fully connected layers are discarded and the pre-trained networks constitute the initialization for our convolutional layers. Now, the last convolutional layer is followed by a MAC layer that performs MAC vector computation (1). The input of a MAC layer is a 3D tensor of activation and the output is a non-negative vector. Then, an ℓ_2-normalization block takes care that output vectors are normalized. In the rest of the paper, MAC corresponds to the ℓ_2-normalized vector $\bar{\mathbf{f}}$.

We adopt a siamese architecture and train a two branch network. Each branch is a clone of the other, meaning that they share the same parameters. Training input consists of image pairs (i, j) and labels $Y(i, j) \in \{0, 1\}$ declaring whether a pair is non-matching (label 0) or matching (label 1). We employ the contrastive loss [29] that acts on the (non-)matching pairs and is defined as

$$\mathcal{L}(i,j) = \frac{1}{2}\left(Y(i,j)||\bar{\mathbf{f}}(i) - \bar{\mathbf{f}}(j)||^2 + (1 - Y(i,j))\left(\max\{0, \tau - ||\bar{\mathbf{f}}(i) - \bar{\mathbf{f}}(j)||\}\right)^2\right), \quad (2)$$

where $\bar{\mathbf{f}}(i)$ is the ℓ_2-normalized MAC vector of image i, and τ is a parameter defining when non-matching pairs have large enough distance in order not to be taken into account in the loss. We train the network using Stochastic Gradient Descent (SGD) and a large training set created automatically (see Section 4).

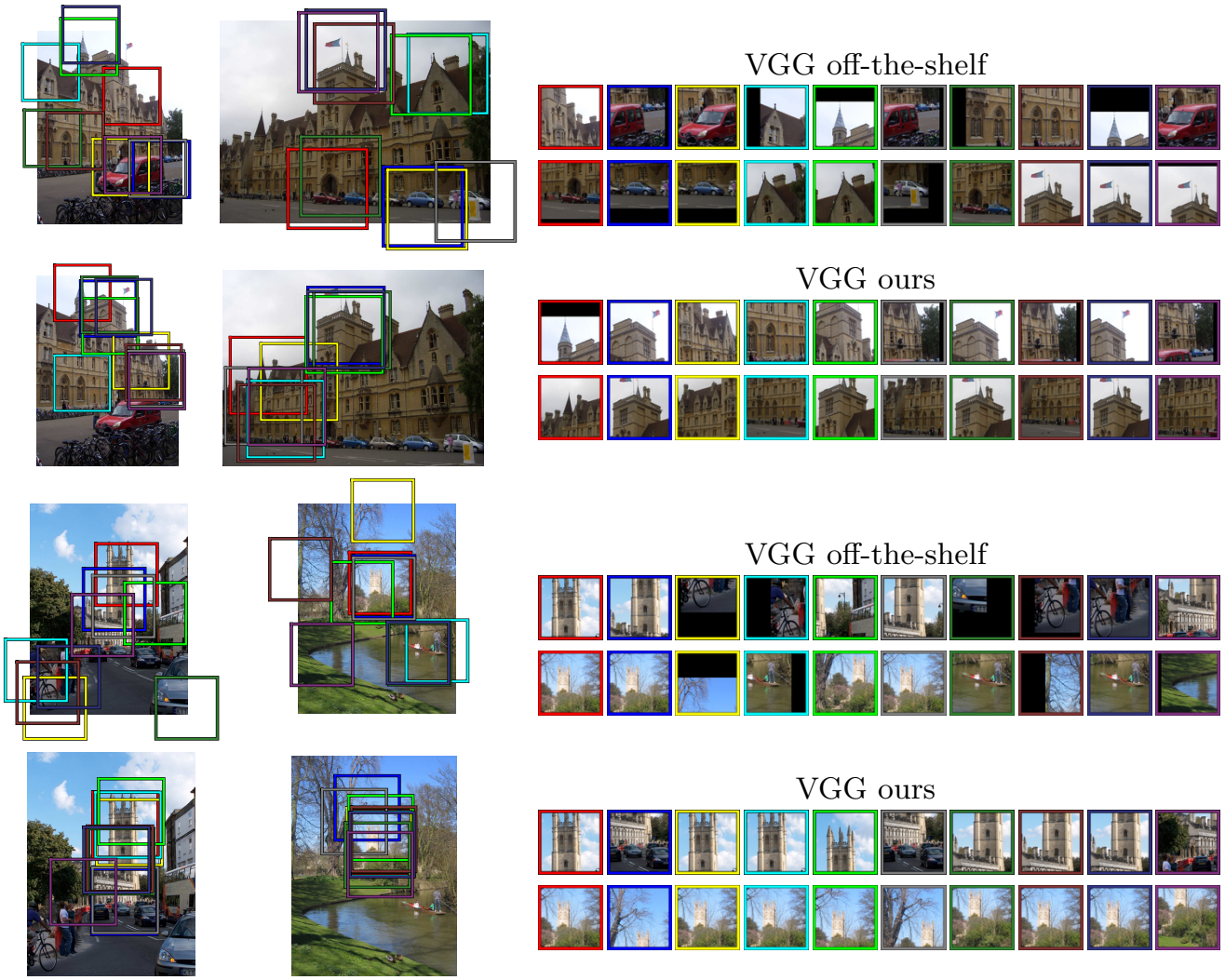


Fig. 1. Visualization of patches corresponding to the MAC vector components that have the highest contribution to the pairwise image similarity. Examples shown use CNN before (top) and after (bottom) fine-tuning of VGG. The same color corresponds to the same vector component (feature map) per image pair. The patch size is equal to the receptive field of the last pooling layer.

3.3 Whitening and dimensionality reduction

In this section, the post-processing of fine-tuned MAC vectors is considered. Previous methods [23,25] use PCA of an independent set for whitening and dimensionality reduction, that is the covariance matrix of all descriptors is analyzed. We propose to take advantage of the labeled data provided by the 3D models and use linear discriminant projections originally proposed by Mikolajczyk and Matas [46]. The projection is decomposed into two parts, whitening and rotation. The whitening part is the inverse of the square-root of the intraclass (matching pairs) covariance matrix $C_S^{-\frac{1}{2}}$, where

$$C_S = \sum_{Y(i,j)=1} \left(\bar{\mathbf{f}}(i) - \bar{\mathbf{f}}(j)\right)\left(\bar{\mathbf{f}}(i) - \bar{\mathbf{f}}(j)\right)^\top . \tag{3}$$

The rotation part is the PCA of the interclass (non-matching pairs) covariance matrix in the whitened space $\mathrm{eig}(C_S^{-\frac{1}{2}} C_D C_S^{-\frac{1}{2}})$, where

$$C_D = \sum_{Y(i,j)=0} \left(\bar{\mathbf{f}}(i) - \bar{\mathbf{f}}(j)\right)\left(\bar{\mathbf{f}}(i) - \bar{\mathbf{f}}(j)\right)^\top . \tag{4}$$

The projection $P = C_S^{-\frac{1}{2}} \text{eig}(C_S^{-\frac{1}{2}} C_D C_S^{-\frac{1}{2}})$ is then applied as $P^\top(\bar{\mathbf{f}}(i) - \mu)$, where μ is the mean MAC vector to perform centering. To reduce the descriptor dimensionality to D dimensions, only eigenvectors corresponding to D largest eigenvalues are used. Projected vectors are subsequently ℓ_2-normalized.

4 Training dataset

In this section we briefly summarize the tightly-coupled BoW and SfM reconstruction system [40,47] that is employed to automatically select our training data. Then, we describe how we exploit the 3D information to select harder matching pairs and hard non-matching pairs with larger variability.

4.1 BoW and 3D reconstruction

The retrieval engine used in the work of Schonberger *et al.* [40] builds upon BoW with fast spatial verification [3]. It uses Hessian affine local features [48], RootSIFT descriptors [49], and a fine vocabulary of 16M visual words [50]. Then, query images are chosen via min-hash and spatial verification, as in [37]. Image retrieval based on BoW is used to collect images of the objects/landmarks. These images serve as the initial matching graph for the succeeding SfM reconstruction, which is performed using state-of-the-art SfM [51,52]. Different mining techniques, *e.g.* zoom in, zoom out [53,54], sideways crawl [40], help to build larger and complete model.

In this work, we exploit the outcome of such a system. Given a large unannotated image collection, images are clustered and a 3D model is constructed per cluster. We use the terms *3D model*, *model* and *cluster* interchangeably. For each image, the estimated camera position is known, as well as the local features registered on the 3D model. We drop redundant (overlapping) 3D models, that might have been constructed from different seeds. Models reconstructing the same landmark but from different and disjoint viewpoints are considered as non-overlapping.

4.2 Selection of training image pairs

A 3D model is described as a bipartite visibility graph $\mathbb{G} = (\mathcal{I} \cup \mathcal{P}, \mathcal{E})$ [55], where images $\mathcal{I}$ and points $\mathcal{P}$ are the vertices of the graph. Edges of this graph are defined by visibility relations between cameras and points, *i.e.* if a point $p \in \mathcal{P}$ is visible in an image $i \in \mathcal{I}$, then there exists an edge $(i, p) \in \mathcal{E}$. The set of points observed by an image i is given by

$$\mathcal{P}(i) = \{p \in \mathcal{P} : (i, p) \in \mathcal{E}\}. \tag{5}$$

We create a dataset of tuples $(q, m(q), \mathcal{N}(q))$, where q represents a query image, $m(q)$ is a positive image that matches the query, and $\mathcal{N}(q)$ is a set of negative images that do not match the query. These tuples are used to form

Fig. 2. Examples of training query images (green border) and matching images selected as positive examples by methods (from left to right) $m_1(q)$, $m_2(q)$, and $m_3(q)$.

training image pairs, where each tuple corresponds to $|\mathcal{N}(q)| + 1$ pairs. For a query image q, a pool $\mathcal{M}(q)$ of candidate positive images is constructed based on the camera positions in the cluster of q. It consists of the k images with closest camera centers to the query. Due to the wide range of camera orientations, these do not necessarily depict the same object. We therefore propose three different ways to sample the positive image. The positives examples are fixed during the whole training process for all three strategies.

Positive images: MAC distance. The image that has the lowest MAC distance to the query is chosen as positive, formally

$$m_1(q) = \underset{i \in \mathcal{M}(q)}{\operatorname{argmin}} ||\bar{\mathbf{f}}(q) - \bar{\mathbf{f}}(i)||. \tag{6}$$

This strategy is similar to the one followed by Arandjelovic *et al.* [35]. They adopt this choice since only GPS coordinates are available and not camera orientations. Downside of this approach is that the chosen matching examples already have low distance, thus not forcing network to learn much out of the positive samples.

Positive images: maximum inliers. In this approach, the 3D information is exploited to choose the positive image, independently of the CNN descriptor. In particular, the image that has the highest number of co-observed 3D points with the query is chosen. That is,

$$m_2(q) = \underset{i \in \mathcal{M}(q)}{\operatorname{argmax}} |\mathcal{P}(q) \cap \mathcal{P}(i)|. \tag{7}$$

This measure corresponds to the number of spatially verified features between two images, a measure commonly used for ranking in BoW-based retrieval. As this choice is independent of the CNN representation, it delivers more challenging positive examples.

Fig. 3. Examples of training query images q (green border), hardest non-matching images $n(q)$ that are always selected as negative examples, and additional non-matching images selected as negative examples by $\mathcal{N}_1(q)$ and $\mathcal{N}_2(q)$ methods respectively.

Positive images: relaxed inliers. Even though both previous methods choose positive images depicting the same object as the query, the variance of viewpoints is limited. Instead of using a pool of images with similar camera position, the positive example is selected at random from a set of images that co-observe enough points with the query, but do not exhibit too extreme scale change. The positive example in this case is

$$m_3(q) = \texttt{random}\left\{ i \in \mathcal{M}(q) : \frac{|\mathcal{P}(i) \cap \mathcal{P}(q)|}{|\mathcal{P}(q)|} \geq t_i,\ \texttt{scale}(i,q) \leq t_s \right\}, \quad (8)$$

where $\texttt{scale}(i,q)$ is the scale change between the two images. This method results in selecting harder matching examples which are still guaranteed to depict the same object. Method m_3 chooses different image than m_1 on 86.5% of the queries. In Figure 2 we present examples of query images and the corresponding positives selected with the three different methods. The relaxed method increases the variability of viewpoints.

Negative images. Negative examples are selected from clusters different than the cluster of the query image, as the clusters are non-overlaping. Following a well-known procedure, we choose hard negatives [44,20], that is, non-matching images with the most similar descriptor. Two different strategies are proposed. In the first, $\mathcal{N}_1(q)$, k-nearest neighbors from all non-matching images are selected. In the other, $\mathcal{N}_2(q)$, the same criterion is used, but at most one image per cluster is allowed. While $\mathcal{N}_1(q)$ often leads to multiple, and very similar, instances of the same object, $\mathcal{N}_2(q)$ provides higher variability of the negative examples, see Figure 3. While positives examples are fixed during the whole training process, hard negatives depend on the current CNN parameters and are re-mined multiple times per epoch.

5 Experiments

In this section we discuss implementation details of our training, evaluate different components of our method, and compare to the state of the art.

5.1 Training setup and implementation details

Our training samples are derived from the dataset used in the work of Schonberger *et al.* [40], which consists of 7.4 million images downloaded from Flickr using keywords of popular landmarks, cities and countries across the world. The clustering procedure [37] gives 19,546 images to serve as query seeds. The extensive retrieval-SfM reconstruction [47] of the whole dataset results in 1,474 reconstructed 3D models. Removing overlapping models leaves us with 713 3D models containing 163,671 unique images from the initial dataset. The initial dataset contained on purpose all images of Oxford5k and Paris6k datasets. In this way, we are able to exclude 98 clusters that contain any image (or their near duplicates) from these test datasets.

The largest model has 11,042 images, while the smallest has 25. We randomly select 551 models (133,659 images) for training and 162 (30,012) for validation. The number of training queries per cluster is 10% of the cluster size for clusters of 300 or less images, or 30 images for larger clusters. A total number of 5,974 images is selected for training queries, and 1,691 for validation queries.

Each training and validation tuple contains 1 query, 1 positive and 5 negative images. The pool of candidate positives consists of $k = 100$ images with closest camera centers to the query. In particular, for method m_3, the inliers overlap threshold is $t_i = 0.2$, and the scale change threshold $t_s = 1.5$. Hard negatives are re-mined 3 times per epoch, *i.e.* roughly every 2,000 training queries. Given the chosen queries and the chosen positives, we further add 20 images per cluster to serve as candidate negatives during re-mining. This constitutes a training set of 22,156 images and it corresponds to the case that all 3D models are included for training.

To perform the fine-tuning as described in Section 3, we initialize by the convolutional layers of AlexNet [16] or VGG [42]. We use learning rate equal to 0.001, which is divided by 5 every 10 epochs, momentum 0.9, weight decay 0.0005, parameter τ for contrastive loss 0.7, and batch size of 5 training tuples. All training images are resized to a maximum 362×362 dimensionality, while keeping the original aspect ratio. Training is done for at most 30 epochs and the best network is selected based on performance, measured via mean Average Precision (mAP) [3], on validation tuples.

5.2 Test datasets and evaluation protocol

We evaluate our approach on Oxford buildings [3], Paris [56] and Holidays[1] [57] datasets. First two are closer to our training data, while the last differentiates by containing similar scenes and not only man made objects or buildings. These

[1] We use the up-right version of Holidays dataset (rotated).

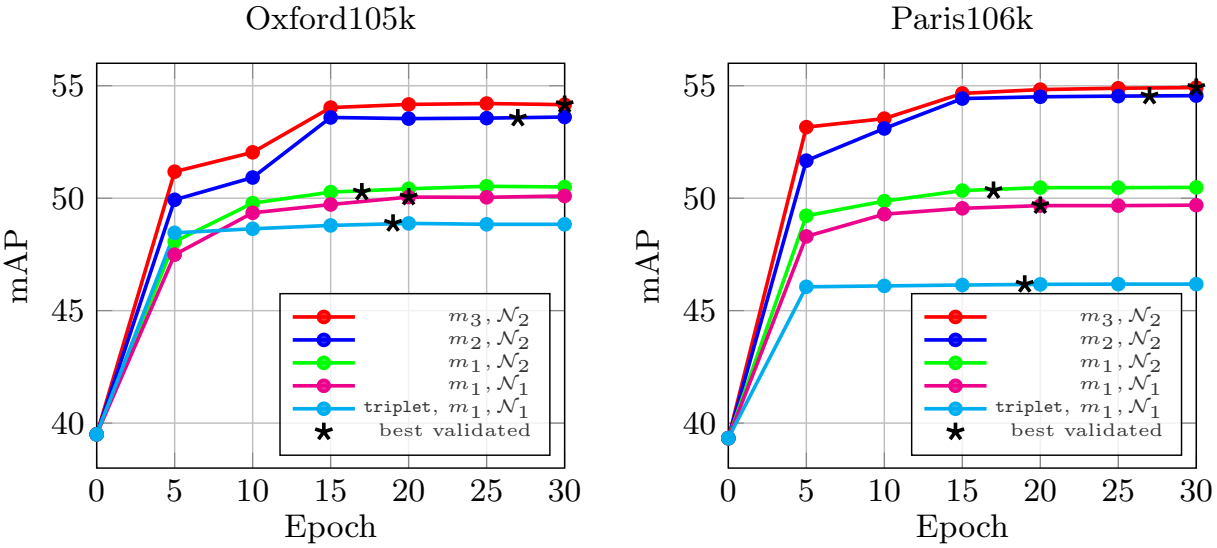


Fig. 4. Performance comparison of methods for positive and negative example selection. Evaluation is performed on AlexNet MAC on Oxford105k and Paris106k datasets. The plot shows the evolution of mAP with the number of training epochs. Epoch 0 corresponds to the off-the-shelf network. All approaches use contrastive loss, except if otherwise stated. The network with the best performance on the validation set is marked with $\star$.

are also combined with 100k distractors from Oxford100k to allow for evaluation at larger scale. The performance is measured via mAP. We follow the standard evaluation protocol for Oxford and Paris and crop the query images with the provided bounding box. The cropped image is fed as input to the CNN. However, to deliver a direct comparison with other methods, we also evaluate queries generated by keeping all activations that fall into this bounding box [23,35] when the full query image is used as input to the network. We refer to the cropped images approach as $\texttt{Crop}_{\mathcal{I}}$ and the cropped activations [23,35] as $\texttt{Crop}_{\mathcal{X}}$. The dimensionality of the images fed into the CNN is limited to 1024×1024 pixels. In our experiments, no vector post-processing is applied if not otherwise stated.

5.3 Results on image retrieval

Learning. We evaluate the off-the-shelf CNN and our fine-tuned ones after different number of training epochs. Our different methods for positive and negative selection are evaluated independently in order to decompose the benefit of each ingredient. Finally, we also perform a comparison with the triplet loss [35], trained on exactly the same training data as the ones used for our architecture with the contrastive loss. Results are presented in Figure 4. The results show that positive examples with larger view point variability, and negative examples with higher content variability, both acquire a consistent increase in the performance. The triplet loss[2] appears to be inferior in our context; we observe oscillation of the error in the validation set from early epochs, which implies over-fitting. In the rest of the paper, we adopt the $m_3, \mathcal{N}_2$ approach.

[2] The margin parameter for the triplet loss is set equal to 0.1 [35].

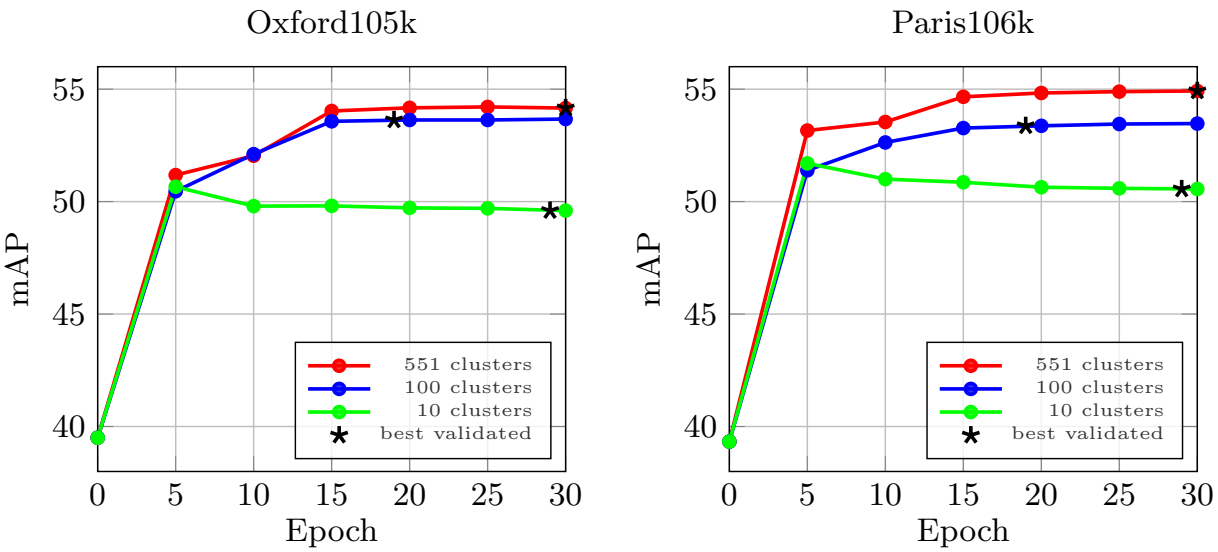


Fig. 5. Influence of the number of 3D models used for CNN fine-tuning. Performance is evaluated on AlexNet MAC on Oxford105k and Paris106k datasets using 10, 100 and 551 (all available) 3D models. The network with the best performance on the validation set is marked with $\star$.

Dataset variability. We perform fine-tuning by using a subset of the available 3D models. Results are presented in Figure 5 with 10, 100 and 551 (all available) clusters, while keeping the amount of training data, *i.e.* training queries, fixed. In the case of 10 and 100 models we use the largest ones, *i.e.* ones with the highest number of images. It is better to train with all 3D models due to the higher variability in the training set. Remarkably, significant increase in performance is achieved even with 10 or 100 models. However, the network is able to over-fit in the case of few clusters. All models are utilized in all other experiments.

Learned projections. The PCA-whitening [41] (PCA_w) is shown to be essential in some cases of CNN-based descriptors [14,23,25]. On the other hand, it is shown that on some of the datasets, the performance after PCA_w substantially drops compared with the raw descriptors (max pooling on Oxford5k [23]). We perform comparison of this traditional way of whitening and our learned whitening (L_w), described in Section 3.3. Table 1 shows results without post-processing and with the two different methods of whitening. Our experiments confirm, that PCA_w often reduces the performance. In contrast to that, the proposed L_w achieves the best performance in most cases and is never the worst performing method. Compared to no post-processing baseline, L_w reduces the performance twice for AlexNet, but the drop is negligible compared to the drop observed for PCA_w. For the VGG, the proposed L_w *always* outperforms the no post-processing baseline.

Our unsupervised learning directly optimizes MAC when extracted from full images, however, we further apply the fine-tuned networks to construct R-MAC representation [25] with regions of three different scales. It consists of extracting

Table 1. Performance comparison of CNN vector post-processing: no post-processing, PCA-whitening [41] (PCA_w) and our learned whitening (L_w). No dimensionality reduction is performed. Fine-tuned AlexNet produces a 256D vector and fine-tuned VGG a 512D vector. The best performance highlighted in **bold**, the worst in **blue**. The proposed method consistently performs either the best (18 out of 24 cases) or on par with the best method. On the contrary, PCA_w [41] often hurts the performance significantly. Best viewed in color.

Net	Post	Oxf5k		Oxf105k		Par6k		Par106k		Hol		Hol101k	
		MAC	R-MAC	MAC	R-MAC	MAC	R-MAC	MAC	R-MAC	MAC	R-MAC	MAC	R-MAC
Alex	–	60.2	**53.9**	**54.2**	**46.4**	67.5	**70.2**	**54.9**	**58.4**	73.1	**77.3**	61.6	**67.1**
	PCA_w	**56.9**	60.0	**44.1**	48.4	**64.3**	**75.1**	**46.8**	61.7	**73.0**	**81.7**	**59.4**	70.4
	L_w	**62.2**	**62.5**	52.8	**53.2**	**68.9**	74.4	54.7	**61.8**	**76.2**	81.5	**63.8**	**70.8**
VGG	–	78.7	**70.1**	72.7	**63.1**	**77.1**	**78.1**	69.6	**70.4**	**76.9**	**80.0**	65.3	**68.8**
	PCA_w	**76.1**	76.3	**68.9**	68.5	79.0	**84.5**	**69.1**	**77.1**	77.1	82.3	**63.6**	71.0
	L_w	**79.7**	**77.0**	**73.9**	**69.2**	**82.4**	83.8	**74.6**	76.4	**79.5**	**82.5**	**67.0**	**71.5**

MAC from multiple sub-windows and then aggregating them. Directly optimizing R-MAC during learning is possible and could offer extra improvements, but this is left for future work. Despite the fact that R-MAC offers improvements due to the regional representation, in our experiments it is not always better than MAC, since the latter is optimized during the end-to-end learning. We apply PCA_w on R-MAC as in [25], that is, we whiten each region vector first and then aggregate. Performance is significantly higher in this way. In the case of our L_w, we directly whiten the final vector after aggregation, which is also faster to compute.

Dimensionality reduction. We compare dimensionality reduction performed with PCA_w [41] and with our L_w. The performance for varying descriptor dimensionality is plotted in Figure 6. The plots suggest that L_w works better in higher dimensionalities, while PCA_w works slightly better for the lower ones. Remarkably, MAC reduced down to 16D outperforms state-of-the-art on BoW-based 128D compact codes [11] on Oxford105k (45.5 vs 41.4). Further results on very short codes can be found in Table 2.

Over-fitting and generalization. In all experiments, all clusters including any image (not only query landmarks) from Oxford5k or Paris6k datasets are removed. To evaluate whether the network tends to over-fit to the training data or to generalize, we repeat the training, this time using all 3D reconstructions, including those of Oxford and Paris landmarks. The same amount of training queries is used for a fair comparison. We observe negligible difference in the performance of the network on Oxford and Paris evaluation results, *i.e.* the difference in mAP was on average +0.3 over all testing datasets. We conclude that the network generalizes well and is relatively insensitive to over-fitting.

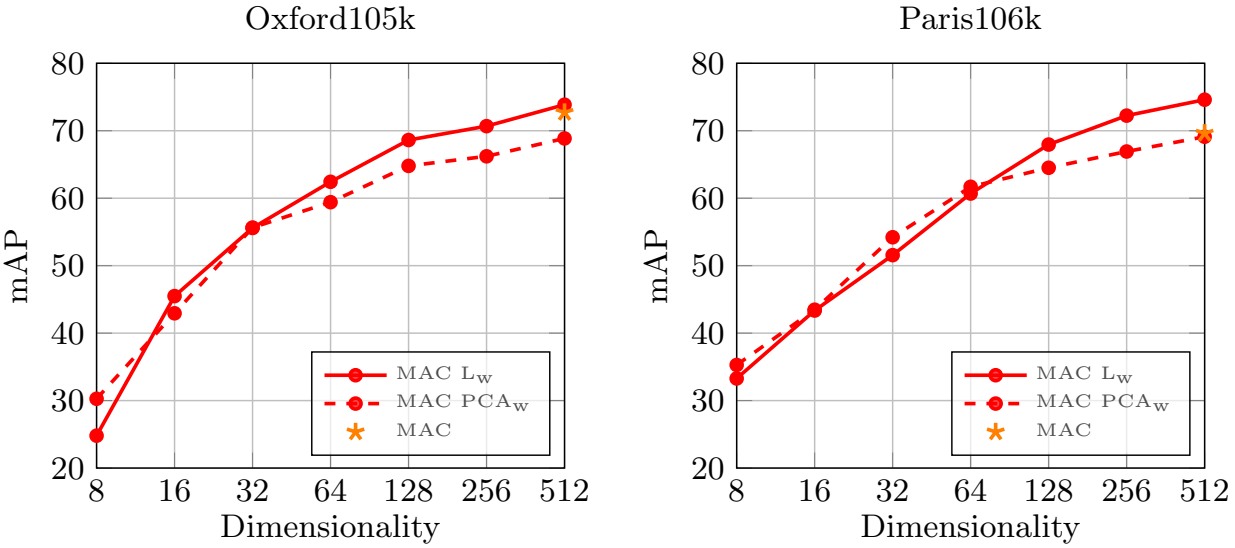


Fig. 6. Performance comparison of the dimensionality reduction performed by PCA_w and our L_w on fine-tuned VGG MAC on Oxford105k and Paris106k datasets.

Comparison with the state of the art. We extensively compare our results with the state-of-the-art performance on compact image representations and extremely short codes. The results for MAC and R-MAC with the fine-tuned networks are summarized together with previously published results in Table 2. The proposed methods outperform the state of the art on Paris and Oxford datasets, with and without distractors with all 16D, 32D, 128D, 256D, and 512D descriptors. On Holidays dataset, the Neural codes [14] win the extreme short code category, while off-the-shelf NetVlad performs the best on 256D and higher.

We additionally combine MAC and R-MAC with recent localization method for re-ranking [25] to further boost the performance. Our scores compete with state-of-the-art systems based on local features and query expansion. These have much higher memory needs and larger query times.

Observations on the recently published NetVLAD [35]: (1) After fine-tuning, NetVLAD performance drops on Holidays, while our training improves off-the-shelf results on all datasets. (2) Our 32D MAC descriptor has comparable performance to 256D NetVLAD on Oxford5k (ours 69.2 vs NetVLAD 63.5), and on Paris6k (ours 69.5 vs NetVLAD 73.5).

6 Conclusions

We addressed fine-tuning of CNN for image retrieval. The training data are selected from an automated 3D reconstruction system applied on a large unordered photo collection. The proposed method does not require any manual annotation and yet outperforms the state of the art on a number of standard benchmarks for wide range (16 to 512) of descriptor dimensionality. The achieved results are reaching the level of the best systems based on local features with spatial matching and query expansion, while being faster and requiring less memory. Training data, fine-tuned networks and evaluation code are publicly available[3].

[3] http://cmp.felk.cvut.cz/~radenfil/projects/siamac.html

Table 2. Performance comparison with the state of the art. Results reported with the use of AlexNet or VGG are marked by (**A**) or (**V**), respectively. Use of fine-tuned network is marked by (**f**), otherwise the off-the-shelf network is implied. D: Dimensionality. Our methods are marked with ⋆ and they are always accompanied by L_w. New state of the art highlighted in **red**, surpassed state of the art in **bold**, state of the art that retained the title in **outline**, and our methods that outperform previous state of the art on a gray background. Best viewed in color.

Method		D	Oxf5k		Oxf105k		Par6k		Par106k		Hol	Hol 101k
			$\text{Crop}_{\mathcal{I}}$	$\text{Crop}_{\mathcal{X}}$	$\text{Crop}_{\mathcal{I}}$	$\text{Crop}_{\mathcal{X}}$	$\text{Crop}_{\mathcal{I}}$	$\text{Crop}_{\mathcal{X}}$	$\text{Crop}_{\mathcal{I}}$	$\text{Crop}_{\mathcal{X}}$		
Compact representations												
mVoc/BoW [11]		128	48.8	–	41.4	–	–	–	–	–	65.6	–
Neural codes† [14]	(**fA**)	128	–	**55.7**	–	**52.3**	–	–	–	–	**78.9**	–
MAC‡	(**V**)	128	53.5	**55.7**	43.8	45.6	69.5	**70.6**	53.4	**55.4**	72.6	**56.7**
CroW [24]	(**V**)	128	**59.2**	–	**51.6**	–	**74.6**	–	**63.2**	–	–	–
⋆ MAC	(**fV**)	128	**75.8**	**76.8**	**68.6**	**70.8**	77.6	78.8	68.0	69.0	73.2	58.8
⋆ R-MAC	(**fV**)	128	72.5	76.7	64.3	69.7	**78.5**	**80.3**	**69.3**	**71.2**	**79.3**	**65.2**
MAC‡	(**V**)	256	54.7	56.9	45.6	47.8	71.5	72.4	55.7	**57.3**	76.5	**61.3**
SPoC [23]	(**V**)	256	–	53.1	–	**50.1**	–	–	–	–	80.2	–
R-MAC [25]	(**A**)	256	56.1	–	47.0	–	72.9	–	60.1	–	–	–
CroW [24]	(**V**)	256	**65.4**	–	**59.3**	–	**77.9**	–	**67.8**	–	83.1	–
NetVlad [35]	(**V**)	256	–	55.5	–	–	–	67.7	–	–	**86.0**	–
NetVlad [35]	(**fV**)	256	–	**63.5**	–	–	–	**73.5**	–	–	84.3	–
⋆ MAC	(**fA**)	256	62.2	65.4	52.8	58.0	68.9	72.2	54.7	58.5	76.2	63.8
⋆ R-MAC	(**fA**)	256	62.5	68.9	53.2	61.2	74.4	76.6	61.8	64.8	81.5	**70.8**
⋆ MAC	(**fV**)	256	**77.4**	**78.2**	**70.7**	**72.6**	80.8	81.9	72.2	73.4	77.3	62.9
⋆ R-MAC	(**fV**)	256	74.9	**78.2**	67.5	72.1	**82.3**	**83.5**	**74.1**	**75.6**	81.4	69.4
MAC‡	(**V**)	512	56.4	**58.3**	47.8	**49.2**	72.3	**72.6**	58.0	**59.1**	76.7	**62.7**
R-MAC [25]	(**V**)	512	66.9	–	61.6	–	**83.0**	–	**75.7**	–	–	–
CroW [24]	(**V**)	512	**68.2**	–	**63.2**	–	79.6	–	71.0	–	84.9	–
⋆ MAC	(**fV**)	512	**79.7**	80.0	**73.9**	**75.1**	82.4	82.9	74.6	75.3	79.5	67.0
⋆ R-MAC	(**fV**)	512	77.0	**80.1**	69.2	74.1	**83.8**	**85.0**	**76.4**	**77.9**	82.5	**71.5**
Extreme short codes												
Neural codes† [14]	(**fA**)	16	–	**41.8**	–	**35.4**	–	–	–	–	**60.9**	–
⋆ MAC	(**fV**)	16	**56.2**	**57.4**	**45.5**	**47.6**	57.3	62.9	43.4	48.5	51.3	25.6
⋆ R-MAC	(**fV**)	16	46.9	52.1	37.9	41.6	**58.8**	**63.2**	**45.6**	**49.6**	54.4	**31.7**
Neural codes† [14]	(**fA**)	32	–	**51.5**	–	**46.7**	–	–	–	–	**72.9**	–
⋆ MAC	(**fV**)	32	**65.3**	**69.2**	**55.6**	**59.5**	**63.9**	**69.5**	51.6	**56.3**	62.4	41.8
⋆ R-MAC	(**fV**)	32	58.4	64.2	50.1	55.1	**63.9**	67.4	**52.7**	55.8	68.0	**49.6**
Re-ranking (R) and query expansion (QE)												
BoW(1M)+QE [6]		–	82.7	–	76.7	–	80.5	–	71.0	–	–	–
BoW(16M)+QE [50]		–	84.9	–	79.5	–	82.4	–	77.3	–	–	–
HQE(65k) [8]		–	**88.0**	–	**84.0**	–	82.8	–	–	–	–	–
R-MAC+R+QE [25]	(**V**)	512	77.3	–	73.2	–	**86.5**	–	**79.8**	–	–	–
CroW+QE [24]	(**V**)	512	72.2	–	67.8	–	85.5	–	79.7	–	–	–
⋆ MAC+R+QE	(**fV**)	512	85.0	**85.4**	81.8	**82.3**	**86.5**	**87.0**	78.8	79.6	–	–
⋆ R-MAC+R+QE	(**fV**)	512	82.9	84.5	77.9	80.4	85.6	86.4	78.3	**79.7**	–	–

†: Full images are used as queries making the results not directly comparable on Oxford and Paris.
‡: Our evaluation of MAC with PCA_w as in [25] with the off-the-shelf network.

Acknowledgment. Work was supported by the MSMT LL1303 ERC-CZ grant.

References

1. Lowe, D.: Distinctive image features from scale-invariant keypoints. IJCV (2004) 1
2. Sivic, J., Zisserman, A.: Video Google: A text retrieval approach to object matching in videos. In: ICCV. (2003) 1
3. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching. In: CVPR. (2007) 1, 6, 9
4. Avrithis, Y., Kalantidis, Y.: Approximate Gaussian mixtures for large scale vocabularies. In: ECCV. (2012) 1
5. Shen, X., Lin, Z., Brandt, J., Wu, Y.: Spatially-constrained similarity measure for large-scale object retrieval. PAMI (2014) 1
6. Chum, O., Mikulik, A., Perdoch, M., Matas, J.: Total recall II: Query expansion revisited. In: CVPR. (2011) 1, 14
7. Danfeng, Q., Gammeter, S., Bossard, L., Quack, T., Gool, L.V.: Hello neighbor: Accurate object retrieval with k-reciprocal nearest neighbors. In: CVPR. (2011) 1
8. Tolias, G., Jégou, H.: Visual query expansion with or without geometry: refining local descriptors by feature aggregation. Pattern Recognition (2014) 1, 14
9. Perronnin, F., Liu, Y., Sanchez, J., Poirier, H.: Large-scale image retrieval with compressed Fisher vectors. In: CVPR. (2010) 1
10. Jégou, H., Perronnin, F., Douze, M., Sánchez, J., Pérez, P., Schmid, C.: Aggregating local descriptors into compact codes. PAMI (2012) 1
11. Radenović, F., Jegou, H., Chum, O.: Multiple measurements and joint dimensionality reduction for large scale image search with short vectors. In: ICMR. (2015) 1, 12, 14
12. Arandjelovic, R., Zisserman, A.: All about VLAD. In: CVPR. (2013) 1
13. Tolias, G., Furon, T., Jégou, H.: Orientation covariant aggregation of local descriptors with embeddings. In: ECCV. (2014) 1
14. Babenko, A., Slesarev, A., Chigorin, A., Lempitsky, V.: Neural codes for image retrieval. In: ECCV. (2014) 1, 2, 3, 11, 13, 14
15. Razavian, A.S., Sullivan, J., Maki, A., Carlsson, S.: A baseline for visual instance retrieval with deep convolutional networks. In: arXiv:1412.6574. (2014) 1, 3, 4
16. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS. (2012) 1, 2, 9
17. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. IJCV (2015) 1
18. Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., Darrell, T.: DeCAF: A deep convolutional activation feature for generic visual recognition. In: arXiv:1310.1531. (2013) 1
19. Razavian, A.S., Azizpour, H., Sullivan, J., Carlsson, S.: CNN features off-the-shelf: An astounding baseline for recognition. In: CVPRW. (2014) 1, 3
20. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR. (2014) 1, 3, 8
21. Iandola, F., Moskewicz, M., Karayev, S., Girshick, R., Darrell, T., Keutzer, K.: DenseNet: Implementing efficient ConvNet descriptor pyramids. In: arXiv:1404.1869. (2014) 1

22. Gong, Y., Wang, L., Guo, R., Lazebnik, S.: Multi-scale orderless pooling of deep convolutional activation features. In: ECCV. (2014) 1, 3
23. Babenko, A., Lempitsky, V.: Aggregating deep convolutional features for image retrieval. In: ICCV. (2015) 1, 3, 5, 10, 11, 14
24. Kalantidis, Y., Mellina, C., Osindero, S.: Cross-dimensional weighting for aggregated deep convolutional features. In: arXiv:1512.04065. (2015) 1, 3, 14
25. Tolias, G., Sicre, R., Jégou, H.: Particular object retrieval with integral max-pooling of CNN activations. In: ICLR. (2016) 1, 3, 4, 5, 11, 12, 13, 14
26. Azizpour, H., Razavian, A.S., Sullivan, J., Maki, A., Carlsson, S.: From generic to specific deep representations for visual recognition. In: CVPRW. (2015) 2, 4
27. Zhang, N., Donahue, J., Girshick, R., Darrell, T.: Part-based R-CNNs for fine-grained category detection. In: ECCV. (2014) 2
28. Oquab, M., Bottou, L., Laptev, I., Sivic, J.: Learning and transferring mid-level image representations using convolutional neural networks. In: CVPR. (2014) 2
29. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: CVPR. (2005) 2, 4
30. Hadsell, R., Chopra, S., LeCun, Y.: Dimensionality reduction by learning an invariant mapping. In: CVPR. (2006) 2
31. Hu, J., Lu, J., Tan, Y.P.: Discriminative deep metric learning for face verification in the wild. In: CVPR. (2014) 2
32. Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, J., Philbin, J., Chen, B., Wu, Y.: Learning fine-grained image similarity with deep ranking. In: CVPR. (2014) 2, 3
33. Schroff, F., Kalenichenko, D., Philbin, J.: FaceNet: A unified embedding for face recognition and clustering. In: CVPR. (2015) 2, 3
34. Hoffer, E., Ailon, N.: Deep metric learning using triplet network. In: ICLR Workshop. (2015) 2, 3
35. Arandjelovic, R., Gronat, P., Torii, A., Pajdla, T., Sivic, J.: NetVLAD: CNN architecture for weakly supervised place recognition. In: CVPR. (2016) 2, 3, 7, 10, 13, 14
36. Gordo, A., Almazan, J., Revaud, J., Larlus, D.: Deep image retrieval: Learning global representations for image search. In: ECCV. (2016) 2
37. Chum, O., Matas, J.: Large-scale discovery of spatially related images. PAMI (2010) 2, 6, 9
38. Weyand, T., Leibe, B.: Discovering details and scene structure with hierarchical iconoid shift. In: ICCV. (2013) 2
39. Philbin, J., Sivic, J., Zisserman, A.: Geometric latent dirichlet allocation on a matching graph for large-scale image datasets. IJCV (2011) 2
40. Schönberger, J.L., Radenović, F., Chum, O., Frahm, J.M.: From single image query to detailed 3D reconstruction. In: CVPR. (2015) 2, 6, 9
41. Jégou, H., Chum, O.: Negative evidences and co-occurences in image retrieval: The benefit of PCA and whitening. In: ECCV. (2012) 2, 11, 12
42. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: arXiv:1409.1556. (2014) 2, 9
43. Zheng, L., Zhao, Y., Wang, S., Wang, J., Tian, Q.: Good practice in CNN feature transfer. In: arXiv:1604.00133. (2016) 3
44. Simo-Serra, E., Trulls, E., Ferraz, L., Kokkinos, I., Moreno-Noguer, F.: Fracking deep convolutional image descriptors. In: arXiv:1412.6537. (2014) 3, 8
45. Papandreou, G., Kokkinos, I., Savalle, P.A.: Modeling local and global deformations in deep learning: Epitomic convolution, multiple instance learning, and sliding window detection. In: CVPR. (2015) 4

46. Mikolajczyk, K., Matas, J.: Improving descriptors for fast tree matching by optimal linear projection. In: ICCV. (2007) 5
47. Radenović, F., Schönberger, J.L., Ji, D., Frahm, J.M., Chum, O., Matas, J.: From dusk till dawn: Modeling in the dark. In: CVPR. (2016) 6, 9
48. Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., Gool, L.V.: A comparison of affine region detectors. IJCV (2005) 6
49. Arandjelovic, R., Zisserman, A.: Three things everyone should know to improve object retrieval. In: CVPR. (2012) 6
50. Mikulik, A., Perdoch, M., Chum, O., Matas, J.: Learning vocabularies over a fine quantization. IJCV (2013) 6, 14
51. Frahm, J.M., Georgel, P., Gallup, D., Johnson, T., Raguram, R., Wu, C., Jen, Y.H., Dunn, E., Clipp, B., Lazebnik, S., Pollefeys, M.: Building Rome on a cloudless day. In: ECCV. (2010) 6
52. Agarwal, S., Furukawa, Y., Snavely, N., Simon, I., Curless, B., Seitz, S.M., Szeliski, R.: Building Rome in a day. Communications of the ACM (2011) 6
53. Mikulik, A., Chum, O., Matas, J.: Image retrieval for online browsing in large image collections. In: SISAP. (2013) 6
54. Mikulík, A., Radenović, F., Chum, O., Matas, J.: Efficient image detail mining. In: ACCV. (2014) 6
55. Li, Y., Snavely, N., Huttenlocher, D.P.: Location recognition using prioritized feature matching. In: ECCV. (2010) 6
56. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Lost in quantization: Improving particular object retrieval in large scale image databases. In: CVPR. (2008) 9
57. Jégou, H., Douze, M., Schmid, C.: Hamming embedding and weak geometric consistency for large scale image search. In: ECCV. (2008) 9

XXIV Mining on Manifolds: Metric Learning without Labels

Title: Mining on Manifolds: Metric Learning without Labels
Authors: A. Iscen, G. Tolias, Y. Avrithis, O. Chum
Published at: CVPR 2018

Mining on Manifolds: Metric Learning without Labels

Ahmet Iscen[1] Giorgos Tolias[1] Yannis Avrithis[2] Ondřej Chum[1]
[1]VRG, FEE, CTU in Prague [2]Inria Rennes

Abstract

In this work we present a novel unsupervised framework for hard training example mining. The only input to the method is a collection of images relevant to the target application and a meaningful initial representation, provided e.g. *by pre-trained CNN. Positive examples are distant points on a single manifold, while negative examples are nearby points on different manifolds. Both types of examples are revealed by disagreements between Euclidean and manifold similarities. The discovered examples can be used in training with any discriminative loss.*

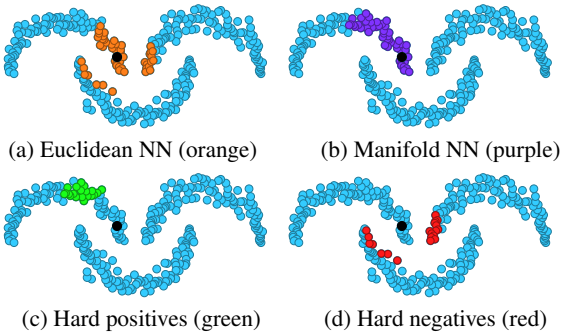
The method is applied to unsupervised fine-tuning of pre-trained networks for fine-grained classification and particular object retrieval. Our models are on par or are outperforming prior models that are fully or partially supervised.

1. Introduction

The success of deep neural networks on large-scale problems has been first demonstrated on the task of supervised classification [26]. It was shown that embeddings, typically provided by the convolutional layers of a network, are applicable beyond classification tasks. These tasks include particular object retrieval [15], local descriptor learning [17], ranking [62], and nearest-neighbor regression [6]. A common practice is to start with a pre-trained network [50, 55, 19] and apply metric learning [9, 62, 18] to fine-tune the network for a particular task.

To improve over the initial network, novel training samples are sought for which the initial network performs poorly. Such training samples are used to re-train the network using loss functions alternative to cross-entropy (*e.g.* contrastive [9], triplet [62] or batch-level [36]). The approaches to obtain relevant training data range from further human supervision [3, 36] to instance clustering [44, 32, 5], exploiting the temporal dimension in video [64, 65], predicting the spatial layout of image patches [11], or using existing computer vision pipelines to match unstructured image collections pairwise [15, 42].

Most recent deep metric learning approaches can learn powerful embeddings but still use class labels. This is unsatisfying not only because we miss the opportunity of learning from unlabeled data, but learned representations of each class are unimodal [44]. Therefore, whatever the loss function [62, 36]) or sampling [18, 34], the problem remains supervised classification. On the other hand, conventional nonlinear dimension reduction or manifold learning methods exploit the manifold structure of the data starting from nearest neighbor graphs and are otherwise unsupervised [56, 45, 7]. However, most do not learn an explicit mapping from the input to the embedding space and have difficulties in generalizing to new data.



(a) Euclidean NN (orange) (b) Manifold NN (purple)
(c) Hard positives (green) (d) Hard negatives (red)

Figure 1. Given an anchor point (black) and its k nearest Euclidean (NN_k^e) and manifold (NN_k^m) neighbors in a dataset, we choose positive samples as $\text{NN}_k^m \setminus \text{NN}_k^e$, and negative as $\text{NN}_k^e \setminus \text{NN}_k^m$. Data is unlabeled and the selection is fully unsupervised, including anchors.

We attempt to bridge this gap in this work. In particular, we propose a novel method of hard training sample mining in a fully unsupervised manner, simply from an unordered collection of images *relevant* to the final task. We observe that a similarity between two images is improved by considering all, even unlabelled, available data. In particular, we exploit similarity measured on a manifold estimated by a random walk process [21].

The learning starts from the initial representation space of unlabeled data. Given an anchor point that is part of the data, neighbors on the manifold that are not Euclidean neighbors are considered positive samples. In the new learned representation space, the positive sample should be attracted to the anchor to reflect the similarity measured on the manifold. Conversely, Euclidean neighbors that are not neighbors on the manifold are considered negative and should be repelled. The idea is illustrated in Figure 1.

The advantage of learning a new representation over using the estimated manifold similarity is twofold. First, estimating the manifold similarity has additional computational and memory requirements at query time. Second, we show that the novel embedding generalizes better not only to previously unseen queries, but also to unseen datasets.

We apply the proposed method to fine-grained classification and to particular object retrieval. Our models obtained by unsupervised fine-tuning of pre-trained networks are on par or are outperforming prior models that are human supervised or use additional domain-specific expertise.

The paper is organized as follows. Section 2 discusses related work. Sections 3 and 4 present our learning method and applications, respectively. Experiments are given in Section 5 conclusions are drawn in Section 6.

2. Related work

This section contains a brief overview of related work on metric learning, embeddings for instance retrieval and representation learning without human labeled data.

Metric learning. Recent approaches based on low-dimensional CNN-based embeddings achieve promising results on this task [47, 36]. A key ingredient of some approaches is *hard negative mining*, which comes from the days of SVMs in object detection [13]. This principle has been known much earlier as *bootstrapping* [54]. Instead of sampling all negative instances for an anchor point, the most challenging negative instances are mined which finally offer faster but equally accurate learning. However, this process is not trivial. Simple hard negative mining from a different class label might corrupt the training process due to mislabeled images [47, 33].

Schroff *et al*. [47] use triplet loss and propose *semi-hard* mining in an attempt to solve this problem. They sample negatives within the batch, such that they are close to the anchor point but further away from positives. This concept is widely adapted in other metric learning approaches [38, 36]. Wu *et al*. [67] improve it by uniformly sampling negative instances weighted by their distance. This allows them to sample points from various regions of the feature space instead of certain clusters.

Other methods, such as *lifted-structure* [36] and n-pair loss [52], focus on the loss function and define constraints on the pairwise distances of all points in a batch. More recently, several methods try to optimize the learned embedding based on the global distribution of the data. Song *et al*. [53] try to optimize the clustering quality. Harwood *et al*. [18] combine triplet loss and global loss using an efficient hard mining procedure. All these methods use class labels during sampling.

Particular object retrieval is another application where descriptors are trained similarly to metric learning. However, class labels do not usually exist as it is intractable to enumerate all possible instances or their viewpoints. Traditional instance search algorithms use hand-designed descriptors [51, 39, 57, 25], but recent advances show the interest of feature learning [42, 15]. Transfer learning from category-level classification is one case [43]. Labeling at landmark level has been attempted as well, treating this task as classification, but the labels are quite noisy [3].

The state-of-the-art approaches start with an off-the-shelf network, and fine-tune it with algorithmic supervision [42, 15]. They make use of geometric matching to mine matching and non-matching image pairs. They involve local feature extraction and require an expensive pre-processing of data. Furthermore, they assume that the training set contains landmarks and buildings that perform well with geometric matching of local features. We make no such assumptions nor require additional computer vision system to perform the mining. Our only assumption is that there are multiple object instances in the training set.

Incomplete, noisy, or unavailable labels are handled in various ways in the literature. In the contrastive loss paper of Hadsell *et al*. [16], the authors show that it is possible to separate different categories to different subspaces just by assigning the Euclidean nearest neighbors as positive training instances. In recent works, the labeling is guided by the information of different modalities or the data collection process. Arandjelović *et al*. [1] learn visual descriptors for location recognition by assigning labels based on the GPS location of each image. Wang and Gupta [64] sample frames from videos and assume that frames from same videos will be positive to each other. Isola *et al*. [22] group objects based on their co-occurence within the same spatial or temporal context. Similarly, learning from the spatial arrangement of the patches within an image is shown feasible [10, 35]. Finally, other cases utilize different modalities, such as learning visual descriptors from text information [14], or learning audio descriptors from unlabeled videos [2]. By contrast, we make no assumptions on the available modalities or contextual information.

Unsupervised methods and manifold learning. There are very few methods on deep metric learning that are unsupervised. Examples are two methods on learning fine-grained similarity by exploiting mutual proximity [6] and ranking [5] in the Euclidean space. Both utilize some form of clustering and splitting the training set in different groups, which is an artificial constraint, and none takes the underlying manifold structure into account. Our method is conceptually simpler and compatible with any loss function requiring positive/negative samples.

The work of Li *et al*. [30] is similar to ours in following a graph-based mining approach. In our comparisons, we show that choosing hard examples is essential and results in better performance for our approach. Recently, Pai *et*

al. [37] learn the manifold structure by encouraging pairs of points to have a Euclidean distance in the embedding space equal to the geodesic distance on the graph. We use a more relaxed objective that is compatible to most metric learning formulations and loss functions, and a manifold similarity that is more scalable than the geodesic distance.

Embeddings are learned to approximate ranking on manifolds with *fast spectral ranking* [20]. However, this approach is dataset specific and does not generalize to unseen images. Improvements on this aspect are introduced by *iterative manifold embedding* [68]. Nevertheless, the extension can handle a query image, or potentially a small set of unseen images, while a larger dataset increase significantly affects the manifold structure.

3. Method

In this section we describe our learning problem, briefly discuss the required background, and present our unsupervised training subset selection and the training process.

3.1. Problem formulation

Let $X = \{\mathbf{x}_1, \dots, \mathbf{x}_n\} \subset \mathcal{X}$ be an unordered and unlabeled collection of items, where $\mathcal{X}$ is the original *input* space of all possible items. Depending on the application, an item corresponds to an image, video, image region, local patch, *etc*. Function $f(\cdot;\theta) : \mathcal{X} \to \mathbb{R}^d$ maps an item $\mathbf{x}$ to a vector $\mathbf{z} = f(\mathbf{x}, \theta)$ in a d-dimensional *embedding* space, where θ is a set of parameters to be learned.

The input items are represented by a set of features $Y = \{\mathbf{y}_1, \dots, \mathbf{y}_n\} \subset \mathcal{Y}$, where $\mathcal{Y}$ is a *feature* space and $\mathbf{y} = g(\mathbf{x})$ for $\mathbf{x} \in X$. Depending on the application, g may be the identity mapping, *i.e*. learn directly from input space, $f(\cdot;\theta_0)$, *i.e*. learn from the same model with initial parameters θ_0, or a different model. An existing model may have been supervised (in any way) or not.

Two items are *matching* if they are considered visually similar, otherwise *non-matching*. Our goal is to learn the model parameters such that matching items are mapped to nearby points in the embedding space, while non-matching items are well separated. This corresponds to a typical metric learning scenario, and common practice is to use manually defined labels in order to construct a training set of matching and non-matching pairs of items [9, 62, 18]. In this work, we only assume that the input items X and their features Y are available at training time.

A training pair is defined w.r.t. to a *reference* (*anchor*) item $\mathbf{x}^r$. A matching pair consists of the anchor and a *positive* item $\mathbf{x}^+$. Similarly a non-matching pair consists of the anchor and a *negative* item $\mathbf{x}^-$. Alternatively, we may use a triplet $(\mathbf{x}^r, \mathbf{x}^+, \mathbf{x}^-)$. Our goal is to mine such training pairs without any supervision [9, 62, 18, 18, 34], without any complementary computer vision system [15, 42] or assumptions on the nature of the training data [64, 65, 11].

3.2. Preliminaries

By $\mathrm{NN}_k^e(\mathbf{y})$ we denote the k *Euclidean nearest neighbors* of $\mathbf{y} \in Y$, *i.e*. the k most similar items in Y according to some *Euclidean similarity* function $s_e : \mathcal{Y}^2 \to \mathbb{R}$. Similarly, by $\mathrm{NN}_k^m(\mathbf{y})$ we denote the k *manifold nearest neighbors* of $\mathbf{y} \in Y$, *i.e*. the k most similar features in Y according to a *manifold similarity* $s_m : Y^2 \to \mathbb{R}$.[1]

Given s_e, we employ a random walk on the Euclidean nearest neighbor graph G to measure the manifold similarity s_m [69]. The graph has Y as nodes. It is is undirected, weighted, represented by sparse symmetric adjacency matrix $A = (a_{ij}) \in \mathbb{R}^{n \times n}$. Edges correspond to reciprocal k-nearest neighbors, with weights

$$a_{ij} = \begin{cases} s_e(\mathbf{y}_i, \mathbf{y}_j), & \text{if } \mathbf{y}_i \in \mathrm{NN}_k^e(\mathbf{y}_j) \wedge \mathbf{y}_j \in \mathrm{NN}_k^e(\mathbf{y}_i) \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

There are no loops in the graph, *i.e*., the diagonal elements of A are zero. Starting from an arbitrary vector $\mathbf{f}^{(0)} \in \mathbb{R}^n$, the random walk for a given feature $\mathbf{y}_i \in Y$ follows the iteration

$$\mathbf{f}_i^{(t)} = \alpha \hat{A} \mathbf{f}^{(t-1)} + (1 - \alpha)\mathbf{e}_i, \tag{2}$$

where $\mathbf{e}_i$ is the i-th column of an $n \times n$ identity matrix, $\hat{A} = D^{-1/2} A D^{-1/2}$ with $D = \mathrm{diag}(A\mathbf{1})$ being the degree matrix, and $\alpha \in [0, 1)$. Iteration (2) converges to the solution $\mathbf{f}_i^\star$ of the linear system

$$(I - \alpha \hat{A})\mathbf{f} = (1 - \alpha)\mathbf{e}_i. \tag{3}$$

Following Iscen *et al*. [21], we use the conjugate gradient method to solve this system efficiently in practice, since $I - \alpha\hat{A}$ is positive-definite. We define the *manifold similarity*

$$s_m(\mathbf{y}_i, \mathbf{y}_j) = \mathbf{f}_i^\star(j), \tag{4}$$

i.e., the j-th element of $\mathbf{f}_i^\star$. Observe that s_m is symmetric because in fact $s_m(\mathbf{y}_i, \mathbf{y}_j)$ is the (i, j)-element of matrix $(1 - \alpha)(I - \alpha\hat{A})^{-1}$, which is symmetric. This matrix is dense but we never compute it; we only compute its columns as needed. For instance, given $\mathbf{y}_i \in Y$, its manifold nearest neighbors $\mathrm{NN}_k^m(\mathbf{y}_i)$ are the k maximum elements of the i-th column.

3.3. Manifold-guided selection of training samples

We are guided by the manifold similarity to select training samples. In particular, we exploit the differences between Euclidean and manifold nearest neighbors of anchor items. We first describe the selection of positives and negatives given an anchor. Then we discuss anchor selection.

[1] In fact, s_e can be any symmetric function but is only a function of two elements of $\mathcal{Y}$; while s_m is a function of two elements in Y but also of the entire set Y.

Positives. Given an anchor item $\mathbf{x}^r$ and the corresponding feature $\mathbf{y}^r = g(\mathbf{x}^r)$, which is used as query, we choose as positives the items that correspond to the manifold nearest neighbors of $\mathbf{y}^r$ that are not Euclidean neighbors. Such difference provides evidence of a matching item that is not retrieved well in the feature space, as illustrated in Figure 1(c). In the embedding space, positives should be attracted to the anchor so that Euclidean and manifold neighbors agree.

We therefore simply compare the sets $\text{NN}_k^m(\mathbf{y}^r)$ and $\text{NN}_k^e(\mathbf{y}^r)$ and select the input items that correspond to their set difference

$$P^+(\mathbf{x}^r) = \{\mathbf{x} \in X : g(\mathbf{x}) \in \text{NN}_k^m(\mathbf{y}^r) \setminus \text{NN}_k^e(\mathbf{y}^r)\} \quad (5)$$

as the *positive pool* of anchor $\mathbf{x}^r$. The value of k controls the visual diversity of positives, with larger values giving the *harder* examples. In practice, we maintain the pool ordered by descending manifold similarity, so that we may truncate by keeping the examples with highest confidence.

Mining hard positive examples, in contrast to negatives, is not common. Apart from positives being fewer than negatives, this is due to the large intraclass variability; it may result in positives that are too hard to learn. It can be achieved in cases with known geometry of the scene such that extreme cases are avoided [48, 42]. In our case, the hardness is controlled by the manifold similarity according to the current model g, so drifting into very tough examples is less likely.

Negatives. Similarly, and as illustrated in Figure 1(d), we choose as negatives the items that correspond to the Euclidean nearest neighbors of $\mathbf{y}^r$ that are not manifold neighbors. Such difference provides evidence of a non-matching item that is too close in the feature space. In the embedding space, positives should be repelled from the anchor. The *negative pool* of anchor $\mathbf{x}^r$ is defined accordingly as

$$P^-(\mathbf{x}^r) = \{\mathbf{x} \in X : g(\mathbf{x}) \in \text{NN}_k^e(\mathbf{y}^r) \setminus \text{NN}_k^m(\mathbf{y}^r)\}. \quad (6)$$

It is common practice, and known to be beneficial [49], to select hard negative examples. By construction, its size is controlled by k. Again, we maintain the pool ordered by descending Euclidean similarity to keep the hardest negative examples.

Anchors. We are interested in anchors that have many relevant images in the collection, which facilitates propagating on the manifold and discovering differences with the Euclidean neighborhood. We are also interested in anchors that are diverse, so that there is as little redundancy during training. Both conditions are satisfied by the modes of the nearest neighbor graph G, which we compute as follows.

We first compute the stationary probability distribution $\boldsymbol{\pi}$ of a random walk [8] on G. This is achieved by the power iteration method [28] yielding the leading left eigenvector of the transition matrix $P = D^{-1}A$, such that $\boldsymbol{\pi}P = \boldsymbol{\pi}$. The probability reflects the *importance* of each node in the graph, as expressed by the probability of a random walker visiting it. We find the local maxima of the stationary distribution on G and out of those, we keep a fixed number having the maximum probability. This is defined as the *anchor set* $\mathcal{A}$. This method has been previously used for image graph visualization [12].

3.4. Training

The *complete training pool* $\mathcal{P}$ is the union of the anchor set $\mathcal{A}$ and the positive and negative pools $P^+(\mathbf{x}), P^-(\mathbf{x})$ for each $\mathbf{x} \in A$. We follow the common practice in metric learning and train a network with two or three branches and use contrastive or triplet loss, respectively. All branches share weights, while the particular network architecture is application specific and is discussed in Section 4.

In both cases of contrastive or triplet loss, we form training tuples of one anchor $\mathbf{x}^r \in A$, one positive $\mathbf{x}^+ \in P^+(\mathbf{x}^r)$, and one negative item $\mathbf{x}^- \in P^-(\mathbf{x}^r)$. At each epoch, the embedding $\mathbf{z} = f(\mathbf{x}; \theta)$ for each $\mathbf{x} \in \mathcal{P}$, where θ is the current set of parameters. For each anchor $\mathbf{x}^r$, a positive item $\mathbf{x}^+$ is drawn at random from its positive pool, and one negative $\mathbf{x}^-$ is drawn at random from a subset of its negative pool. This subset consists of the items corresponding to the Euclidean nearest neighbors of $\mathbf{z}^r = f(\mathbf{x}^r; \theta)$ in the embedding space. Thus, while the manifold neighbors and the training pool are computed once at the beginning, hard negative sampling uses the current network representation. Finally, the training set for this epoch is the set of such tuples $(\mathbf{x}^r, \mathbf{x}^+, \mathbf{x}^-)$.

Given a tuple $(\mathbf{x}^r, \mathbf{x}^+, \mathbf{x}^-)$, we compute the embeddings $\mathbf{z}^r = f(\mathbf{x}^r; \theta)$ $\mathbf{z}^+ = f(\mathbf{x}^r; \theta)$ and $\mathbf{z}^- = f(\mathbf{x}^-; \theta)$, and use the *constrastive loss* [16], combining one positive and one negative pair in a single input,

$$l_c(\mathbf{z}^r, \mathbf{z}^+, \mathbf{z}^-) = \|\mathbf{z}^r - \mathbf{z}^+\|^2 + [m - \|\mathbf{z}^r - \mathbf{z}^-\|]_+^2, \quad (7)$$

or the *triplet loss* [62]

$$l_t(\mathbf{z}^r, \mathbf{z}^+, \mathbf{z}^-) = [m + \|\mathbf{z}^r - \mathbf{z}^+\|^2 - \|\mathbf{z}^r - \mathbf{z}^-\|]_+^2, \quad (8)$$

where $[\cdot]_+$ denotes the positive part and m is a *margin* parameter. We also consider a *weighted* variant of both loss functions, where the loss is multiplied by the manifold similarity $s_m(\mathbf{x}^r, \mathbf{x}^+)$ of the positive sample to the anchor. Thus we down-weigh the contribution of the tuples where the positive sample is too hard.

Of course, given the positive and negative pools, there are many more possibilities in sampling positives and negatives and forming losses that are functions of more than two or three items [29, 36, 5, 27, 59, 67]. It is also possible to iterate our approach, updating the graph and the pools based on the current embedding space, updating the embeddings,

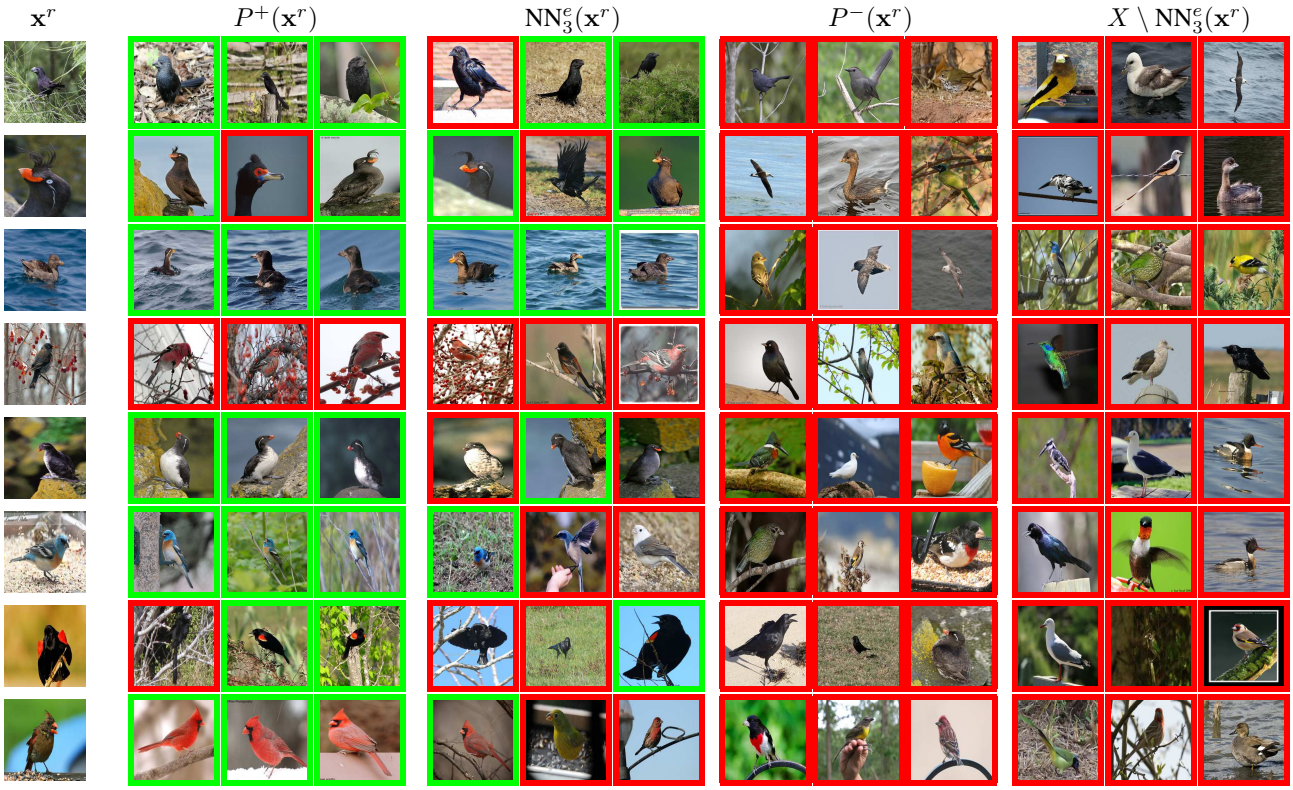


Figure 2. Sample CUB200-2011 anchor images ($\mathbf{x}^r$), positive images from our method ($P^+(\mathbf{x}^r)$) and baseline ($\mathrm{NN}_3^e(\mathbf{x}^r)$), and negative images from our method ($P^-(\mathbf{x}^r)$) and baseline ($X \setminus \mathrm{NN}_3^e(\mathbf{x}^r)$). The baseline is Euclidean nearest neighbors and non-neighbors [16]. Positive (negative) ground-truth framed in green (red). Labels are only used for qualitative evaluation and not during training.

and so on. In this case, given current parameters θ, the set $Z = \{f(\mathbf{x};\theta) : \mathbf{x} \in X\}$ plays the role of Y for the following iteration. This alternating training scheme is common for methods involving a global dataset structure like a graph or a set of clusters [44, 4, 68]. Our idea is orthogonal to most concurrent improvements on metric learning.

4. Applications

We apply the proposed method to learn visual representations on two different tasks: fine-grained categorization [66, 18] and instance-based image retrieval [15, 42]. In both cases, both the features Y and the initial model $f(\cdot,\theta)$ are based on a pre-trained model. We assume there are no labeled images available.

4.1. Fine-grained categorization

We use the CUB200-2011 dataset [61] comprising 200 bird species. The goal is to learn embeddings that better discriminate instances of the same class from instances of different classes. Following the setup of [36], the training set contains half (100) classes on which embedding is learned, while the rest are used for testing. Given a test query image, the remaining test images of the same species should be top-ranked w.r.t. Euclidean distance to the query.

All prior approaches are fully supervised and use the manually assigned labels of the training set. Our method is unsupervised, but otherwise we choose the same settings with the literature in our comparisons. In particular, we initialize the network by GoogLeNet [55] as pre-trained on ImageNet and add a fully connected layer right after the average pooling layer, reducing the embedding dimensionality to $d = 64$. We perform training with the triplet loss using all training images as anchors, which is affordable due to the small size (6k images) of the training set.

Our initial features are formed by R-MAC [58] on the last convolutional feature map of the pre-trained GoogLetNet, right before the average pooling layer, aggregated over 3 input image scales and whitened. The feature set Y contains all such vectors $\mathbf{y} = g(\mathbf{x})$ for $\mathbf{x}$ in the entire training set X. In Figure 2 we show examples of anchors and subsets of their positive and negative pools. Despite the absence of labels and the challenges of fine-grained similarity, we achieve a very clean negative pool and a reasonably clean positive one.

Figure 3. Examples of anchor images selected by the proposed method for the image retrieval application. Random samples from the top 100 (1000) anchors are shown in the top (bottom) row according to the node importance.

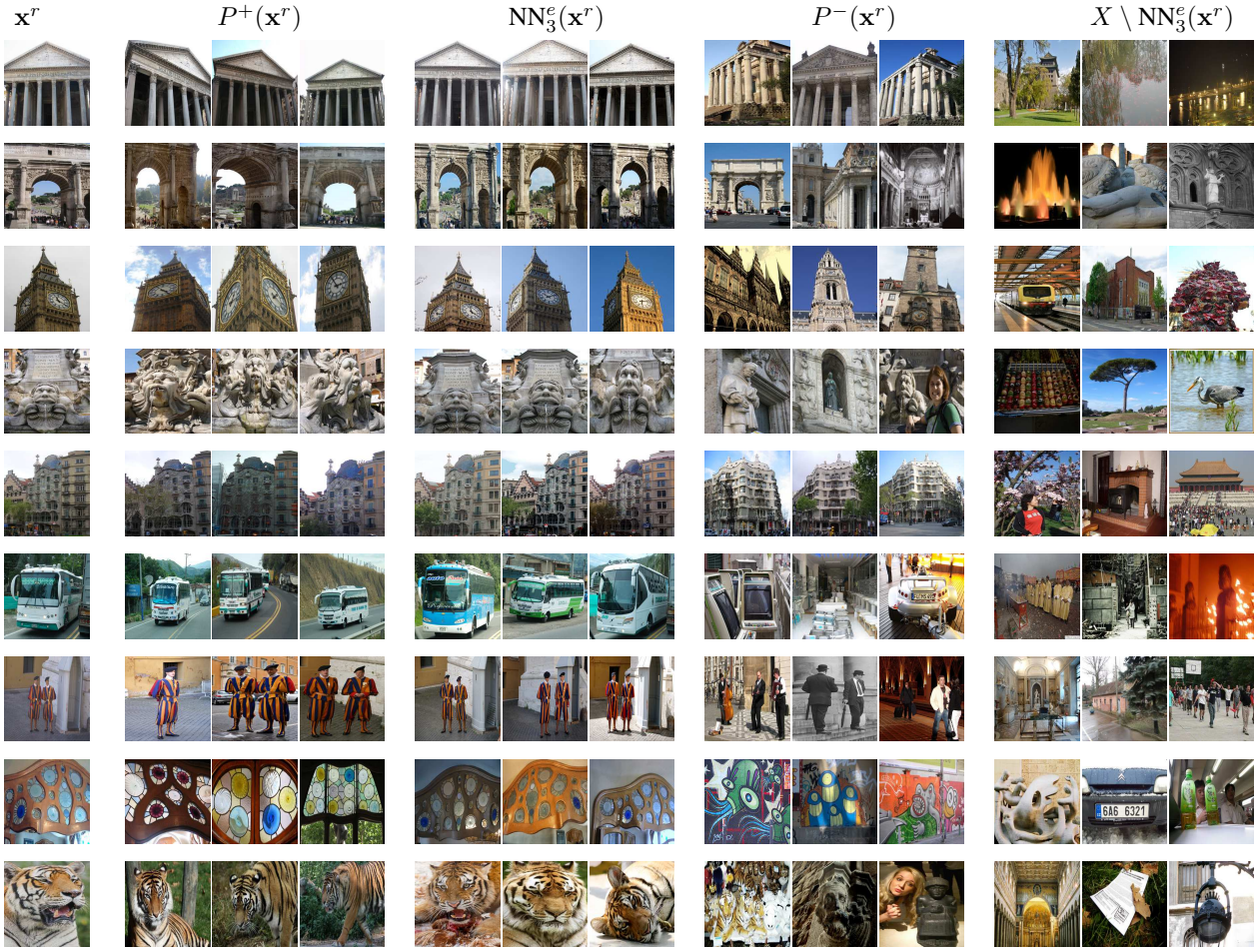


Figure 4. Sample anchor images ($\mathbf{x}^r$), positive images from our method ($P^+(\mathbf{x}^r)$) and baseline ($\mathrm{NN}_3^e(\mathbf{x}^r)$), and negative images from our method ($P^-(\mathbf{x}^r)$) and baseline ($X \setminus \mathrm{NN}_3^e(\mathbf{x}^r)$). The baseline is Euclidean nearest neighbors and non-neighbors [16].

4.2. Particular object retrieval

Particular object retrieval differs from bird species classification in that images are instances of the same object and not of the same (sub-)category, *i.e*. the similarity is even more fine-grained than in bird species. Objects are less deformable, but there is extreme diversity in viewpoint, illumination conditions, occlusion and background clutter.

Methods trained with category-level labeling do not perform well [3], while the state-of-the-art approaches use direct geometric matching [15] or Structure-from-Motion (SfM) [42] to automatically mine matching and non-matching image pairs. This is appropriate since the geometry of the scene is known, but the whole process assumes the existence of another computer vision system based on local descriptors, which is rather expensive [46].

To be comparable to the state-of-the-art fine-tuned MAC obtained through SfM by Radenovic *et al*. [42], we use similar training set and design choices. In particular, we start from the same set of 7M images, referred to as *Flickr 7M* in

the following[2], which is downloaded from Flickr with text tag queries. We limit the training set by randomly sampling 1M images. We initialize our network by VGG [50] as pre-trained on ImageNet and we fine-tune MAC representation with contrastive loss.

Our initial features are formed by R-MAC [58] on the last convolutional feature map of the pre-trained VGG network. The feature set Y contains all such vectors $\mathbf{y} = g(\mathbf{x})$ for $\mathbf{x}$ in the entire training set X. We sample an anchor set $\mathcal{A}$ and construct the complete training pool $\mathcal{P}$. Anchor selection is essential here, as using all images as queries would be very expensive. Compared to [42], our complete training pool is larger (50k vs 22k), however, we only choose 1k anchors per epoch vs 6k.

Examples of selected anchors are shown in Figure 3. They usually correspond to popular locations frequently appearing in the dataset. Examples of training pools are shown in Figure 4, comparing to a baseline where positives and negatives are Euclidean nearest neighbors and non-neighbors, respectively. The same baseline is used for comparisons in our experiments. The baseline positives contain only mild viewpoint and illumination changes, while negatives are random. On the contrary, our positives contain more challenging changes and very interesting negatives: different objects, which still look similar in one way or another. The training set variability, in terms of different objects and viewing conditions, seems a desirable property.

4.3. Implementation details

We always create graph G by considering $k = 30$ nearest neighbors in (1). Exact computation takes 80min on 12 CPU threads on the 1M retrieval training set. We use the Euclidean similarity function $s_e(\mathbf{x}_i, \mathbf{x}_j) = [\mathbf{x}_i^\top \mathbf{x}_j]_+^3$ and $\alpha = 0.99$ following Iscen *et al.* [21]. In order to create the positive pool we consider 50 neighbors in (5), while for the negative pool we use 100 and 10,000 neighbors in (6) for fine-grained categories and retrieval, respectively, due to the different size of the training dataset. We finally restrict the negative pool of each anchor to have 50 instances at most. All vector representations used for an image in the feature and the embedding space are ℓ_2-normalized.

We use stochastic gradient descent with momentum for optimization. The learning rate is initialized at 10^{-2}, and scaled by 0.1 every 10 epochs. The momentum parameter is 0.9. The margin m is set to 0.5 for triplet loss in fine-grained categorization, and to 0.7 for contrastive loss in particular object retrieval. The batch size includes 42 triplets for fine-grained categories [36] and pairs for 5 anchors in the retrieval application [42]. We train for 100 epochs on fine-grained categorization and 30 epochs on particular object retrieval experiments.

[2]Images depicting buildings that are part of the Oxford5k or Paris6k test set are removed as in [42].

Positive	Negative	CUB	Oxford5k	
Anchors		All	Random	$\mathcal{A}$
Initial		35.0	52.6	
NN_5^e	$X \backslash \text{NN}_5^e$	38.5	37.4	41.9
P^+	$X \backslash \text{NN}_5^e$	43.0	48.2	38.1
NN_5^e	P^-	42.1	57.8	71.3
P^+	P^-	43.5	64.4	73.7
$P^+ + W$	$P^- + W$	**45.3**	67.0	**76.7**

Table 1. Impact of choices of anchors and pools of positive and negative examples on Recall@1 on CUB-200-2011 and mAP on Oxford5k. On CUB, all images are used as anchors, while on Oxford5K anchors are selected either at random or by the proposed method ($\mathcal{A}$). The positive and negative pools are formed by either the baseline with Euclidean nearest neighbors (NN_5^e) [16] or our selection (P^+ and P^-), optionally with our weighted loss ($+W$).

Method	Labels	R@1	R@2	R@4	R@8	NMI
Initial	No	35.0	46.8	59.3	72.0	48.1
Triplet+semi-hard [47]	Yes	42.3	55.0	66.4	77.2	55.4
Lifted-Structure (LS) [36]	Yes	43.6	56.6	68.6	79.6	56.5
Triplet+ [18]	Yes	45.9	57.7	69.6	79.8	58.1
Clustering [53]	Yes	48.2	61.4	71.8	81.9	59.2
Triplet+++[18]	Yes	49.8	62.3	74.1	83.3	59.9
Cyclic match[30]	No	40.8	52.8	65.1	76.0	52.6
Ours	No	45.3	57.8	68.6	78.4	55.0

Table 2. Recall@k and NMI on CUB-200-2011. All methods expect for ours and cyclic match [30] use ground-truth labels during training.

5. Experiments

Fine-grained categorization is evaluated on CUB200-2011 [61], where we use the training set without labels for training and then evaluate on the test set, measuring Recall@k as well as clustering quality by NMI [31]. Particular object retrieval is evaluated by mean average precision (mAP) on a challenging and diverse set of test datasets comprising landmark and building images (Oxford5k [40] and Paris6k [41]), natural landscapes (Holidays [24]), as well as planar and 3D objects (Instre [63]). Large scale experiments are performed on Oxford and Paris by adding 100k distractors [40], namely Oxford105k and Paris106k respectively. We first evaluate the importance of different components of the selection strategy and then compare our method against state-of-the-art on each task.

5.1. Impact of positives, negatives, and anchors

We consider the unsupervised approach proposed by Hadsell *et al.* [16] as a baseline method. It sets positives to be the 5 nearest neighbors with Euclidean distance. All other elements are considered negatives out of which we randomly draw the negative pool of an anchor. In this case, hard negative mining per epoch is impossible and random choice is the only choice. We present the results in Table 1. Our method improves the pre-trained network in both tasks

Method	Representation	Labels	Oxford5k	Oxford105k	Paris6k	Paris106k	Holidays	Instre
Pre-trained [58]	MAC	ImageNet	58.5	50.3	73.0	59.0	79.4	48.5
CNN from BoW [42]		SfM	**79.7**	73.9	82.4	74.6	81.4	48.5
Ours		No	78.7	**74.3**	**83.1**	**75.6**	**82.6**	**55.5**
Pre-trained [58]	R-MAC	ImageNet	68.0	61.0	76.6	72.1	87.0	55.6
CNN from BoW [42]		SfM	77.8	70.1	84.1	76.8	84.4	47.7
Ours		No	**78.2**	**72.6**	**85.1**	**78.0**	**87.5**	**57.7**

Table 3. mAP on particular object retrieval datasets. We compare VGG as pre-trained on ImageNet, the fine-tuned network of Radenovic *et al*. [42], and our fine-tuned one. Fine-tuning is always performed for MAC, but at testing we evaluate both global MAC and regional R-MAC [58] representations.

without any supervision, while the weighted loss consistently helps. Furthermore, we observe that our hard negatives are beneficial and necessary, while our anchors are essential for the large scale training set. Given popular anchors, even simple nearest neighbors work well as positives. However, our harder positives further improve.

CUB's annotation allows us to measure the true positive and true negative rate in our positive and negative pools. These measurements are 40% and 96%, respectively. Additionally, we exploit the labels and train with a *positive (negative) oracle*, where we replace our positive (*resp*. negative) pool with one based on labels, yielding 46.7 (*resp*. 36.5) Recall@1. This shows that our hard positives with weighting are almost as good as true positives and that our hard negatives are better than hard annotated negatives. The latter is due to errors in annotation of CUB dataset, quantified to 4.4% [60]. In this case, hard negative mining frequently finds false negative examples, which are known to cause training failure [47].

5.2. Comparisons on fine-grained categories

To our knowledge there is no other unsupervised method that evaluates on CUB200-2011 dataset. We evaluate the unsupervised approach of Li *et al*. [30] by constructing the same graph as in our method, then using the provided code to construct the positive/negative pool and finally training the same way as in our method. We also compare to supervised methods that use ground-truth labels on the training set, but otherwise an identical experimental setup. As shown in in Table 2, our method competes or even outperforms fully supervised methods. We also outperform the only unsupervised competitor. Although their true positive rate is higher (76%), their positive pairs are mostly extremely similar and not challenging enough for training. Note that there are better-performing methods in the literature with sophisticated sampling schemes [67], which can be complementary to ours.

5.3. Comparisons on retrieval

We initialize by VGG as pre-trained on ImageNet and fine-tune using MAC representation. At testing, the fine-tuned network is evaluated with both global MAC and regional R-MAC representations [58]. This is the same process as [42], which makes it comparable in this respect, although the training set and sampling pool sizes are not the same as discussed in section 4.2.

Descriptor whitening is known to be essential for MAC and R-MAC. We follow the common practice in the literature and perform unsupervised PCA whitening [23] for the pre-trained networks, and supervised LDA-based whitening for the fine-tuned networks, learned on a subset of *Flickr 7M*. In particular, as supervision we use SfM labels for the network of Radenovic *et al*. [42], and matching pairs consisting of NN^m_{50} per anchor for our method.

As shown in Table 3, we improve over the pre-trained network on all test sets. Moreover, we outperform [42] with only one exception on Oxford5k. Remarkably, we perform better even on building or landmark oriented test sets, while their method specifically favors this kind of images. With our training pool being more diverse, we improve on Holidays and Instre test sets, where [42] shows little improvement or is even inferior to the pre-trained network.

6. Conclusions

In this work, we depart from using discrete category level labeling in order to learn fine grained similarity. Not only we avoid the expensive or nearly impossible manual annotation, but also do not restrict the problem to supervised classification. Our unsupervised and manifold-aware sampling of training data is applied to perform metric learning. The learning attracts points that lie on the same manifold and repels points on different manifolds. The method is conceptually simple and applicable with standard contrastive and triplet loss. It is shown to be effective for fine-grained categorization and particular object retrieval, competing or surpassing fully supervised approaches.

Acknowledgments The authors were supported by the MSMT LL1303 ERC-CZ grant.

References

[1] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *CVPR*, 2016. 2

[2] Y. Aytar, C. Vondrick, and A. Torralba. Soundnet: Learning sound representations from unlabeled video. In *NIPS*, 2016. 2

[3] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky. Neural codes for image retrieval. In *ECCV*, 2014. 1, 2, 6

[4] S.-Y. Bai, S. Agethen, T.-H. Chao, and W. Hsu. Semi-supervised learning for convolutional neural networks via online graph construction. In *arXiv*, 2015. 5

[5] M. A. Bautista, A. Sanakoyeu, and B. Ommer. Deep unsupervised similarity learning using partially ordered sets. In *arXiv*, 2017. 1, 2, 4

[6] M. A. Bautista, A. Sanakoyeu, E. Tikhoncheva, and B. Ommer. CliqueCNN: Deep unsupervised exemplar learning. In *NIPS*, 2016. 1, 2

[7] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6), 2003. 1

[8] M. Cho and K. M. Lee. Mode-seeking on graphs via random walks. In *CVPR*, 2012. 4

[9] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *CVPR*, 2005. 1, 3

[10] C. Doersch, A. Gupta, and A. A. Efros. Unsupervised visual representation learning by context prediction. In *CVPR*, 2015. 2

[11] C. Doersch and A. Zisserman. Multi-task self-supervised visual learning. In *ICCV*, 2017. 1, 3

[12] M. Douze, H. Jegou, and F. Perronnin. Polysemous codes. In *ECCV*, 2016. 4

[13] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Trans. PAMI*, 32, September 2010. 2

[14] L. Gomez, Y. Patel, M. Rusinol, D. Karatzas, and C. V. Jawahar. Self-supervised learning of visual features through embedding images into text topic spaces. In *CVPR*, 2017. 2

[15] A. Gordo, J. Almazan, J. Revaud, and D. Larlus. Deep image retrieval: Learning global representations for image search. *ECCV*, 2016. 1, 2, 3, 5, 6

[16] R. Hadsell, S. Chopra, and Y. Lecun. Dimensionality reduction by learning an invariant mapping. In *CVPR*, 2006. 2, 4, 5, 6, 7

[17] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg. MatchNet: Unifying feature and metric learning for patch-based matching. In *CVPR*, 2015. 1

[18] B. Harwood, V. Kumar B G, G. Carneiro, I. Reid, and T. Drummond. Smart mining for deep metric learning. In *ICCV*, 2017. 1, 2, 3, 5, 7

[19] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1

[20] A. Iscen, Y. Avrithis, G. Tolias, T. Furon, and O. Chum. Fast spectral ranking for similarity search. In *CVPR*, 2018. 3

[21] A. Iscen, G. Tolias, Y. Avrithis, T. Furon, and O. Chum. Efficient diffusion on region manifolds: Recovering small objects with compact cnn representations. In *CVPR*, 2017. 1, 3, 7

[22] P. Isola, D. Zoran, D. Krishnan, and E. H. Adelson. Learning visual groups from co-occurrences in space and time. In *arXiv*, 2015. 2

[23] H. Jégou and O. Chum. Negative evidences and cooccurrences in image retrieval: The benefit of PCA and whitening. In *ECCV*, October 2012. 8

[24] H. Jégou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *ECCV*, October 2008. 7

[25] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid. Aggregating local descriptors into compact codes. In *IEEE Trans. PAMI*, September 2012. 2

[26] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 1

[27] B. Kumar, G. Carneiro, I. Reid, et al. Learning local image descriptors with deep siamese and triplet convolutional networks by minimising global loss functions. In *CVPR*, 2016. 4

[28] A. N. Langville and C. D. Meyer. Deeper inside PageRank. *Internet Mathematics*, 1(3):335–380, 2004. 4

[29] M. Law, N. Thome, and M. Cord. Quadruplet-wise image similarity learning. In *ICCV*, 2013. 4

[30] D. Li, W.-C. Hung, J.-B. Huang, S. Wang, N. Ahuja, and M.-H. Yang. Unsupervised visual representation learning by graph-based consistent constraints. In *ECCV*, 2016. 2, 7, 8

[31] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008. 7

[32] B. J. Meyer, B. Harwood, and T. Drummond. Nearest neighbour radial basis function solvers for deep neural networks. In *arXiv*, 2017. 1

[33] A. Mishchuk, D. Mishkin, F. Radenovic, and J. Matas. Working hard to know your neighbor's margins: Local descriptor learning loss. In *NIPS*, 2017. 2

[34] Y. Movshovitz-Attias, A. Toshev, T. K. Leung, S. Ioffe, and S. Singh. No fuss distance metric learning using proxies. In *ICCV*, 2017. 1, 3

[35] M. Noroozi and P. Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *ECCV*, 2016. 2

[36] H. Oh Song, Y. Xiang, S. Jegelka, and S. Savarese. Deep metric learning via lifted structured feature embedding. In *CVPR*, 2016. 1, 2, 4, 5, 7

[37] G. Pai, R. Talmon, and R. Kimmel. Parametric manifold learning via sparse multidimensional scaling. *arXiv preprint arXiv:1711.06011*, 2017. 3

[38] O. M. Parkhi, A. Vedaldi, A. Zisserman, et al. Deep face recognition. In *BMVC*, 2015. 2

[39] F. Perronnin and C. R. Dance. Fisher kernels on visual vocabularies for image categorization. In *CVPR*, June 2007. 2

[40] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, June 2007. 7

[41] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *CVPR*, June 2008. 7

[42] F. Radenović, G. Tolias, and O. Chum. CNN image retrieval learns from bow: Unsupervised fine-tuning with hard examples. *ECCV*, 2016. 1, 2, 3, 4, 5, 6, 7, 8

[43] A. S. Razavian, J. Sullivan, S. Carlsson, and A. Maki. Visual instance retrieval with deep convolutional networks. *ITE Transactions on Media Technology and Applications*, 4:251–258, 2016. 2

[44] O. Rippel, M. Paluri, P. Dollar, and L. Bourdev. Metric learning with adaptive density discrimination. In *arXiv*, 2015. 1, 5

[45] L. Saul and S. Roweis. Think globally, fit locally: unsupervised learning of low dimensional manifolds. *Journal of Machine Learning Research*, 4:119–155, 2003. 1

[46] J. L. Schonberger, F. Radenovic, O. Chum, and J.-M. Frahm. From single image query to detailed 3d reconstruction. In *CVPR*, pages 5126–5134, 2015. 6

[47] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015. 2, 7, 8

[48] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *ICCV*, 2015. 4

[49] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, and F. Moreno-Noguer. Fracking deep convolutional image descriptors. In *arXiv*, 2014. 4

[50] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2014. 1, 7

[51] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *ICCV*, 2003. 2

[52] K. Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *NIPS*, pages 1857–1865, 2016. 2

[53] H. O. Song, S. Jegelka, V. Rathod, and K. Murphy. Deep metric learning via facility location. In *CVPR*, 2017. 2, 7

[54] K.-K. Sung. Learning and example selection for object and pattern detection. Technical report, MIT, 1996. 2

[55] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, 2015. 1, 5

[56] J. Tenenbaum, V. de Silva, and J. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000. 1

[57] G. Tolias, Y. Avrithis, and H. Jégou. To aggregate or not to aggregate: Selective match kernels for image search. In *ICCV*, December 2013. 2

[58] G. Tolias, R. Sicre, and H. Jégou. Particular object retrieval with integral max-pooling of cnn activations. *ICLR*, 2016. 5, 7, 8

[59] E. Ustinova and V. Lempitsky. Learning deep embeddings with histogram loss. In *NIPS*, 2016. 4

[60] G. Van Horn, S. Branson, R. Farrell, S. Haber, J. Barry, P. Ipeirotis, P. Perona, and S. Belongie. Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In *CVPR*, 2015. 8

[61] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The caltech-ucsd birds-200-2011 dataset. Technical report, California Institute of Technology, 2011. 5, 7

[62] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu. Learning fine-grained image similarity with deep ranking. In *CVPR*, 2014. 1, 3, 4

[63] S. Wang and S. Jiang. Instre: a new benchmark for instance-level object retrieval and recognition. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 11:37, 2015. 7

[64] X. Wang and A. Gupta. Unsupervised learning of visual representations using videos. In *CVPR*, 2015. 1, 2, 3

[65] X. Wang, K. He, and A. Gupta. Transitive invariance for self-supervised visual representation learning. In *arXiv*, 2017. 1, 3

[66] Y. Wang, J. Choi, V. Morariu, and L. S. Davis. Mining discriminative triplets of patches for fine-grained classification. In *CVPR*, 2016. 5

[67] C.-Y. Wu, R. Manmatha, A. J. Smola, and P. Krähenbühl. Sampling matters in deep embedding learning. In *arXiv*, 2017. 2, 4, 8

[68] J. Xu, C. Wang, C. Qi, C. Shi, and B. Xiao. Iterative manifold embedding layer learned by incomplete data for large-scale image retrieval. In *arXiv*, 2017. 3, 5

[69] D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Schölkopf. Ranking on data manifolds. In *NIPS*, 2003. 3

XXV Label propagation for deep semi-supervised learning

Title: Label propagation for deep semi-supervised learning
Authors: A. Iscen, G. Tolias, Y. Avrithis, O. Chum
Published at: CVPR 2019

Label Propagation for Deep Semi-supervised Learning

Ahmet Iscen[1] Giorgos Tolias[1] Yannis Avrithis[2] Ondřej Chum[1]
[1]VRG, FEE, CTU in Prague [2]Univ Rennes, Inria, CNRS, IRISA

Abstract

Semi-supervised learning is becoming increasingly important because it can combine data carefully labeled by humans with abundant unlabeled data to train deep neural networks. Classic methods on semi-supervised learning that have focused on transductive learning have not been fully exploited in the inductive framework followed by modern deep learning. The same holds for the manifold assumption—that similar examples should get the same prediction. In this work, we employ a transductive label propagation method that is based on the manifold assumption to make predictions on the entire dataset and use these predictions to generate pseudo-labels for the unlabeled data and train a deep neural network. At the core of the transductive method lies a nearest neighbor graph of the dataset that we create based on the embeddings of the same network. Therefore our learning process iterates between these two steps. We improve performance on several datasets especially in the few labels regime and show that our work is complementary to current state of the art.

1. Introduction

Modern approaches to many computer vision problems exploit deep neural networks. These are popular for being very efficient and providing great performance at test time. The downside is a requirement of large amounts of training examples, which are labeled either by humans or automatically on proxy tasks.

Visual data are available in large quantities, however, data reliably annotated by humans are still very scarce. Obtaining large amounts of annotated training data for every single task is not only impractical, potentially costly, but it also turns out to be error prone. The low quality of crowdsourced annotation is a common motivation to minimize the need of annotation.

In the domain of *metric learning*, promising results have been recently achieved by *unsupervised* methods for either learning from scratch or fine-tuning a supervised network for domain adaptation, which devise proxy tasks for learning. These tasks exploit the distribution of data in the original space, for instance pairwise relations of training examples [42], relations between examples and cluster centroids [1], or considering the manifold structure of data [19]. Alternatively, in *self-supervised* learning, one can take advantage of additional information like spatial layout in images [5, 12] or temporal relation in videos [40, 28]; or mine for such information in unstructured data by *algorithmic supervision* using conventional methods [13, 30]. However, most such proxy tasks are inferior when directly compared to laboriously annotated data by humans.

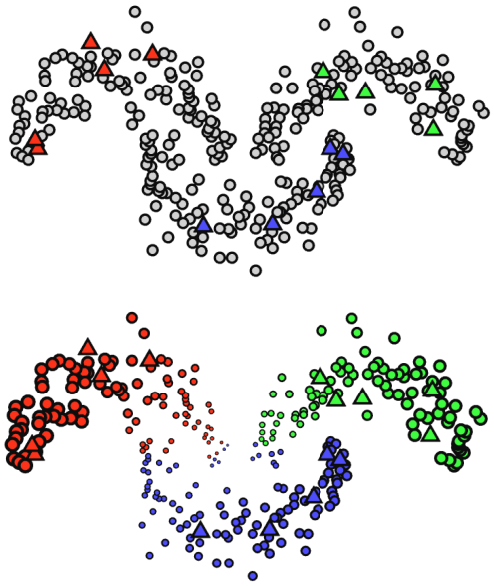


Figure 1. Label propagation on manifolds toy example. Triangles denote labeled, and circles un-labeled training data, respectively. Top: color-coded ground truth for labeled points, and gray color for unlabeled points. Bottom: color-coded pseudo-labels inferred by diffusion that are used to train the CNN. The size reflects the certainty of the pseudo-label prediction.

In classification, *semi-supervised* methods attempt to reduce the number of labeled examples, whereby the fully supervised performance on all data acts as an upper bound. In *transductive learning* [43, 45], label inference restricted to a given set of unlabeled examples is of interest. In *inductive learning*, the goal is generalization to new unseen data, while the original training data are discarded. This is achieved *e.g.* by combining classification loss on labeled

data with unsupervised objectives on all data, where the latter act as regularization [41, 38]. Or, an existing classifier can be used to assign pseudo-labels [24, 35], which is another form of algorithmic supervision. Using a powerful classifier trained on carefully annotated data can provide high-quality pseudo-labels, opening the door to learning from real unlabeled, large scale data. In such *omni-supervised* learning [31], the fully supervised performance on the labeled part is actually the lower bound. This only refreshes the interest in inductive semi-supervised methods.

In this paper, we use efficient transductive label propagation [43] to infer pseudo-labels for unlabeled data, which are used to train the classifier. Label propagation is a graph-based method, and in this work the graph is constructed exploiting the embeddings obtained by the classification network itself. Thus, the proposed method alternates between two steps. First, the network is trained from labeled and pseudo-labeled data. The second step uses the embeddings of the network trained in the previous step to construct a nearest neighbor graph. Label propagation is then used to infer pseudo-labels for unlabeled images, as well as a certainty score per image and per class. Training is performed on all data, using certainty-based weights.

We experimentally show on standard datasets that the proposed method outperforms other semi-supervised approaches. The less labeled data is available, the more pronounced the advantage of the proposed approach is.

2. Related work

The literature is rich in the problem of *semi-supervised learning* (SSL). The reader is advised to see [3] for an extensive overview. The same holds for SSL in image classification [10, 16, 4, 37]. In this section, we mostly restrict the discussion to approaches that use deep learning for SSL and perform the training on a large image collection with mini-batch optimization.

Prior work on semi-supervised deep learning for image classification is divided into two main categories. The first consists of methods, *e.g.* [15, 23, 34, 38], that add an *unsupervised loss* term (often called a regularizer) into the loss function. This term is applied to either all images or only the unlabeled ones. Methods in the second category, *e.g.* [24, 36], assign *pseudo-labels* to the unlabeled examples. The pseudo-labeled data are then used in training with a supervised loss, such as cross entropy. Both categories use a standard loss term that is trained with supervision from labeled images. A thorough evaluation of SSL deep image classification can be found in Miyato *et al.* [27].

Our contribution belongs to the second category, and is conceptually and implementation-wise orthogonal to the first. It is therefore straightforward to combine the proposed method with any method from the first category. We do combine it with [38] as shown in Section 5.

Unsupervised loss in deep SSL. Assuming that every training image, labeled or not, belongs to a single category, a natural requirement on the classifier is to make a confident prediction on the training set. This idea was formalized by Sajjadi *et al.* [35], where the regularizer is designed to minimize the entropy of the network output. Such a loss term is easily combined with other terms. A similar combination is performed for denoising auto-encoders that are applied on all images in an unsupervised manner [32].

A direction attracting a lot of attention is that of *consistency loss*, where two related cases, *e.g.* coming from two similar images, or made by two networks with related parameters, are encouraged to have similar network outputs. Sajjadi *et al.* [34] is the first, to our knowledge, to use a consistency loss between the outputs of a network on random perturbations of the same image. Laine and Aila [23] rather apply consistency between the output of the current network and the temporal average of outputs during training. The state-of-the-art *mean teacher* (MT) method [38] replaces output averaging by averaging of network parameters. Consistency loss is commonly measured by squared Euclidean distance. The Jensen-Shannon divergence is used instead by Qiao *et al.* [29], while complementarity of the two networks is enforced via adversarial examples. A similar idea is proposed by Miyato *et al.* [26].

Pseudo-labeling in deep SSL. Lee [24] uses the current network to infer pseudo-labels of unlabeled examples, by choosing the most confident class. These pseudo-labels are treated like human-provided labels in the cross entropy loss. Its impact is similar to that of entropy minimization [35]; in both cases the network is forced to have more confident predictions. The same principle is adopted by Shi *et al.* [36], where the authors further add contrastive loss to the consistency loss. Our method is different from all such prior work in that pseudo-labels are inferred by label propagation rather than network predictions.

Label propagation has been extensively used in a transductive setup (see chapter 11 [3]). Recently, Douze *et al.* [7] perform label propagation on a large image dataset with CNN descriptors for few shot learning. Unseen images are classified via online label propagation, which requires storing the entire dataset, while the network is trained in advance and descriptors are fixed. Our work is different in that we perform label propagation on the training set offline while training the network, such that inference is possible without accessing the original training set. *Learning by association* [17] can been seen as two steps of propagation on a constrained bi-partite graph between labeled and unlabeled examples. *Graph transduction game* (GTG) [9], a form of label propagation, has been used for pseudo-labels [8] as in our work, but in this case the network is pre-trained, the graph remains fixed and there is no weighting mechanism. We compare to this approach in Section 5.

3. Preliminaries

In this section we formulate the *semi-supervised learning* problem and then we discuss the classifier, different loss functions that are commonly used in prior work, and finally a transductive learning approach that our method is based on. In our experiments we use a *convolutional neural network* (CNN) to perform image classification, but this formulation applies to any network architecture in any domain.

Problem formulation. We assume a collection of n examples $X := (x_1, \ldots, x_l, x_{l+1}, \ldots, x_n)$ with $x_i \in \mathcal{X}$. The first l examples x_i for $i \in L := \{1, \ldots, l\}$, denoted by X_L, are labeled according to $Y_L := (y_1, \ldots, y_l)$ with $y_i \in C$, where $C := \{1, \ldots, c\}$ is a discrete label set for c classes. The remaining $u := n - l$ examples x_i for $i \in U := \{l+1, \ldots, n\}$, denoted by X_U, are unlabeled. The goal in SSL is to use all examples X and labels Y_L to train a classifier that maps previously unseen samples to class labels.

Classifier. The network takes an input example from $\mathcal{X}$ and produces a vector of class confidence scores. We denote it by $f_\theta : \mathcal{X} \to \mathbb{R}^c$, where θ are the network parameters. It is conceptually divided in two parts. The first is a feature extraction network $\phi_\theta : \mathcal{X} \to \mathbb{R}^d$ mapping the input to a feature vector, or descriptor. We denote the descriptor of the i-th example by $\mathbf{v}_i := \phi_\theta(x_i)$. The second typically consists of a *fully connected* (FC) layer applied on top of ϕ_θ and followed by softmax, producing a vector of *confidence scores*. Function f_θ is the mapping from input space directly to confidence scores. The output of the network for the i-th example is $f_\theta(x_i)$ and the *prediction* is the one of maximum confidence score

$$\hat{y}_i := \arg\max_j f_\theta(x_i)_j, \tag{1}$$

where subscript j denotes the j-th dimension of the vector.

Supervised loss. In supervised learning, the network is trained by minimizing a *supervised* loss term of the form

$$L_s(X_L, Y_L; \theta) := \sum_{i=1}^{l} \ell_s\left(f_\theta(x_i), y_i\right), \tag{2}$$

which applies only to labeled examples in X_L. Such term is part of the total loss when training a network in a semi-supervised setup [36, 38, 29]. A standard choice for the loss function ℓ_s in classification is *cross-entropy*, given by $\ell_s(\mathbf{s}, y) := -\log \mathbf{s}_y$ for $\mathbf{s} \in \mathbb{R}^c$ and $y \in C$.

Pseudo-labeling is the process of assigning a pseudo-label $\hat{y}_i$ to each example x_i for $i \in U$. Denoting by $\hat{Y}_U := (\hat{y}_{l+1}, \ldots, \hat{y}_n)$ the collection of pseudo-labels for X_U, the following additional *pseudo-label* loss term applies

$$L_p(X_U, \hat{Y}_U; \theta) := \sum_{i=l+1}^{n} \ell_s\left(f_\theta(x_i), \hat{y}_i\right), \tag{3}$$

where again ℓ_s is any supervised loss function like cross-entropy. An example is the approach proposed by Lee [24], who first train network f_θ with (2) and then assign pseudo-labels according to (1) for $i \in U$.

Unsupervised loss is another common alternative where the loss function applies to both labeled and unlabeled examples and encourages consistency under different transformations of the data or the network. The so-called *consistency loss* [36, 38, 36] is defined as

$$L_u(X; \theta) := \sum_{i=1}^{n} \ell_u(f_\theta(x_i), f_{\tilde{\theta}}(\tilde{x}_i)), \tag{4}$$

where $\tilde{x}_i$ refers to a different transformation of example x_i. Note that according to the standard practice of data augmentation, every forward pass of x_i during training is performed under some random transformation. Parameter set $\tilde{\theta}$ is either equal to θ or any other transformation of it, such as a moving average over the sequence of network updates [38]. A simple choice of ℓ_u is the squared Euclidean distance, *i.e.* $\ell_u(\mathbf{s}, \tilde{\mathbf{s}}) := ||\mathbf{s} - \tilde{\mathbf{s}})||^2$ for $\mathbf{s}, \tilde{\mathbf{s}} \in \mathbb{R}^c$, forcing the two outputs to be as close as possible.

Transductive learning solves a more specific problem. Instead of training a generic classifier able to classify new, yet unseen, examples, the goal is to use X and Y_L to infer labels for examples in X_U. In this work, we adopt the graph-based approach of Zhou *et al.* [43] for transductive learning by diffusion[1].

Diffusion for transductive learning [43]. Let $V = (\mathbf{v}_1, \ldots, \mathbf{v}_l, \mathbf{v}_{l+1}, \ldots, \mathbf{v}_n)$ be the descriptor set, where $\mathbf{v}_i$ corresponds to x_i as defined earlier. A symmetric *adjacency matrix* $W \in \mathbb{R}^{n \times n}$ with zero diagonal is constructed, whose elements w_{ij} are non-negative pairwise similarities between $\mathbf{v}_i$ and $\mathbf{v}_j$. Its symmetrically normalized counterpart is given by $\mathcal{W} = D^{-1/2} W D^{-1/2}$, where $D := \mathrm{diag}(W \mathbf{1}_n)$ is the *degree matrix* and $\mathbf{1}_n$ is the all-ones n-vector. A $n \times c$ *label matrix* Y is defined with elements

$$Y_{ij} := \begin{cases} 1, & \text{if } i \in L \wedge y_i = j \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

That is, the rows of Y corresponding to labeled examples are one-hot encoded labels and the rest are zero. Diffusion amounts to computing the $n \times c$ matrix

$$Z := (I - \alpha \mathcal{W})^{-1} Y, \tag{6}$$

where $\alpha \in [0, 1)$ is a parameter. Finally, the class prediction for an unlabeled example x_i is

$$\hat{y}_i := \arg\max_j z_{ij}, \tag{7}$$

where z_{ij} is the (i, j) element of matrix Z.

[1] We first present the original approach and discuss our design choices in the following section.

It is interesting to observe that matrix Z as defined by (6) is the minimizer of the following quadratic cost function

$$J(Z) := \frac{\alpha}{2} \sum_{i,j=1}^{n} w_{ij} \left\| \frac{\mathbf{z}_i}{\sqrt{d_{ii}}} - \frac{\mathbf{z}_j}{\sqrt{d_{jj}}} \right\|^2 + (1-\alpha) \left\| Y - Z \right\|_F^2 , \tag{8}$$

where $\mathbf{z}_i$ is the i-th row of matrix Z, d_{ii} is the i-th diagonal diagonal element of D and $\|\cdot\|_F$ is the Frobenius norm. The first term encourages *smoothness* such that nearby examples get the same predictions, while the second attempts to maintain predictions for the labeled examples [43].

4. Method

In the following, we begin by providing an overview of our approach. We then develop the main elements of our solution, put everything together in a concrete algorithm, and discuss how our approach is complementary to approaches using unsupervised loss for SSL [38, 36, 36]. Finally, we discuss the relation to prior work that encourages smoothness in deep networks.

Overview. We introduce a new iterative process for semi-supervised learning that can be summarized as follows. First, we construct a nearest neighbor graph and perform label propagation by transductive learning on the training set. Then, we estimate of a weight reflecting the uncertainty of label propagation for each unlabeled example. Finally, we inject the obtained labels into the network training process. These ideas are developed below, while a graphical overview of the proposed approach is shown in Figure 2.

Nearest neighbor graph. Given a network with parameters θ, we construct the descriptor set $V = (\mathbf{v}_1, \ldots, \mathbf{v}_l, \mathbf{v}_{l+1}, \ldots, \mathbf{v}_n)$, where $\mathbf{v}_i := \phi_\theta(x_i)$. A sparse *affinity matrix* $A \in \mathbb{R}^{n \times n}$ with elements

$$a_{ij} := \begin{cases} [\mathbf{v}_i^\top \mathbf{v}_j]_+^\gamma, & \text{if } i \neq j \wedge \mathbf{v}_i \in \text{NN}_k(\mathbf{v}_j) \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

is constructed, where NN_k denotes the set of k nearest neighbors in X, and γ is a parameter following recent work on manifold-based search [20]. Note that constructing the affinity matrix of the nearest neighbor graph is efficient even for large n [20], while constructing the full affinity matrix as in Zhou *et al*. is not tractable. Then, let $W := A + A^\top$, which is indeed a symmetric nonnegative adjacency matrix with zero diagonal.

Label propagation. Estimating matrix Z by (6) is impractical for large n because the inverse matrix $(I - \alpha\mathcal{W})^{-1}$ is not sparse. We rather use the the conjugate gradient (CG) method to solve linear system

$$(I - \alpha\mathcal{W})Z = Y, \tag{10}$$

which applies because matrix $(I - \alpha\mathcal{W})$ is positive-definite. This solution is known to be faster than the iterative solution of Zhou *et al*. [43], and has been used in semi-supervised learning [44], interactive image segmentation [14], image retrieval [20] and semantic image segmentation [2]. Finally, we infer the pseudo-labels $\hat{Y}_U = (\hat{y}_{l+1}, \ldots, \hat{y}_n)$, where $\hat{y}_i$ is given by (7).

Pseudo-label certainty and class balancing. Inferring pseudo-labels from matrix Z by hard assignment has two undesired effects: first, we define pseudo-labels on all unlabeled examples while clearly we do not have the same certainty for each example. Second, pseudo-labels may not be balanced over classes, which will impede learning.

To deal with the former issue we associate with each pseudo-label a weight reflecting the certainty of the prediction. We use *entropy*, as a measure of uncertainty, to assign weight ω_i to example x_i, defined by

$$\omega_i := 1 - \frac{H(\hat{\mathbf{z}}_i)}{\log(c)}, \tag{11}$$

where $\hat{Z}$ is the row-wise normalized counterpart of Z, *i.e.* $\hat{z}_{ij} = z_{ij} / \sum_k z_{ik}$, and function $H : \mathbb{R}^c \to \mathbb{R}$ is the entropy function. Weight ω_i is normalized in $[0, 1]$ because $\log(c)$ is the maximum possible entropy in $\mathbb{R}^c$.

To deal with the latter issue of class imbalance, we assign weight ζ_j to class j that is inversely proportional to class population, defined as $\zeta_j := (|L_j| + |U_j|)^{-1}$, where L_j (resp. U_j) are the examples labeled (resp. pseudo-labeled) as class j.

Given the above definitions of per-example and per-class weights, we associate the following *weighted loss* to the labeled and pseudo-labeled examples

$$\begin{aligned} L_w(X, Y_L, \hat{Y}_U; \theta) := & \sum_{i=1}^{l} \zeta_{y_i} \ell_s \left(f_\theta(x_i), y_i \right) \\ & + \sum_{i=l+1}^{n} \omega_i \zeta_{\hat{y}_i} \ell_s \left(f_\theta(x_i), \hat{y}_i \right) , \end{aligned} \tag{12}$$

which is the sum of weighted versions of L_s (2) and L_p (3). In contrast to (3), pseudo-labels originate in diffusion rather than network predictions.

A toy example showing the result of label propagation and the estimated weights is shown in Figure 3.

Iterative training. Given the above definitions of nearest neighbor graph definition, label propagation, example/class weighting and pseudo-label loss, we plug those components into an iterative learning process. We begin by randomly initializing the network parameters θ and we train the network for T epochs in a fully supervised manner on the l labeled examples X_L using the supervised loss term (2). The trained network then provides the starting point for the

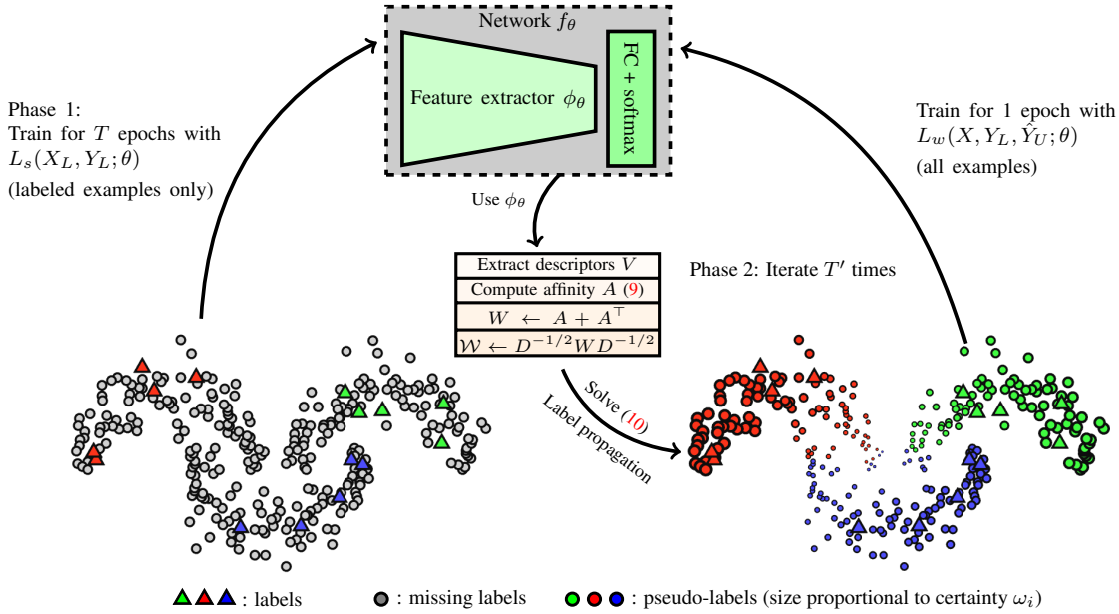


Figure 2. Overview of the proposed approach. Starting from a randomly initialized network, we first train it in a supervised fashion on the labeled examples. Then we initiate an iterative process where at each iteration we compute a nearest neighbor graph of the entire training set in the feature space of the current network, we propagate labels by transductive learning, and then we train the network on the entire training set, with true labels or pseudo-labels on the labeled or unlabeled examples respectively. The pseudo-labels are weighted per example and per class according to prediction certainty and inverse class population, respectively.

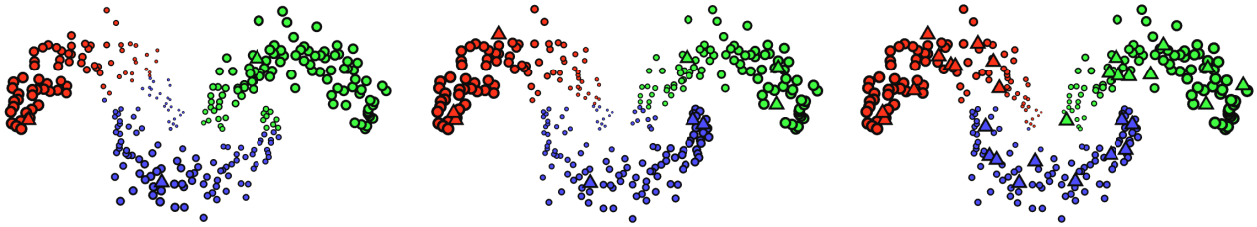


Figure 3. Toy example with 300 examples demonstrating label propagation for different number of labeled examples. Triangle markers correspond the labeled examples and circles to the unlabeled ones which are finally pseudo-labeled by label propagation. The class is color-coded and the size of the circles corresponds to weight ω_i. The true labels are the same as the example of Figure 1 (top).

following iterative process. First, we extract descriptors V on the entire training set X and compute nearest neighbors to construct the adjacency matrix W. Second, we perform label propagation by solving linear system (10) and assign pseudo-labels to unlabeled examples X_U by (7). Finally, we train the network for one epoch on the entire training set X using the weighted loss L_w (12). We repeat this iterative process for T' epochs. The above is summarized in Algorithm 1.

Procedure OPTIMIZE() refers to the mini-batch optimization of the corresponding loss term for one epoch, *i.e.* all examples are fed to the network once. More details about batch construction are given in the implementation details.

Combination with other approaches. Our contribution falls in the case of pseudo-label loss in the form of (3). It is orthogonal to approaches that use unsupervised loss, for instance (4), applied to both labeled and unlabeled examples. Combination of the two comes in a straightforward way by adding term (4) to the total loss optimized in lines 4 and 16 of Algorithm 1. This is exactly the way we combine the proposed approach with the state-of-the-art Mean-Teacher approach [38] in our experiments.

Discussion. In an inductive framework, if $\mathbf{z}_i/\sqrt{d_{ii}}$ is replaced by the network output $f_\theta(x_i)$ in the smoothness term of (8), then this becomes an unsupervised loss term, *e.g.* like (4), only now it encourages consistency between nearby example predictions. And indeed such solution is adopted *e.g.* by Weston *et al.* [41]. This is not very efficient because the adjacency matrix is typically sparse with non-zero-elements only on nearest neighbors, and then the gradient of the smoothness term will propagate from each example to its neighbors only at each iteration.

Algorithm 1 Label propagation for deep SSL

procedure LPDSSL(Training examples X, labels Y_L)
  $\theta \leftarrow$ initialize randomly
  for epoch $\in [1, \dots, T]$ **do**
    $\theta \leftarrow$ OPTIMIZE($L_s(X_L, Y_L; \theta)$) ▷ mini-batch optimization
  end for
  for epoch $\in [1, \dots, T']$ **do**
    for $i \in \{1, \dots, n\}$ **do** $\mathbf{v}_i \leftarrow \phi_\theta(x_i)$ ▷ extract descriptors
    for $(i, j) \in \{1, \dots, n\}^2$ **do** $a_{ij} \leftarrow$ affinity values (9)
    $W \leftarrow A + A^\top$ ▷ symmetric affinity
    $\mathcal{W} \leftarrow D^{-1/2} W D^{-1/2}$ ▷ symmetrically normalized affinity
    $Z \leftarrow$ solve (10) with CG ▷ diffusion
    for $(i, j) \in U \times C$ **do** $\hat{z}_{ij} \leftarrow z_{ij} / \sum_k z_{ik}$ ▷ normalize Z
    for $i \in U$ **do** $\hat{y}_i \leftarrow \arg\max_j \hat{z}_{ij}$ ▷ pseudo-label
    for $i \in U$ **do** $\omega_i \leftarrow$ certainty of $\hat{y}_i$ (11) ▷ pseudo-label weight
    for $j \in C$ **do** $\zeta_j \leftarrow (|L_j| + |U_j|)^{-1}$ ▷ class weight/balancing
    $\theta \leftarrow$ OPTIMIZE($L_w(X, Y_L, \hat{Y}_U; \theta)$) ▷ mini-batch optimization
  end for
end procedure

Our main idea therefore is that *instead of just encouraging nearby examples to get the same predictions, we encourage all examples to get predictions same as the ones we would get by transductive learning* according to the quadratic cost (8) and its solution Z (6). Computing Z is efficient because it is performed outside our main optimization process, *i.e.* it does not need iterating on mini-batches of data and backpropagating through the network. Then, given Z, the main optimization process drives all examples directly to that solution, as if they were all labeled.

5. Experiments

We present the datasets used in our experiments and the SSL setup that is followed. Then, we discuss the training details of our method and the methods reproduced for fair comparison. Finally, we perform experiments to show the impact of different components involved in the proposed method and to compare with the state of the art. All error rates reported are produced by our own implementation unless otherwise stated.

5.1. Datasets

We use three image classification datasets, namely CIFAR-10 [22], CIFAR-100 [22] and Mini-ImageNet [39]. Each dataset is used in an SSL setup where part of the training images are labeled and the rest are unlabeled. We evaluate the performance on an independent test set. Unless otherwise specified, error rate is reported in our experiments.

CIFAR-10. The training set consists of 50k images coming from 10 classes, while the test set consists of 10k images from the same 10 classes. All images have resolution 32×32. Evaluation is performed with 50, 100, 200, and 400 labeled images per classes, corresponding to $l = 500$, 1k, 2k, and 4k label images in total. We use the same random selection of labeled images that is used in Mean Teacher [38] when available (1k, 2k and 4k labels). The selection process is repeated 10 times, resulting in 10 different dataset splits for SSL on CIFAR 10. We follow the common practice which is to use each of them and report mean error and standard deviation.

CIFAR-100. Similarly to CIFAR-10, CIFAR-100 has 50k training and 10k test images of resolution 32×32, coming from 100 classes. We follow a protocol equivalent to the one of CIFAR-10. We evaluate with 40 and 100 labeled images per class, corresponding to 4k and 10k labeled images in total. There are 3 such dataset splits, mean error and standard deviation are reported.

Mini-ImageNet. We introduce an SSL evaluation setup for Mini-ImageNet [39] which is a subset of the well-known ImageNet [6] dataset and has been previously used for few-shot learning [11]. We use the train/test splits created in the work of Ravi and Larochelle [33]. It consists of 100 classes with 600 images per class, of resolution 84×84. We randomly assign 500 images from each class to the training set, and 100 images to the test set. The result is a train and test set of 50k and 10k images, respectively. We create three dataset splits for the case of 40 and 100 labeled images per class that correspond to 4k and 10k labeled images in total. Mean error and standard deviation over the three dataset splits are reported.

5.2. Training

We list the reproduced baselines, and provide training details per algorithm and dataset.

Implementation. We build our implementation on top of the publicly available Pytorch code for the Mean Teacher (MT) approach [38][2]. The fully supervised baseline and MT are reproduced identically as the original implementation. In all our experiments SGD optimization is used.

Networks. Experiments on CIFAR-10 and CIFAR-100 are performed with the "13-layer" network that is used in prior work [23, 38], while on Mini-ImageNet, Resnet-18 [18] is engaged. Both networks consist of a feature extractor ϕ_θ followed by an FC layer and softmax. We add an ℓ_2-normalization layer right after ϕ_θ (before the FC layer) providing unit-norm descriptors for the graph construction. The same choice is also adopted in the fully supervised baseline. One exception is all variants of MT as we observed that the ℓ_2-normalization layer slightly harms performance. We normalize images to have channel-wise zero mean and unit variance over the entire training set. Unlike prior work [38], we do not normalize the input images with ZCA, nor add Gaussian noise to the input layer, which result in worse performance according to our experiments.

Hyper-parameters and training choices are adapted from the MT method and implementation. These are fixed

[2] https://github.com/CuriousAI/mean-teacher/tree/master/pytorch

Pseudo-labeling	ω_i	ζ_j	CIFAR-10
Diffusion (7)			36.53 ± 1.42
		✓	36.17 ± 1.98
	✓		33.32 ± 1.53
	✓	✓	$\mathbf{32.40 \pm 1.80}$
GTG [8]	✓	✓	35.20 ± 2.23
Network (1)	✓	✓	35.17 ± 2.46

Table 1. Impact of weights ω_i, class weights ζ_j, and pseudo-labeling by diffusion prediction (7) or network prediction (1). Error rate is reported on CIFAR-10 with 500 labels.

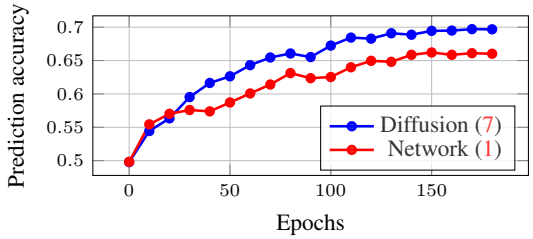


Figure 4. Accuracy of predicted pseudo-labels according to ground-truth on CIFAR-10 with 500 labeled images. Diffusion predictions (7) are compared against network predictions (1).

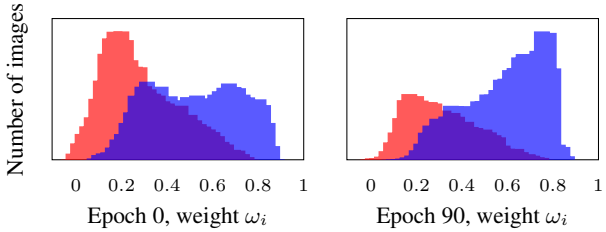


Figure 5. Distribution of weights ω_i for unlabeled images at epoch 0 (left) and epoch 90 (right) during the training of CIFAR-10 with 500 labels. Correct pseudo-labels according to ground-truth are shown in blue and incorrect in red.

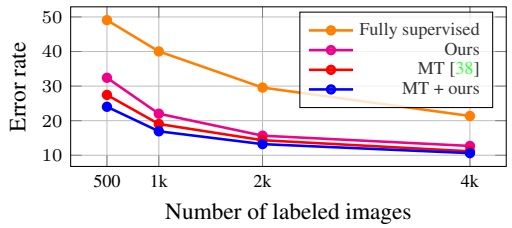


Figure 6. Error rate versus number of labeled images on CIFAR-10 using different methods.

for all approaches (re)produced by this work. The training is performed for 180 epochs in total. Initial learning rate l_0 is decayed with cosine annealing [25] so that it would have reached zero after 210 epochs, while $l_0 = 0.05$ on CIFAR-10, and $l_0 = 0.2$ on CIFAR-100 and Mini-ImageNet. Random data augmentation is performed by 4×4 random translations [38] followed by horizontal flip in CIFAR-10 and CIFAR-100. On Mini-ImageNet, each image is randomly rotated by 10 degrees before random horizontal flip. Batch

Figure 7. Examples of incorrectly pseudo-labeled images with highest ω_i in CIFAR-10. Predicted class and ω_i are shown below each image.

size is 100 for CIFAR-10 and 128 for CIFAR-100 and Mini-ImageNet. All other learning parameters remain unchanged from MT implementation.

The fully supervised approach corresponds to training with (2) and labeled images only. MT uses the additional dual output trick with coefficient 0.01. Both these approaches are reproduced.

Our approach is performed with mini-batch size $B = B_U + B_L$, where B_L images are labeled and B_U images are originally unlabeled. We set $B_L = 50$ for CIFAR-10 and $B_L = 31$ for CIFAR100 and Mini-ImageNet. Same is also applied for MT. One epoch is defined as one pass through all originally unlabeled examples in the training set, meaning that images in I_L appear multiple times per epoch. We follow the same diffusion parameters as Iscen *et al.* [20]. We set $k = 50$ for graph construction, $\gamma = 3$ in (9), and $\alpha = 0.99$ in (10). We solve (10) with at most 20 iterations of CG. Pairwise similarities for the graph are computed with the publicly available FAISS library [21]. Confidence weights ω_i are normalized over all examples s.t. $\max_i \omega_i = 1$. Class weights ζ_j are normalized over c classes such that the average class weight is 1. Pseudo-label predictions, ω_i, and ζ_j are updated after each epoch.

To assess the benefit of diffusion, we finally evaluate a variant of our approach where the pseudo-labels are not provided by diffusion but derived from the network with (1) or from GTG propagation [8] instead. Training is performed with (12), as with our method. This is in the spirit of pseudo-labeling in prior work [36, 24].

5.3. Ablation Study

We investigate the impact of different components of our method. First, we study the effectiveness of weights introduced in the loss function (12). Table 1 shows the classification performance on CIFAR-10 test set, when using only 500 labeled examples for training and the rest of the training set is considered unlabeled. Different weighting schemes are evaluated by setting all ω_i to one, all ζ_i to one, or both to one. It is shown that both weights have positive contributions. We also show the benefit of predicting with diffusion over predicting by the trained network or GTG propagation. Pseudo-labeling by the network predictions uses examples

Dataset	CIFAR-10			
Nb. labeled images	500	1000	2000	4000
Fully supervised	49.08 ± 0.83	40.03 ± 1.11	29.58 ± 0.93	21.63 ± 0.38
TDCNN [36]†	-	32.67 ± 1.93	22.99 ± 0.79	16.17 ± 0.37
Network prediction (1) + weights	35.17 ± 2.46	23.79 ± 1.31	16.64 ± 0.48	13.21 ± 0.61
Ours: Diffusion prediction (7) + weights	32.40 ± 1.80	22.02 ± 0.88	15.66 ± 0.35	12.69 ± 0.29
VAT [26]†	-	-	-	11.36
Π model [23]†	-	-	-	12.36 ± 0.31
Temporal Ensemble [23]†	-	-	-	12.16 ± 0.24
MT [38]†	-	27.36 ± 1.30	15.73 ± 0.31	12.31 ± 0.28
MT [38]	27.45 ± 2.64	19.04 ± 0.51	14.35 ± 0.31	11.41 ± 0.25
MT + Ours	$\mathbf{24.02 \pm 2.44}$	$\mathbf{16.93 \pm 0.70}$	$\mathbf{13.22 \pm 0.29}$	$\mathbf{10.61 \pm 0.28}$

Table 2. Comparison with the state of the art on CIFAR-10. Error rate is reported. "13-layer" network is used. The top part of the table corresponds to training with pseudo-labels, while the bottom part of the table includes methods that are complementary to ours, as shown by the combination of our method with MT. † denotes scores reported in prior work.

Dataset	CIFAR-100		Mini-ImageNet-*top1*		Mini-ImageNet-*top5*	
Nb. labeled images	4000	10000	4000	10000	4000	10000
Fully supervised	55.43 ± 0.11	40.67 ± 0.49	74.78 ± 0.33	60.25 ± 0.29	53.07 ± 0.68	38.28 ± 0.38
Ours	46.20 ± 0.76	38.43 ± 1.88	$\mathbf{70.29 \pm 0.81}$	57.58 ± 1.47	$\mathbf{47.58 \pm 0.94}$	36.14 ± 2.19
MT [38]	45.36 ± 0.49	36.08 ± 0.51	72.51 ± 0.22	57.55 ± 1.11	49.35 ± 0.22	32.51 ± 1.31
MT + Ours	$\mathbf{43.73 \pm 0.20}$	$\mathbf{35.92 \pm 0.47}$	72.78 ± 0.15	$\mathbf{57.35 \pm 1.66}$	50.52 ± 0.39	$\mathbf{31.99 \pm 0.55}$

Table 3. Performance comparison on CIFAR-100 and Mini-ImageNet with 4k and 10k labeled images. Error rate is reported. "13-layer" network is used for CIFAR-100 and Resnet-18 is used for Mini-ImageNet. All methods are reproduced by us.

that the network can already classify, while diffusion allows for accurate predictions beyond those examples. In Figure 4, we report the progress of the pseudo-label accuracy on unlabeled images X_U throughout the training. Diffusion predictions are consistently better than network predictions.

Figure 5 demonstrates how ω_i accurately estimates the certainty of the prediction. From the plots we observe that predictions become more accurate as the training evolves, while at the beginning most examples are misclassified. The proposed weighting mechanism is robust to incorrect pseudo-labels and prevents model collapse. Figure 7 shows some of the incorrectly pseudo-labeled images with high certainty ω_i. Most of the incorrect labels come from trucks labeled as automobiles or birds labeled as frogs.

5.4. Comparison with the state-of-the-art

We present a comparison with state-of-the-art on all 3 datasets in Tables 2 and 3. The comparison includes performance reported in prior work and our reproduced results. In the case of the work by Shi *et al.* [36], we only compare with their TDCNN variant which refers to pseudo-labeling for network training. The other loss terms in their work are complementary to ours, similarly to MT. We additionally compare with our implementation of pseudo-labeling with network predictions combined with the proposed weights.

The proposed approach performs the best out of the pseudo-label based approaches on CIFAR-10. Results in Figure 6 show that our benefit is larger when the number of labels is reduced. The results on CIFAR-10 show that our approach is complementary to unsupervised loss, such as the one used by MT. This combination achieves the best performance on this dataset. The same holds for CIFAR-100 and Mini-ImageNet for 10k available labels. Our method also achieves a lower error rate than temporal ensemble (38.65 ± 0.51) and Π-model (39.19 ± 0.36) on CIFAR-100 [23] with 10k labels. On Mini-ImageNet with 4k available labels, the best performance is achieved when using our method without combining with Mean Teacher.

6. Conclusions

Most recent approaches for deep SSL rely on training with unsupervised loss on both labeled and unlabeled images. We have proposed an approach that relies on graph-based label propagation to infer pseudo-labels for the unlabeled images. An additional training set is formed with these pseudo-labels, which are shown to be more valuable than the pseudo-labels inferred by the network itself. Our method is in principle complementary to unsupervised loss terms, which is experimentally shown in this work.

Acknowledgments This work is supported by the GAČR grant 19-23165S and the OP VVV funded project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics".

References

[1] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. *ECCV*, 2018. 1

[2] Siddhartha Chandra and Iasonas Kokkinos. Fast, exact and multi-scale inference for semantic image segmentation with deep Gaussian CRFs. In *ECCV*, 2016. 4

[3] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. *Semi-Supervised Learning*. MIT Press, 2006. 2

[4] Dengxin Dai and Luc Van Gool. Ensemble projection for semi-supervised image classification. In *ICCV*, 2013. 2

[5] Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised visual representation learning by context prediction. In *ICCV*, 2015. 1

[6] Wei Dong, Richard Socher, Li Li-Jia, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, June 2009. 6

[7] Matthijs Douze, Arthur Szlam, Bharath Hariharan, and Hervé Jégou. Low-shot learning with large-scale diffusion. In *CVPR*, 2018. 2

[8] Ismail Elezi, Alessandro Torcinovich, Sebastiano Vascon, and Marcello Pelillo. Transductive label augmentation for improved deep network learning. *arXiv preprint arXiv:1805.10546*, 2018. 2, 7

[9] Aykut Erdem and Marcello Pelillo. Graph transduction as a noncooperative game. *Neural Computation*, 24, 2012. 2

[10] Rob Fergus, Yair Weiss, and Antonio Torralba. Semi-supervised learning in gigantic image collections. In *NIPS*, 2009. 2

[11] Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *CVPR*, 2018. 6

[12] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *ICLR*, 2018. 1

[13] Albert Gordo, Jon Almazan, Jerome Revaud, and Diane Larlus. End-to-end learning of deep visual representations for image retrieval. *IJCV*, 124(2), 2017. 1

[14] Leo Grady. Random walks for image segmentation. *IEEE Trans. PAMI*, 28(11):1768–1783, 2006. 4

[15] Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. In *NIPS*, 2005. 2

[16] Matthieu Guillaumin, Jakob Verbeek, and Cordelia Schmid. Multimodal semi-supervised learning for image classification. In *CVPR*, 2010. 2

[17] Philip Haeusser, Alexander Mordvintsev, and Daniel Cremers. Learning by association – a versatile semi-supervised training method for neural networks. In *CVPR*, 2017. 2

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 6

[19] Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondrej Chum. Mining on manifolds: Metric learning without labels. In *CVPR*, 2018. 1

[20] Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, Teddy Furon, and Ondrej Chum. Efficient diffusion on region manifolds: Recovering small objects with compact cnn representations. In *CVPR*, 2017. 4, 7

[21] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*, 2017. 7

[22] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. 6

[23] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *ICLR*, 2017. 2, 6, 8

[24] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *ICMLW*, 2013. 2, 3, 7

[25] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *ICLR*, 2017. 7

[26] Takeru Miyato, Shin-ichi Maeda, Shin Ishii, and Masanori Koyama. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE Trans. PAMI*, 2018. 2, 8

[27] Avital Oliver, Augustus Odena, Colin Raffel, Ekin D Cubuk, and Ian J Goodfellow. Realistic evaluation of deep semi-supervised learning algorithms. In *ICLRW*, 2018. 2

[28] Deepak Pathak, Ross B Girshick, Piotr Dollár, Trevor Darrell, and Bharath Hariharan. Learning features by watching objects move. In *CVPR*, 2017. 1

[29] Siyuan Qiao, Wei Shen, Zhishuai Zhang, Bo Wang, and Alan Yuille. Deep co-training for semi-supervised image recognition. In *ECCV*, 2018. 2, 3

[30] Filip Radenović, Giorgos Tolias, and Ondřej Chum. Fine-tuning CNN image retrieval with no human annotation. *IEEE Trans. PAMI*, 2018. 1

[31] Ilija Radosavovic, Piotr Dollar, Ross Girshick, Georgia Gkioxari, and Kaiming He. Data distillation: Towards omni-supervised learning. In *CVPR*, 2018. 2

[32] Antti Rasmus, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. Semi-supervised learning with ladder networks. In *NIPS*, 2015. 2

[33] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *ICLR*, 2016. 6

[34] Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. Mutual exclusivity loss for semi-supervised deep learning. In *ICIP*, 2016. 2

[35] Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In *NIPS*, 2016. 2

[36] Weiwei Shi, Yihong Gong, Chris Ding, Zhiheng Ma, Xiaoyu Tao, and Nanning Zheng. Transductive semi-supervised deep learning using min-max features. In *ECCV*, 2018. 2, 3, 4, 7, 8

[37] Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Constrained semi-supervised learning using attributes and comparative attributes. In *ECCV*, 2012. 2

[38] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *NIPS*, 2017. 2, 3, 4, 5, 6, 7, 8

[39] Oriol Vinyals, Charles Blundell, Tim Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *NIPS*, 2016. 6

[40] Xiaolong Wang, Kaiming He, and Abhinav Gupta. Transitive invariance for selfsupervised visual representation learning. In *ICCV*, 2017. 1

[41] Jason Weston, Frédéric Ratle, and Ronan Collobert. Deep learning via semi-supervised embedding. In *ICML*, 2008. 2, 5

[42] Zhirong Wu, Yuanjun Xiong, Stella Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance-level discrimination. *CVPR*, 2018. 1

[43] Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. In *NIPS*, 2003. 1, 2, 3, 4

[44] Xiaojin Zhu, John Lafferty, and Ronald Rosenfeld. *Semi-Supervised Learning with Graphs*. PhD thesis, Carnegie Mellon University, Language Technologies Institute, School of Computer Science Pittsburgh, PA, 2005. 4

[45] Xiaojin Zhu, John D Lafferty, and Zoubin Ghahramani. Semi-supervised learning: From Gaussian fields to Gaussian processes. Technical report, 2003. 1

XXVI Graph convolutional networks for learning with few clean and many noisy labels

Title: Graph convolutional networks for learning with few clean and many noisy labels
Authors: A. Iscen, G. Tolias, Y. Avrithis, O. Chum, C. Schmid
Published at: ECCV 2020

Graph convolutional networks for learning with few clean and many noisy labels

Ahmet Iscen[1], Giorgos Tolias[2], Yannis Avrithis[3], Ondřej Chum[2], and Cordelia Schmid[1]

[1] Google Research
[2] VRG, Faculty of Electrical Engineering, Czech Technical University in Prague
[3] Inria, Univ Rennes, CNRS, IRISA

Abstract. In this work we consider the problem of learning a classifier from noisy labels when a few clean labeled examples are given. The structure of clean and noisy data is modeled by a graph per class and Graph Convolutional Networks (GCN) are used to predict class relevance of noisy examples. For each class, the GCN is treated as a binary classifier, which learns to discriminate clean from noisy examples using a weighted binary cross-entropy loss function. The GCN-inferred "clean" probability is then exploited as a relevance measure. Each noisy example is weighted by its relevance when learning a classifier for the end task. We evaluate our method on an extended version of a few-shot learning problem, where the few clean examples of novel classes are supplemented with additional noisy data. Experimental results show that our GCN-based cleaning process significantly improves the classification accuracy over not cleaning the noisy data, as well as standard few-shot classification where only few clean examples are used.

1 Introduction

State-of-the-art deep learning methods require a large amount of manually labeled data. The need for supervision may be reduced by decoupling representation learning from the end task and/or using additional training data that is unlabeled, weakly labeled (with noisy labels), or belong to different domains or classes. Example approaches are *transfer learning* [39], *unsupervised representation learning* [39], *semi-supervised learning* [42], *learning from noisy labels* [16] and *few-shot learning* [33].

However, for several classes, only very few or even no clean labeled examples might be available at the representation learning stage. *Few-shot learning* severely limits the number of labeled samples on the end task, while the representation is learned on a large training set of different classes [12,33,38]. Nevertheless, in many situations, more data with noisy labels can be acquired or is readily available for the end task.

One interesting mix of few-shot learning with additional large-scale data is the work of Douze *et al.* [5], where labels are propagated from few clean labeled examples to a large-scale collection. This collection is unlabeled and actually

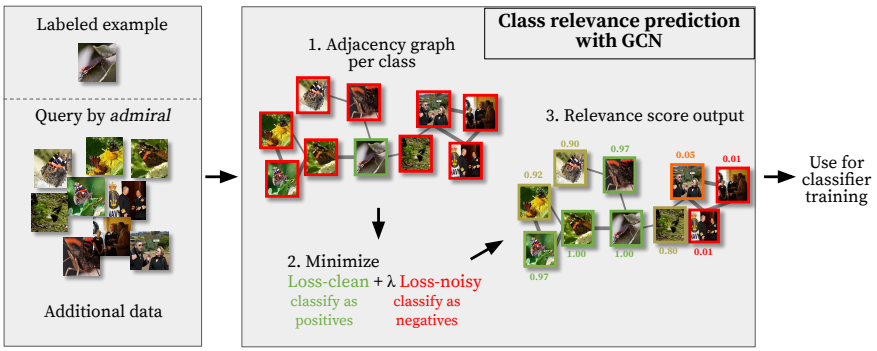


Fig. 1. Overview of our cleaning approach for 1-shot learning with noisy examples. We use the class name *admiral* to crawl noisy images from the web and create an adjacency graph based on visual similarity. We then assign a relevance score to each noisy example with a graph convolutional network (GCN). Relevance scores are displayed next to the images.

contains data for many more classes than the end task. Their method overall improves the classification accuracy, but at an additional computational cost. It is a *transductive* method, *i.e.*, instead of learning a parametric classifier, the large-scale collection is still necessary at inference.

In this work, we learn a classifier from a few clean labeled examples and additional weakly labeled data, while the representation is learned on different classes, similarly to few-shot learning. We assume that the class names are known, and we use them to search an existing large collection of images with textual description. The result is a set of images with novel class labels, but potentially incorrect (noisy). As shown in Figure 1, we clean this data using a *graph convolutional network* (GCN) [17], which learns to predict a class relevance score per image based on connections to clean images in the graph. Both the clean and the noisy images are then used to learn a classifier, where the noisy examples weighted by relevance. Unlike most existing work, our method operates independently per class and applies when clean labeled examples are few or even only one per class.

We make the following contributions:

1. We learn a classifier on a *large-scale weakly-labeled* collection jointly with only a *few clean labeled* examples.
2. To our knowledge, we are the first to use a GCN to clean noisy data: we cast a GCN as a *binary classifier* which learns to discriminate clean from noisy data, and we use its inferred "clean" probabilities as a relevance score per example.
3. We apply our method to two few-shot learning benchmarks and show significant improvement in accuracy, outperforming the method by Douze *et al.* [5] using the same large-scale collection of data without labels.

2 Related work

Learning with noisy labels is often concerned with estimating or learning a *transition matrix* [24,25,34] or *knowledge graph* [19] between labels and correcting the loss function, which does not apply in our case since the true classes in the noisy data is unknown. Most recent work on learning from large-scale weakly-labeled data focuses on learning the *representation* through *metric learning* [18,40], *bootstrapping* [28], or *distillation* [19]. In our case however, since the clean labeled examples are scarce, we need to keep the representation mostly fixed.

Dealing with the noise by thresholding [18], outlier detection [40] or *reweighting* [20], is applicable while the representation is learned, based on the gradient of the loss [30]. In contrast, the relatively-shallow GCN that we propose effectively decouples reweighting from both representation learning and classifier learning. *Learning to clean* the noisy labels [36] typically assumes adequate human verified labels for training, which again is not the case in this work.

Few-shot learning. *Meta-learning* [37] refers to learning at two levels, where generic knowledge is acquired before adapting to more specific tasks. In few-shot learning, this translates to learning on a set of *base classes* how to learn from few examples on a distinct set of *novel classes* without overfitting. For instance, *optimization meta-learning* [6,7,27] amounts to learning a model that is easy to fine-tune in few steps. In our work, we study an extension of few-shot learning where more novel class instances are available, reducing the risk of overfitting when fine-tuning the model. *Metric learning* approaches learn how to compare queries for instance to few examples [38] or to the corresponding *class prototypes* [33]. Hariharan and Girshick [12] and Wang *et al.* [41] learn how to *generate* novel-class examples, which is not needed when more data is actually available.

Gidaris and Komodakis [9] learn on base classes a simpler cosine similarity-based parametric classifier, or simply *cosine classifier*, without meta-learning. The same classifier has been introduced independently by Qi *et al.* [26], who further fine-tune the network, assuming access to the base class training set. A recent survey [3] confirms the superiority of the cosine classifier to previous work including meta-learning [6]. We use the cosine classifier in this work, both for base and novel classes.

Making use of *unlabeled data* has been little explored in few-shot learning until recently. Ren *et al.* [29] introduce a semi-supervised few-shot classification task, where some labels are unknown. Liu *et al.* [21] follow the same semi-supervised setup, but use graph-based *label propagation* (LP) [45] for classification and consider all test images jointly. These methods assume a meta-learning scenario, where only small-scale data is available at each training episode; arguably, such a small amount of data limits the representation adaptation and generalization to unseen data. Similarly, Rohrbach *et al.* [31] use label propagation in a *transductive* setting, but at a larger scale assuming that all examples come from a set of known classes. Douze *et al.* [5] extend to an even larger scale,

leveraging 100M unlabeled images in a graph without using additional text information. We focus on the latter large-scale scenario using the same 100M dataset. However, we filter by text to obtain *noisy data* and follow an *inductive* approach by training a classifier for novel classes, such that the 100M collection is not needed at inference.

Graph neural networks are generalizations of convolutional networks to non-Euclidean spaces [1]. Early *spectral methods* [2,14] have been succeeded by *Chebyshev polynomial* approximations [4], which avoid the high computational cost of computing eigenvectors. *Graph convolutional networks* (GCN) [17] provide a further simplification by a *first-order* approximation of graph filtering and are applied to *semi-supervised* [17] and subsequently *few-shot learning* [8]. Kipf and Welling [17] apply the loss function to labeled examples to make predictions on unlabeled ones. Similarly, Garcia and Bruna [8] use GCNs to make predictions on novel class examples. Gidaris and Komodakis [10] use Graph Neural Networks as denoising autoencoders to generate class weights for novel classes. By contrast, we cast GCNs as *binary classifiers* discriminating clean from noisy examples: we apply a loss function to all examples, and then use the inferred probabilities as a class relevance measure, effectively cleaning the data.

Our counter-intuitive objective of treating all noisy examples as negative can be compared to treating each example as a different class in *instance-level discrimination* [43]. In fact, our loss function is similar to *noise-contrastive estimation* (NCE) [11]. Our experiments show that our GCN-based classifier outperforms classical LP [45] used for a similar purpose by [31].

3 Problem formulation

We consider a space $\mathcal{X}$ of examples. We are given a set $X_{\mathcal{C}} \subset \mathcal{X}$ of examples, each having a *clean* (manually verified) label in a set C of classes with $|C| = K$. We assume that the number $|X^c_{\mathcal{C}}|$ of examples[4] labeled in each class $c \in C$ is only k, typically in $\{1, 2, 5, 10, 20\}$. We are also given an additional set $X_{\mathcal{N}}$ of examples, each with a set of *noisy* labels in C. The *extended* set of examples for class c is now $X^c_{\mathcal{E}} = X^c_{\mathcal{C}} \cup X^c_{\mathcal{N}}$. Examples or sets of examples having clean (noisy) labels are referred to as clean (noisy) as well. The goal is to train a K-way classifier, using the additional noisy set in order to improve the accuracy compared to only using the small clean set.

We assume that we are given a feature extractor $g_\theta : \mathcal{X} \to \mathbb{R}^d$, mapping an example to a d-dimensional vector. For instance, when examples are images, the feature extractor is typically a *convolutional neural network* (CNN) and θ are the parameters of all layers.

In this work, we assume that the noisy set $X_{\mathcal{N}}$ is collected via web crawling. Examples are images accompanied by free-form text descriptions and/or user tags originating from community photo collections. To make use of text data, we assume that the names of the classes in C are given. An example in $X_{\mathcal{N}}$ is

[4] For any set $X \subset \mathcal{X}$, we denote by X^c its subset of examples labeled in class $c \in C$.

given a label in class $c \in C$ if its textual information contains the name of class c; it may then have none, one or more labels. In this way, we automatically infer labels for $X_{\mathcal{N}}$ without human effort, which are however *noisy*.

4 Cleaning with graph convolutional networks

We perform cleaning by predicting a *class relevance* measure for each noisy example in $X^c_{\mathcal{N}}$, independently for each class $c \in C$. To simplify the notation, we drop superscript c where possible in this subsection and we denote $X^c_{\mathcal{E}}$ by $\{x_1, \ldots, x_k, x_{k+1}, \ldots, x_N\}$, where $X^c_{\mathcal{C}} = \{x_1, \ldots, x_k\}$ and $X^c_{\mathcal{N}} = \{x_{k+1}, \ldots, x_N\}$. The features of these examples are similarly represented by matrix $V = [\mathbf{v}_1, \ldots, \mathbf{v}_k, \mathbf{v}_{k+1}, \ldots, \mathbf{v}_N] \in \mathbb{R}^{d \times N}$, where $\mathbf{v}_i = g_\theta(x_i)$ for $i = 1, \ldots, N$.

We construct an affinity matrix $A \in \mathbb{R}^{N \times N}$ with elements $a_{ij} = [\mathbf{v}_i^\top \mathbf{v}_j]_+$ if examples $\mathbf{v}_i$ and $\mathbf{v}_j$ are reciprocal nearest neighbors in $X^c_{\mathcal{E}}$ and 0 otherwise. Matrix A has zero diagonal, but self-connections are added before A is normalized as $\tilde{A} = D^{-1}(A + I)$ with $D = \mathrm{diag}((A + I)\mathbf{1})$ being the degree matrix of $A + I$ and $\mathbf{1}$ the all-ones vector.

Graph convolutional networks (GCNs) [17] are formed by a sequence of layers. Each layer is a function $f_\Theta : \mathbb{R}^{N \times N} \times \mathbb{R}^{l \times N} \to \mathbb{R}^{n \times N}$ of the form

$$f_\Theta(\tilde{A}, Z) = h(\Theta^\top Z \tilde{A}), \tag{1}$$

where $Z \in \mathbb{R}^{l \times N}$ represents the input features, $\Theta \in \mathbb{R}^{l \times n}$ holds the parameters of the layer to be learned, and h is a nonlinear activation function. Function f_Θ maps l-dimensional input features to n-dimensional output features.

In this work we consider a two-layer GCN with a scalar output per example. This network is a function $F_\Theta : \mathbb{R}^{N \times N} \times \mathbb{R}^{d \times N} \to \mathbb{R}^N$ given by

$$F_\Theta(\tilde{A}, V) = \sigma(\Theta_2^\top [\Theta_1^\top V \tilde{A}]_+ \tilde{A}), \tag{2}$$

where $\Theta = \{\Theta_1, \Theta_2\}$, $\Theta_1 \in \mathbb{R}^{d \times m}$, $\Theta_2 \in \mathbb{R}^{m \times 1}$, $[\cdot]_+$ is the positive part or ReLU function [23] and $\sigma(a) = (1 + e^{-a})^{-1}$ for $a \in \mathbb{R}$ is the sigmoid function. Function F_Θ performs feature propagation through the affinity matrix in an analogy to classical graph-based propagation methods for classification [45] or search [46].

The output $F_\Theta(\tilde{A}, V)$ is a vector of length N, with element $F_\Theta(\tilde{A}, V)_i$ in $[0, 1]$ representing a relevance value of example x_i for class c. To learn the parameters Θ, we treat the GCN as a *binary classifier* where target output 1 corresponds to clean examples and 0 to noisy. In particular, we minimize the loss function

$$L_{\mathcal{G}}(V, \tilde{A}; \Theta) = -\frac{1}{k} \sum_{i=1}^{k} \log\Big(F_\Theta(\tilde{A}, V)_i\Big) - \frac{\lambda}{N-k} \sum_{i=k+1}^{N} \log\Big(1 - F_\Theta(\tilde{A}, V)_i\Big). \tag{3}$$

This is a binary cross-entropy loss function where noisy examples are given an importance weight λ. Given the propagation on the nearest neighbor graph, and depending on the relative importance λ of the second term, noisy examples that are strongly connected to clean ones are still expected to receive high class

relevance, while noisy examples that are not relevant to the current class are expected to get a class relevance near zero.

The impact of parameter λ is validated in Section 6, where we show that the fewer the available clean images are (smaller k) the smaller the importance weight should be. As is standard practice for GCNs in classification [17], training is performed in batches of size N, that is the entire set of features.

Figure 2 shows examples of clean images, corresponding noisy ones and the predicted relevance. Using the visual similarity to the clean image, we can use relevance to resolve cases of polysemy, *e.g. black widow (spider) vs. black widow (superhero)*, or cases like *pineapple vs. pineapple juice*.

Discussion. Loss function (3) is similar to *noise-contrastive estimation* (NCE) [11] as used by We *et al.* [43] for *instance-level discrimination*, whereas we discriminate clean from noisy examples. The semi-supervised learning setup of GCNs [17] uses a loss function that applies only to the labeled examples, and makes discrete predictions on unlabeled examples. In our case, all examples contribute to the loss but with different importance, as we infer real-valued class relevance for the noisy examples, to be used for subsequent learning.

Function F_Θ in (2) reduces to a Multi-Layer Perceptron (MLP) when the affinity matrix A is zero, in which case all examples are disconnected. Using an MLP to perform cleaning would take each example into account independently of the others, while the GCN considers the collection of examples as a whole. MLP training is performed identically to GCN by minimizing (3). We compare the two alternatives in our experiments.

5 Learning a classifier with few clean and many noisy examples

Our cleaning process applies when the clean labeled examples are few, but assumes a feature extractor[5] g_θ. That is, representation learning, label cleaning and classifier learning are decoupled. We perform GCN-based cleaning as described in Section 4, and learn a classifier by weighting examples according to class relevance. The process of training the classifier is described below.

5.1 Cosine-similarity based classifier

We use a *cosine-similarity based classifier* [9,26], or *cosine classifier* for short. Each class $c \in C$ is represented by a learnable parameter $\mathbf{w}_c \in \mathbb{R}^d$. The *prediction* of example $x \in \mathcal{X}$ is the class c of maximum cosine similarity $\hat{\mathbf{w}}_c^\top \hat{g}_\theta(x)$[6]

$$\pi_{\theta,W}(x) = \arg\max_c \hat{\mathbf{w}}_c^\top \hat{g}_\theta(x), \tag{4}$$

where $W = [\mathbf{w}_1, \ldots, \mathbf{w}_K] \in \mathbb{R}^{d \times K}$.

[5] For instance, after training on a different task or a set of classes other than C. Learning of the feature extractor used in our experiments is described in Appendix.

[6] We denote the ℓ_2-normalized counterpart of vector $\mathbf{x}$ by $\hat{\mathbf{x}}$. Similarly, if $\mathbf{y} = f(x)$, we denote $\hat{\mathbf{y}}$ by $\hat{f}(x)$.

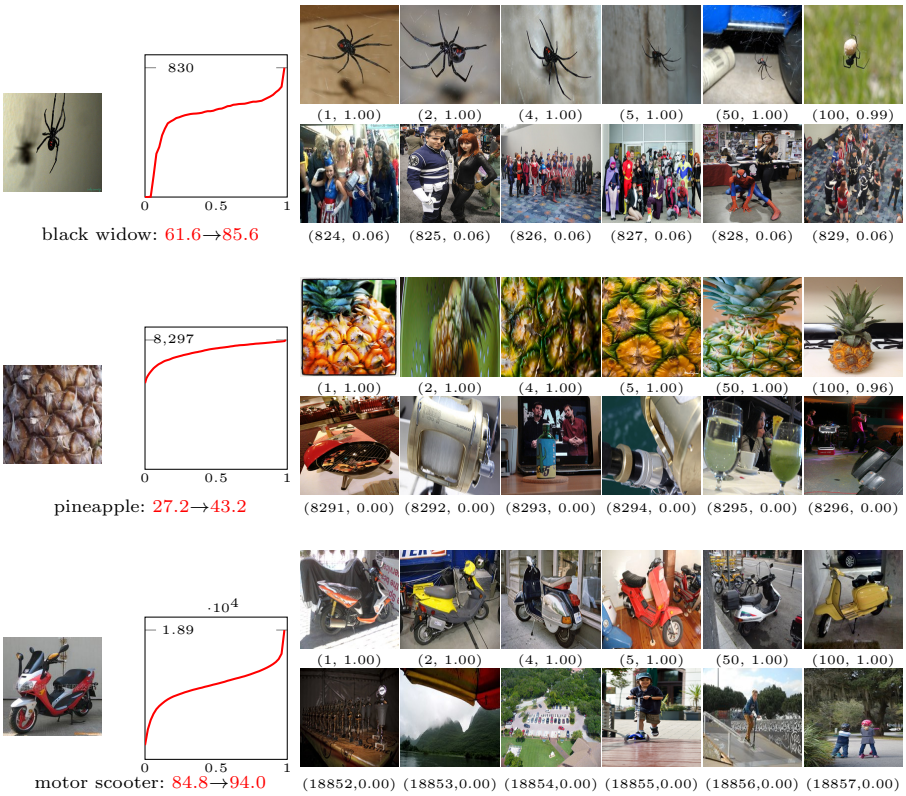


Fig. 2. Examples of clean images from the Low-Shot ImageNet Benchmark (left) for 1-shot classification, cumulative histogram of the predicted relevance for noisy images (middle) and representative noisy images (right) by descending order of relevance, with relevance value reported below. Test accuracy without and with additional data using class prototypes (5) is shown next to class names.

5.2 Classifier learning

The goal is to learn a K-way classifier for unseen data in $\mathcal{X}$. Unlike the typical few-shot learning task, each class contains a few clean and many noisy examples.

Prior to learning classifiers, training examples $x_i \in X_{\mathcal{E}}^c$ are weighted by their relevance $r(x_i)$ to class c. For a noisy example $x_i \in X_{\mathcal{N}}^c$, we define $r(x_i) = F_\Theta(\tilde{A}, V)_i$ where $F_\Theta(\tilde{A}, V)$ is the output vector of the GCN, while for a clean example $x_i \in X_{\mathcal{C}}^c$ we fix $r(x_i) = 1$. Note that optimizing (3) does not guarantee $F_\Theta(\tilde{A}, V)_i = 1$ for clean examples $x_i \in X_{\mathcal{C}}^c$. We define $r(X) = \sum_{x \in X} r(x)$ for any set $X \subset \mathcal{X}$.

We first assume that the given feature extractor is fixed and consider two different classifiers, namely class prototypes and cosine-similarity based classifier.

Then, this assumption is dropped and the classifier and feature representation are learned jointly by fine-tuning the entire network.

Class prototypes. For each class $c \in C$, we define *prototype* $\mathbf{w}_c$ by

$$\mathbf{w}_c = \frac{1}{r(X_{\mathcal{E}}^c)} \sum_{x \in X_{\mathcal{E}}^c} r(x) g_\theta(x). \tag{5}$$

Prototypes are fixed vectors, not learnable parameters. Collecting them into matrix $W = [\mathbf{w}_1, \ldots, \mathbf{w}_K] \in \mathbb{R}^{d \times K}$, K-way prediction is made by classifier $\pi_{\theta,W}$ (4).

Cosine classifier learning. Given examples $X_{\mathcal{E}}$, we learn a parametric cosine classifier with parameters $W = [\mathbf{w}_1, \ldots, \mathbf{w}_K] \in \mathbb{R}^{d \times K}$ by minimizing the weighted cross entropy loss $L(C, X_{\mathcal{E}}, \theta; W)$ over W, given by

$$L(C, X_{\mathcal{E}}, \theta; W) = -\sum_{c \in C} \frac{1}{r(X_{\mathcal{E}}^c)} \sum_{x \in X_{\mathcal{E}}^c} r(x) \log(\boldsymbol{\sigma}(s\hat{W}^\top \hat{g}_\theta(x))_c), \tag{6}$$

where $\boldsymbol{\sigma} : \mathbb{R}^K \to \mathbb{R}^K$ is the softmax function with $\boldsymbol{\sigma}(\mathbf{a})_c = e^{a_c} / \sum_{j \in C} e^{a_j}$ for $\mathbf{a} \in \mathbb{R}^K$, s is a *scale parameter* and $\hat{W} = [\hat{\mathbf{w}}_1, \ldots, \hat{\mathbf{w}}_K]$. The parameters θ of the feature extractor are fixed. The scale parameter s is also fixed according to the training of the feature extractor. Prediction is made as in the previous case.

Deep network fine-tuning. An alternative is to drop the assumption that the feature extractor is fixed. In this case, we jointly learn the parameters θ of the feature extractor and W of the K-way cosine classifier by minimizing the right-hand side of (6). This requires access to examples $X_{\mathcal{E}}$, while for the previous two classifiers, access to features $g_\theta(x)$ is enough. Note that, due to over-fitting on the few available examples, such learning is typically avoided in a few-shot learning setup.

6 Experiments

6.1 Experimental setup

Datasets and task setup. We extend the *Low-Shot ImageNet benchmark* [12] by assuming many noisy examples in addition to the few clean ones. In this benchmark, the 1000 ImageNet classes [32] are split into 389 base classes and 611 novel classes. The validation set contains 193 base and 300 novel classes, and the test set the remaining 196 base and 311 novel classes. The base classes are used to learn the feature extractor (see supplementary material), while the novel classes form the set of classes C on which we apply the cleaning and learn the classifier. We only assume noisy examples for the novel classes, not for the base ones. Additionally, we apply a similar setup to the Places365 dataset [44]. We randomly choose 183 test and 182 validation classes. We use the model learned on the base classes of Low-Shot ImageNet benchmark as the feature extractor.

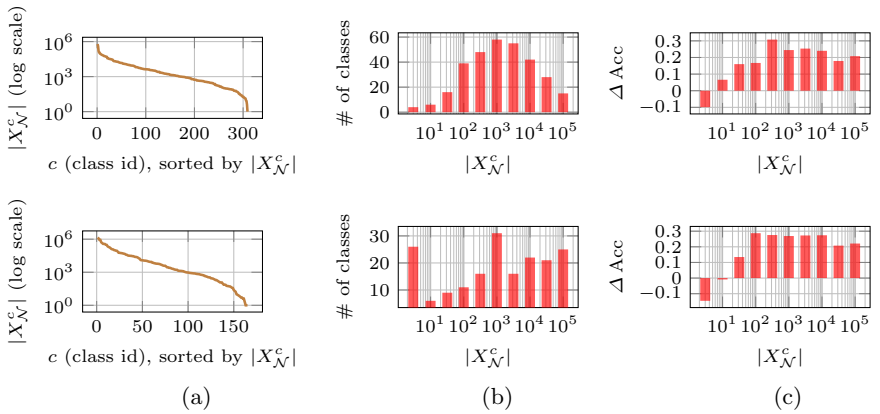


Fig. 3. Noisy data statistics for Low-Shot ImageNet(top) and Low-Shot Places365(bottom). (a) Number of additional images collected from YFCC-100M per class c for all novel classes. (b) Number of classes per group, when groups are created according to $|X^c_{\mathcal{N}}|$ in logarithmic scale. (c) Accuracy improvement Δ Acc (difference of accuracy between our method with noisy examples and the baseline without noisy examples) for prototype classifier, for same groups as in (b).

Therefore, all classes in Places365 dataset are considered *novel*. We refer to this setup as *Low-Shot Places365 benchmark*.

The standard benchmark includes k-shot classification, *i.e.* classification on k clean examples per class, with $k \in \{1, 2, 5, 10, 20\}$. We extend it to k clean and many noisy examples per class. Similar to the work of Hariharan and Girshick [12], we perform 5 episodes, each drawing a subset of k clean examples per class. We report the average top-5 accuracy over the 5 episodes on novel classes of the test set. Accuracy over all classes (base and novel) is reported in supplementary material.

Noisy data and statistics. We use the YFCC100M dataset [35] as a source of additional data with noisy labels. It contains approximatively 100M images collected from Flickr. Each image comes with a text description obtained from the user title and caption. We use the text description to obtain images with noisy labels, as discussed in Section 3.

Figure 3 (top) shows the statistics of noisy examples for Low-Shot ImageNet benchmark. The noisy examples for novel classes are long tailed in log scale (a). Noisy examples per class differ significantly for different classes, with a minimum of zero for classes *maillot* and *missile*, and a maximum of 620,142 for the class *church/church building*. There is a significant number of classes where we obtain less than 1000 extra examples, but we improve nevertheless; see Figure 3 (c). A small exception is 4 very rare classes out of 311, with around 3 additional images per class (leftmost bin in Figure 3 (b) and (c)). One could use more resources like web queries to collect additional data in real world applications.

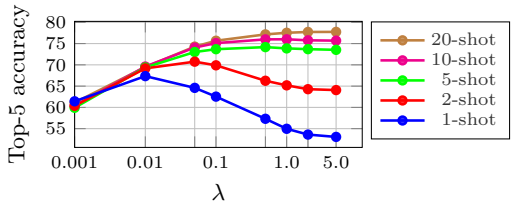


Fig. 4. Impact of λ on the validation set of the extended Low-shot ImageNet benchmark with YFCC-100M for noisy examples using class prototypes (5).

Figure 3 (bottom) presents the same statistics for Low-Shot Places365 benchmark. The trends are similar to those from Figure 3, except that there are more than 20 classes without any additional noisy data in this task (b). Nevertheless, there is a more consistent improvement in accuracy for classes that do have sufficient noisy data(c).

Representation and classifier learning. In most experiments, we use ResNet-10 [13] as feature extractor as in [9]. Classification for novel classes is performed with *class prototypes* (5), *cosine classifier learning* (6) or *deep network fine-tuning*. Hyper-parameters, such as batch size and number of epochs, are tuned on the validation set. We cross-validate the possible values of 512, 1024, 2048, 4096, and 8192 for batchsize and 10, 20, 30 and 50 for number of epochs. The learning rate starts from 0.1 and is reduced to 0.001 at the end of training with *cosine annealing* [22]. We handle the imbalance of the noisy set by normalizing by $r(X_c)$ in (6). Prototypes (5) are used to initialize the weights W of the cosine classifier in (6). We ignore examples x_i with relevance $r(x_i) < 0.1$ to reduce the complexity when fine-tuning the network. We also report results with ResNet-50 as feature extractor, using the model trained on base classes by [12]. Following [5], we apply PCA to the features to reduce their dimensionality to 256. Base classes are represented by class prototypes (5) in this case.

GCN training is performed with the Adam optimizer and a learning rate of 0.1 for 100 iterations. We use dropout with probability 0.5. The dimensionality of the input descriptors is $d = 512$ for ResNet-10 and $d = 256$ for ResNet-50 (after PCA). Dimensionality of the internal representation in (1) is $m = 16$. The affinity matrix is constructed with reciprocal top-50 nearest neighbors.

Baseline methods. We implement and evaluate several baseline methods:

1. *β-cleaning* assigns $r(x_i) = \beta$ to all additional examples. We report results for $\beta = 1.0$ (unit relevance score) and β^*, the optimal β for all k obtained on the validation set.
2. *Similarity* uses the scaled cosine similarity as the relevance weight, *i.e.* $r(x_i) = (1+\mathbf{v}_i^\top \mathbf{x})/2$, where $\mathbf{x}$ is the class prototype created with the features of clean examples.
3. *Linear* learns a linear binary classifier where the positive instances are the k labeled examples, and the negative examples are chosen randomly from other classes.

Method	k=1	2	5	10	20
			Few Clean Examples		
Class proto. [9]	45.3±0.65	57.1±0.37	69.3±0.32	74.8±0.20	77.8±0.24
			Few Clean & Many Noisy Examples		
Similarity	49.8±0.29	56.3±0.27	64.2±0.32	68.4±0.14	71.2±0.12
β-weighting, $\beta = 1$	56.1±0.06	56.4±0.08	57.1±0.05	57.7±0.08	58.7±0.06
β-weighting, β^*	55.6±0.24	58.3±0.14	63.4±0.25	67.5±0.34	71.0±0.22
Linear	59.8±0.00	59.3±0.00	58.4±0.00	58.6±0.00	59.4±0.00
Label Propagation	62.6±0.35	67.0±0.41	74.6±0.30	76.3±0.23	77.7±0.18
MLP	63.6±0.41	68.8±0.42	73.7±0.25	75.6±0.21	77.6±0.21
Ours	67.8±0.10	70.9±0.30	73.9±0.17	76.1±0.12	78.2±0.14

Table 1. Comparison with baselines using noisy examples on the Low-shot ImageNet benchmark. We report top-5 accuracy on novel classes with classification by class prototypes (5).

4. *Label Propagation* (LP) [45] propagates information by a linear operation. It solves the linear system $(I - \alpha D^{-1/2} A D^{-1/2})\mathbf{r}_c = \mathbf{y}_c$ [15] for each class c, where D is the degree matrix of A, $\alpha = 0.9$ and $\mathbf{y}_c \in \mathbb{R}^N$ is a k-hot binary vector indicating the clean (labeled) examples of class c. Relevance $r(x_i)$ is the i-th element $(\mathbf{r}_c)_i$ of the solution.
5. *MLP*, discussed in Section 4, learns a nonlinear mapping to assign relevance weights, but does not propagate over the graph. It is trained using (3) and therefore includes part of our contribution.

6.2 Experimental results

The impact of the importance weight λ is measured on the validation set and the best performing value is used on the test set for each value of k. Results on Low-shot ImageNet benchmark are shown in Figure 4. The larger the value of λ, the more the loss encourages noisy examples to be classified as negatives. As a consequence, large (small) λ results in smaller (larger) relevance, on average, for noisy examples. The optimal λ per value of k suggests that as the number of clean examples decreases, the need for additional noisy ones increases.

Comparison with baselines using additional data on Low-shot Imagenet benchmark is presented in Table 1. Qualitative results are presented in Figure 2. The use of additional data is mostly harmful for β-weighting except for 1 and 2-shot. MLP offers improvements in most cases, implying that it manages to appropriately down-weight irrelevant examples. The consistent improvement of GCN compared to MLP, especially large for small k, suggests that it is beneficial to incorporate relations, with the affinity matrix A modeling the structure of the feature space. LP is a classic approach that also uses A but is a linear operation with no parameters, and is inferior to our method. The gain of cleaning ($\beta = 1$ *vs.* ours) ranges from 11% to 20%.

Comparison with the state of the art on Low-shot Imagenet benchmark is presented in Table 2. We significantly improve the performance by using additional data and cleaning compared to a number of different approaches, including

Method	Top-5 accuracy on novel classes				
	k=1	2	5	10	20
ResNet-10 – Few Clean Examples					
Proto.-Nets [33]†	39.3	54.4	66.3	71.2	73.9
Logistic reg. w/ H [41]†	40.7	50.8	62.0	69.3	76.5
PMN w/ H [41]†	45.8	57.8	69.0	74.3	77.4
Class proto. [9]	45.3±0.65	57.1±0.37	69.3±0.32	74.8±0.20	77.8±0.24
Class proto. w/ Att. [9]	45.8±0.74	57.4±0.38	69.6±0.27	75.0±0.29	78.2±0.23
ResNet-10 – Few Clean & Many Noisy Examples					
Ours - class proto. (5)	**67.8**±0.10	**70.9**±0.30	**73.7**±0.20	**76.1**±0.16	**78.2**±0.14
Ours - cosine (6)	**73.2**±0.14	**75.3**±0.25	**75.6**±0.24	**78.5**±0.32	**80.7**±0.26
Ours - fine-tune	**74.1**±0.19	**76.2**±0.28	**77.7**±0.23	**80.6**±0.31	**82.6**±0.24
ResNet-50 – Few Clean Examples					
Proto.-Nets [33]†	49.6	64.0	74.4	78.1	80.0
PMN w/ H [41]†	54.7	66.8	77.4	81.4	83.8
ResNet-50 – Few Clean & Many Unlabeled Examples					
Diffusion [5]†	63.6±0.61	69.5±0.60	75.2±0.40	78.5±0.34	80.8±0.18
Diffusion - logistic [5]†	64.0±0.70	71.1±0.82	79.7±0.38	83.9±0.10	86.3±0.17
ResNet-50 – Few Clean & Many Noisy Examples					
Ours - class proto. (5)	**69.7**±0.44	**73.7**±0.56	**77.0**±0.20	**79.9**±0.30	**81.9**±0.29
Ours - cosine (6)	**78.0**±0.38	**80.2**±0.33	**80.9**±0.17	**83.7**±0.19	**85.7**±0.11
Ours - fine-tune	**80.2**±0.33	**82.6**±0.14	**83.3**±0.26	**85.9**±0.22	**88.3**±0.21

Table 2. Comparison to the state of the art on the Low-shot ImageNet benchmark. We report top-5 accuracy on novel classes. We use class prototypes (5), cosine classifier learning (6) and deep network fine-tuning for classification with our GCN-based data addition method. † denotes numbers taken from the corresponding papers. All other experiments are re-implemented by us.

the work by Gidaris and Komodakis [9], which is our starting point. As expected, the gain is more pronounced for small k, reaching more than 20% improvement for 1-shot novel accuracy.

Closest to ours is the work by Douze *et al.* [5], who use the same experimental setup and the same additional data, but without filtering by text or using noisy labels. We outperform this approach in all cases, while requiring much less computation: *offline*, we construct a separate small graph per class rather than a single graph over the entire 100M collection; *online*, we perform inference by cosine similarity to one prototype per class or a learned classifier rather than iterative diffusion on the entire collection. By ignoring examples that are not given any noisy label, we are only using a tiny fraction of the 100M collection: in particular, only 3,744,994 images for the 311-class test split of the Low-shot ImageNet benchmark. In contrast to [5], additional data brings improvement even at 20-shot with classifier learning or network fine-tuning. Most importantly, our approach does not require the entire 100M collection at inference.

Analysis of relevance weights. We manually label all the noisy examples from 20 classes in order to quantitatively measure the accuracy of the assigned relevance. We measure the noise ratio per class, *i.e.* the ratio of irrelevant (negative) noisy images to all noisy (positive and negative) images. Positive and negative

Method	k=1	2	5	10	20
Few Clean Examples					
Class proto. [9]	28.7±1.12	38.0±0.37	50.5±0.51	57.9±0.35	62.3±0.25
Few Clean & Many Noisy Examples - Class proto. (5)					
β-weighting, $\beta = 1$	44.0±0.34	45.7±0.22	48.4±0.31	50.0±0.12	50.8±0.25
Label Propagation	39.6±0.78	46.5±0.22	54.8±0.42	59.6±0.11	62.0±0.14
MLP	46.9±0.78	50.1±0.38	55.4±0.29	59.2±0.26	61.5±0.31
Ours	47.1±0.70	50.5±0.31	55.1±0.50	59.0±0.32	61.9±0.22
Few Clean & Many Noisy Examples - Other classifiers					
Ours - cosine (6)	50.7±0.61	53.5±0.49	57.0±0.54	59.8±0.22	62.3±0.12
Ours - fine-tune	51.8±0.69	54.8±0.57	59.5±0.63	62.9±0.39	66.0±0.27

Table 3. Comparison with baselines using noisy examples on the Low-shot Places365 benchmark. We report top-5 accuracy on novel classes.

images are defined according to the manual labels. The 20 classes are selected such that 10 of them have the highest 1-shot accuracy, and the rest have the lowest. This allows us to examine success and failure cases.

In the case of 1-shot classification ($k = 1$), the average relevance weight is 0.71 for positive and 0.40 for negative examples. A success case is the "bee eater" class with noise ratio equal to 0.52. Our method achieves 98.4% accuracy for 1-shot classification, compared to 68.8% without any additional data. The average relevance weight is 0.99 for positive examples and 0.25 for negative examples of this class. One failure case is the "muzzle" class; it corresponds to the muzzle of an animal. The noise ratio is high; 94% of the 980 collected images are not relevant with most being animals without a muzzle or a firearm. The 1-shot classification accuracy without noisy data is 4%. Our method offers only a small increase to 8%. This can be explained by inaccurate relevance weights, which are on average 0.18 for positive and 0.30 for negative examples.

Experiments in Low-Shot Places365 are reported in Table 3. Our results indicate that our method consistently outperforms the baselines on this benchmark as well. Note that *MLP*, which is also competitive for this task, is trained with our proposed loss function 3. This is our contribution as well as the use of GCN. These methods significantly improve over existing methods, such as Label Propagation [45]. Further improvements are brought by cosine classifier learning (6) and deep network fine-tuning.

We also present qualitative results on Low-Shot Places365 in Figure 5. The first example at the top shows that top-ranked images by relevance depict different views of cafeterias for the *cafeteria* class, while bottom-ranked images depict food served in a cafeteria, which are irrelevant to our task. Similarly, our method assigns high relevance to images of soccer stadiums and low relevance to soccer players for the *soccer* class. Finally, our method finds similar images to the clean example for the *ruin* class. In general, top-ranking images exhibit diversity and are not just near-duplicates.

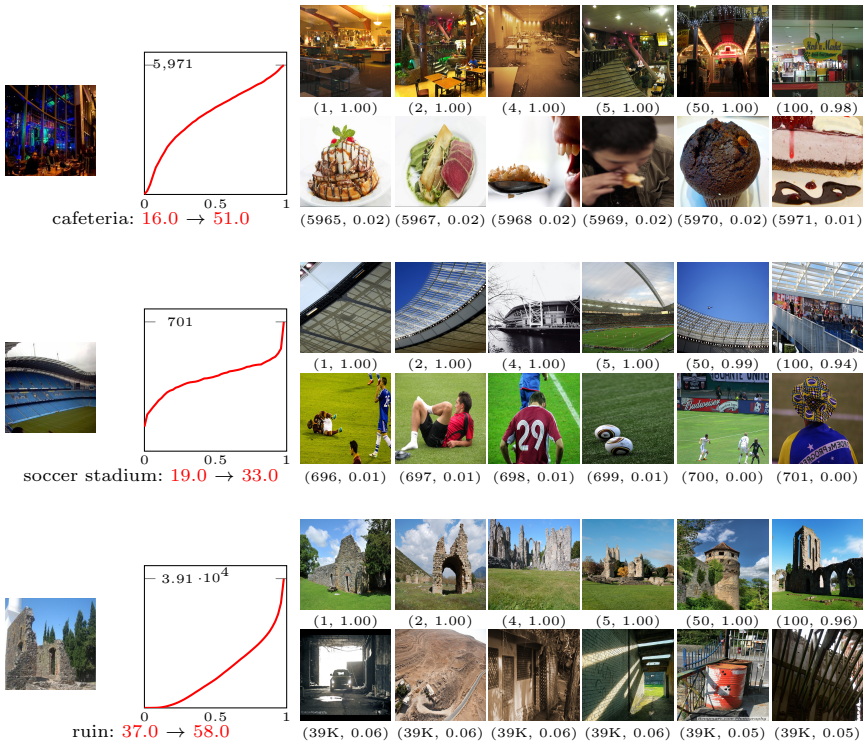


Fig. 5. Examples of clean images on Low-Shot Places365 Benchmark (left) for 1-shot classification, cumulative histogram of the predicted relevance for noisy images (middle), and representative noisy images (right), each having its position in the descending ranked list according to relevance value reported below. Test accuracy without and with additional data using prototypes (5) is shown next to class names.

7 Conclusions

In this paper we have introduced a new method for assigning class relevance to noisy images obtained by textual queries with class names. Our approach leverages one or a few labeled images per class and relies on a graph convolutional network (GCN) to propagate visual information from the labeled images to the noisy ones. The GCN is trained as a binary classifier discriminating clean from noisy examples using a weighted binary cross-entropy loss function and inferring "clean" probability as a relevance score for that class. Experimental results show that using noisy images weighted by this relevance score significantly improves the classification accuracy.

Acknowledgements. This work is funded by MSMT LL1901 ERC-CZ grant and OP VVV funded project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics".

References

1. Bronstein, M.M., Bruna, J., LeCun, Y., Szlam, A., Vandergheynst, P.: Geometric deep learning: going beyond euclidean data. IEEE Signal Processing Magazine **34**(4), 18–42 (2017)
2. Bruna, J., Zaremba, W., Szlam, A., Lecun, Y.: Spectral networks and locally connected networks on graphs. In: ICLR (2014)
3. Chen, W.Y., Liu, Y.C., Kira, Z., Wang, Y.C.F., Huang, J.B.: A closer look at few-shot classification. ICLR (2019)
4. Defferrard, M., Bresson, X., Vandergheynst, P.: Convolutional neural networks on graphs with fast localized spectral filtering. In: NeurIPS (2016)
5. Douze, M., Szlam, A., Hariharan, B., Jégou, H.: Low-shot learning with large-scale diffusion. In: CVPR (2018)
6. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: ICML (2017)
7. Finn, C., Xu, K., Levine, S.: Probabilistic model-agnostic meta-learning. In: NeurIPS (2018)
8. Garcia, V., Bruna, J.: Few-shot learning with graph neural networks. In: ICLR (2018)
9. Gidaris, S., Komodakis, N.: Dynamic few-shot visual learning without forgetting. In: CVPR (2018)
10. Gidaris, S., Komodakis, N.: Generating classification weights with gnn denoising autoencoders for few-shot learning. In: CVPR (2019)
11. Gutmann, M., Hyvärinen, A.: Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In: International Conference on Artificial Intelligence and Statistics (2010)
12. Hariharan, B., Girshick, R.: Low-shot visual recognition by shrinking and hallucinating features. In: CVPR (2017)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
14. Henaff, M., Bruna, J., LeCun, Y.: Deep convolutional networks on graph-structured data. arXiv preprint arXiv:1506.05163 (2015)
15. Iscen, A., Tolias, G., Avrithis, Y., Furon, T., Chum, O.: Efficient diffusion on region manifolds: Recovering small objects with compact cnn representations. In: CVPR (2017)
16. Joulin, A., van der Maaten, L., Jabri, A., Vasilache, N.: Learning visual features from large weakly supervised data. In: ECCV (2016)
17. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: ICLR (2017)
18. Lee, K.H., He, X., Zhang, L., Yang, L.: CleanNet: Transfer learning for scalable image classifier training with label noise. In: CVPR (2018)
19. Li, Y., Yang, J., Song, Y., Cao, L., Luo, J., Li, L.J.: Learning from noisy labels with distillation. In: ICCV (2017)
20. Liu, T., Tao, D.: Classification with noisy labels by importance reweighting. IEEE Trans. PAMI **38**(3) (2015)
21. Liu, Y., Lee, J., Park, M., Kim, S., Yang, E., Hwang, S.J., Yang, Y.: Learning to propagate labels: Transductive propagation network for few-shot learning. In: ICLR (2019)
22. Loshchilov, I., Hutter, F.: Sgdr: Stochastic gradient descent with warm restarts. ICLR (2017)

23. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: ICML (2010)
24. Natarajan, N., Dhillon, I.S., Ravikumar, P.K., Tewari, A.: Learning with noisy labels. In: NeurIPS (2013)
25. Patrini, G., Rozza, A., Krishna Menon, A., Nock, R., Qu, L.: Making deep neural networks robust to label noise: A loss correction approach. In: CVPR (2017)
26. Qi, H., Brown, M., Lowe, D.G.: Low-shot learning with imprinted weights. In: CVPR (June 2018)
27. Ravi, S., Larochelle, H.: Optimization as a model for few-shot learning. In: ICLR (2017)
28. Reed, S., Lee, H., Anguelov, D., Szegedy, C., Erhan, D., Rabinovich, A.: Training deep neural networks on noisy labels with bootstrapping. ICLR (2015)
29. Ren, M., Ravi, S., Triantafillou, E., Snell, J., Swersky, K., Tenenbaum, J.B., Larochelle, H., Zemel, R.S.: Meta-learning for semi-supervised few-shot classification. In: ICLR (2018)
30. Ren, M., Zeng, W., Yang, B., Urtasun, R.: Learning to reweight examples for robust deep learning. In: ICML (2018)
31. Rohrbach, M., Ebert, S., Schiele, B.: Transfer learning in a transductive setting. In: NeurIPS (2013)
32. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. IJCV **115**(3) (2015)
33. Snell, J., Swersky, K., Zemel, R.: Prototypical networks for few-shot learning. In: NeurIPS (2017)
34. Sukhbaatar, S., Bruna, J., Paluri, M., Bourdev, L., Fergus, R.: Training convolutional networks with noisy labels. arXiv preprint arXiv:1406.2080 (2014)
35. Thomee, B., Shamma, D.A., Friedland, G., Elizalde, B., Ni, K., Poland, D., Borth, D., Li, L.J.: Yfcc100m: The new data in multimedia research. Commun. ACM **59**(2) (2016)
36. Veit, A., Alldrin, N., Chechik, G., Krasin, I., Gupta, A., Belongie, S.: Learning from noisy large-scale datasets with minimal supervision. In: CVPR (July 2017)
37. Vilalta, R., Drissi, Y.: A perspective view and survey of meta-learning. Artificial intelligence review **18**(2), 77–95 (2002)
38. Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al.: Matching networks for one shot learning. In: NeurIPS (2016)
39. Wang, X., Gupta, A.: Unsupervised learning of visual representations using videos. In: CVPR (2015)
40. Wang, Y., Liu, W., Ma, X., Bailey, J., Zha, H., Song, L., Xia, S.T.: Iterative learning with open-set noisy labels. In: CVPR (2018)
41. Wang, Y.X., Girshick, R., Hebert, M., Hariharan, B.: Low-shot learning from imaginary data. In: CVPR (2018)
42. Weston, J., Ratle, F., Collobert, R.: Deep learning via semi-supervised embedding. In: ICML (2008)
43. Wu, Z., Xiong, Y., Yu, S., Lin, D.: Unsupervised feature learning via non-parametric instance-level discrimination. CVPR (2018)
44. Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., Torralba, A.: Places: A 10 million image database for scene recognition. IEEE Trans. PAMI **40**(6) (2017)
45. Zhou, D., Bousquet, O., Lal, T.N., Weston, J., Schölkopf, B.: Learning with local and global consistency. In: NeurIPS (2003)
46. Zhou, D., Weston, J., Gretton, A., Bousquet, O., Schölkopf, B.: Ranking on data manifolds. In: NeurIPS (2003)

Supplementary Material: Graph convolutional networks for learning with few clean and many noisy labels

Anonymous ECCV submission

Paper ID 1060

A The role of base classes

The proposed method is applicable with any given feature extractor. Herein, we describe the learning of the feature extractor on a set of base classes according to a standard few-shot learning setup and benchmark [12]. Then, we describe the extended classifiers to the union of all classes, *i.e.* base classes and novel classes, which are the ones used in Section 5.

A.1 Representation learning on base classes

We are given a set $X_{\mathcal{B}} \subset \mathcal{X}$ of examples, each having a clean label in a set of *base classes* $C_{\mathcal{B}}$ with $|C_{\mathcal{B}}| = K_{\mathcal{B}}$. Base classes $C_{\mathcal{B}}$ are disjoint from C, which are also known as novel classes. These data are used to learn a feature representation, *i.e.* a feature extractor g_θ, by learning a $K_{\mathcal{B}}$-way base-class classifier for unseen data in $\mathcal{X}$. The parameters θ of the feature extractor and $W_{\mathcal{B}}$ of the classifier are jointly learned by minimizing the cross entropy loss

$$L_{\mathcal{B}}(C_{\mathcal{B}}, X_{\mathcal{B}}; \theta, W_{\mathcal{B}}) = -\sum_{c \in C_{\mathcal{B}}} \frac{1}{|X_{\mathcal{B}}^c|} \sum_{x \in X_{\mathcal{B}}^c} \log(\boldsymbol{\sigma}(s\hat{W}_{\mathcal{B}}^\top \hat{g}_\theta(x))_c). \tag{1}$$

The learned feature extractor parameters θ and the learned scale parameter s are used by our method as described Sections 4 and 5.

A.2 Classification on all classes

The classifier parameters $W_{\mathcal{B}}$ are used, combined with classifier parameters W learned as described in Section 5, for classification on *all classes* $C_{\mathcal{A}} = C \cup C_{\mathcal{B}}$.

Class prototypes. The concatenated parameter matrix $W_{\mathcal{A}} = [W_{\mathcal{B}}, W]$ is used for $K_{\mathcal{A}}$-way prediction on all (base and novel) classes by $\pi_{\theta, W_{\mathcal{A}}}$, where $K_{\mathcal{A}} = K + K_{\mathcal{B}}$. $W_{\mathcal{B}}$ is learned according to $L_{\mathcal{B}}(C_{\mathcal{B}}, X_{\mathcal{B}}; \theta, W_{\mathcal{B}})$ (1), while W is learned according to (5).

Cosine classifier learning. Prediction on all classes is made as in the previous case, but W is learned according to (6).

Deep network fine-tuning. We now assume that base class examples are accessible too and, given all examples $X_{\mathcal{A}} = X_{\mathcal{B}} \cup X_{\mathcal{E}}$, we jointly learn the parameters θ of the feature extractor and $W_{\mathcal{A}} = [W_{\mathcal{B}}, W]$ of the $K_{\mathcal{A}}$-way cosine classifier for all classes by minimizing loss function

$$L_{\mathcal{A}}(C_{\mathcal{A}}, X_{\mathcal{A}}; \theta, W_{\mathcal{A}}) = L_{\mathcal{B}}(C_{\mathcal{B}}, X_{\mathcal{B}}; \theta, W_{\mathcal{B}}) + L(C, X_{\mathcal{E}}; \theta, W). \tag{2}$$

Method	Top-5 accuracy on all classes				
	k=1	2	5	10	20
	ResNet-10 – Few Clean Examples				
Proto.-Nets [33]†	49.5	61.0	69.7	72.9	74.6
Logistic reg. w/ H [41]†	54.4	61.0	69.0	73.7	76.5
PMN w/ H [41]†	40.8	49.9	64.2	71.9	76.9
Class proto. [9]	57.0±0.36	64.7±0.16	72.5±0.18	75.8±0.16	77.4±0.19
Class proto. w/ Att. [9]	58.1±0.48	65.2±0.15	72.9±0.25	76.6±0.18	78.8±0.16
	ResNet-10 – Few Clean & Many Noisy Examples				
Ours - class proto. (5)	**70.3±0.05**	**72.1±0.18**	**74.1±0.12**	**75.6±0.13**	**76.9±0.09**
Ours - cosine (6)	**72.4±0.07**	**73.4±0.21**	**77.2±0.20**	**78.8±0.21**	**79.2±0.17**
Ours - fine-tune	**76.0±0.10**	**77.3±0.13**	**78.7±0.19**	**80.7±0.25**	**82.2±0.14**
	ResNet-50 – Few Clean Examples				
Proto.-Nets [33]†	61.4	71.4	78.0	80.0	81.1
PMN w/ H [41] †	65.7	73.5	80.2	82.8	84.5
	ResNet-50 – Few Clean & Many Noisy Examples				
Ours - class proto. (5)	**73.8±0.33**	**76.6±0.36**	**78.9±0.19**	**80.8±0.21**	**82.2±0.14**
Ours - cosine (6)	**78.2±0.25**	**79.6±0.23**	**80.4±0.18**	**82.4±0.19**	**84.1±0.09**
Ours - fine-tune	**81.6±0.20**	**83.2±0.16**	**84.3±0.23**	**86.2±0.17**	**87.8±0.03**

Table 1. Comparison to the state of the art on the Low-shot ImageNet benchmark. We report top-5 accuracy on all classes. We use class prototypes (5), cosine classifier learning (6) and deep network fine-tuning for classification with our GCN-based data addition method. † denotes numbers taken from the corresponding papers. All other experiments are re-implemented by us.

Note that in contrast to (6), the last term of (2) optimizes parameters θ too. As mentioned earlier, such learning is typically avoided in a few-shot learning setup. In few cases, it takes the form of fine-tuning including all base class data [26], or only lasts for a few iterations when the base class data is not accessible [6].

A.3 Results on all classes

We report the accuracy over all classes in Table 1. When fine-tuning the network by (2), the learned W is used to initialize the corresponding part of $W_{\mathcal{A}}$ and we train all layers for 10 epochs with learning rate 0.01. The results indicate that our method still brings significant improvements when all classes are used.

B Results on Mini-Imagenet

We evaluate the proposed method on another popular benchmark, *i.e.* few-shot learning on Mini-ImageNet [38]. The dataset is a subset of ImageNet [32], and contains 100 different classes, split into 64 base, 16 validation and 20 test classes [27]. Each class contains 600 images that are re-sized to a resolution of 84×84. We use the ConvNet-128 model with cosine classifier, following [9]. Novel categories are classified using class prototypes (5).

Method	k=1	k=5
Few Clean Examples		
Class proto. [9]	$54.2_{\pm 0.77}$	$71.2_{\pm 0.61}$
Class proto. w/ Att. [9]	$56.2_{\pm 0.81}$	$72.9_{\pm 0.62}$
Few Clean & Many Noisy Examples - Class proto. (5)		
β-weighting, $\beta = 1$	$63.5_{\pm 0.77}$	$65.2_{\pm 0.81}$
Label Propagation	$67.0_{\pm 0.74}$	$74.8_{\pm 0.61}$
MLP	$65.9_{\pm 0.78}$	$73.9_{\pm 0.63}$
Ours	$68.2_{\pm 0.76}$	$74.7_{\pm 0.59}$

Table 2. Comparison with baselines using noisy examples on the Mini-ImageNet dataset. We report the accuracy for 5-way k-shot experiments where $k = 1$ and $k = 5$.

Table 2 shows the accuracy on Mini-Imagenet for the 5-way k-shot classification scenario with $k = 1$ and $k = 5$. We report the average accuracy over 600 trials along with the confidence interval. Our method brings significant improvements for $k = 1$, showing its generalization across different few-shot datasets and benchmarks.

XXVII Revisiting Oxford and Paris: Large-Scale Image Retrieval Benchmarking

Title: Revisiting Oxford and Paris: Large-Scale Image Retrieval Benchmarking
Authors: F. Radenović, A. Iscen, G. Tolias, Y. Avrithis, O. Chum
Published at: CVPR 2018

Revisiting Oxford and Paris: Large-Scale Image Retrieval Benchmarking

Filip Radenović[1] Ahmet Iscen[1] Giorgos Tolias[1] Yannis Avrithis[2] Ondřej Chum[1]
[1]VRG, FEE, CTU in Prague [2]Inria Rennes

Abstract

In this paper we address issues with image retrieval benchmarking on standard and popular Oxford 5k and Paris 6k datasets. In particular, annotation errors, the size of the dataset, and the level of challenge are addressed: new annotation for both datasets is created with an extra attention to the reliability of the ground truth. Three new protocols of varying difficulty are introduced. The protocols allow fair comparison between different methods, including those using a dataset pre-processing stage. For each dataset, 15 new challenging queries are introduced. Finally, a new set of 1M hard, semi-automatically cleaned distractors is selected.

An extensive[1] comparison of the state-of-the-art methods is performed on the new benchmark. Different types of methods are evaluated, ranging from local-feature-based to modern CNN based methods. The best results are achieved by taking the best of the two worlds. Most importantly, image retrieval appears far from being solved.

1. Introduction

Image retrieval methods have gone through significant development in the last decade, starting with descriptors based on local-features, first organized in bag-of-words [41], and further expanded by spatial verification [33], hamming embedding [16], and query expansion [7]. Compact representations reducing the memory footprint and speeding up queries started with aggregating local descriptors [18]. Nowadays, the most efficient retrieval methods are based on fine-tuned convolutional neural networks (CNNs) [10, 37, 30].

In order to measure the progress and compare different methods, standardized image retrieval benchmarks are used. Besides the fact that a benchmark should simulate a real-world application, there are a number of properties that determine the quality of a benchmark: the *reliability of the annotation*, the *size*, and the *challenge level*.

The authors were supported by the MSMT LL1303 ERC-CZ grant.
[1]We thank Facebook for the donation of GPU servers, which made the evaluation tractable.

Errors in the annotation may systematically corrupt the comparison of different methods. Too small datasets are prone to over-fitting and do not allow the evaluation of the efficiency of the methods. The reliability of the annotation and size of the dataset are competing factors, as it is difficult to secure accurate human annotation of large datasets. The size is commonly increased by adding a distractor set, which contains irrelevant images that are selected in an automated manner (different tags, GPS information, *etc*.) Finally, benchmarks where all the methods achieve almost perfect results [23] cannot be used for further improvement or qualitative comparison.

Many datasets have been introduced to measure the performance of image retrieval. Oxford [33] and Paris [34] datasets belong to the most popular ones. Numerous methods of image retrieval [7, 31, 5, 27, 47, 3, 48, 20, 37, 10] and visual localization [9, 1] have used these datasets for evaluation. One reason for their popularity is that, in contrast to datasets that contain small groups of 4-5 similar images like Holidays [16] and UKBench [29], Oxford and Paris contain queries with up to hundreds of positive images.

Despite the popularity, there are known issues with the two datasets, which are related to all three important properties of evaluation benchmarks. First, there are errors in the annotation, including both false positives and false negatives. Further inaccuracy is introduced by queries of different sides of a landmark, sharing the annotation despite being visually distinguishable. Second, the annotated datasets are relatively small (5,062 and 6,392 images respectively). Third, current methods report near-perfect results on both the datasets. It has become difficult to draw conclusions from quantitative evaluations, especially given the annotation errors [14].

The lack of difficulty is not caused by the fact that non-trivial instances are not present in the dataset, but due to the annotation. The annotation was introduced about ten years ago. At that time, the annotators had different perception of what the limits of image retrieval are. Many instances that are nowadays considered as a change of viewpoint expected to be retrieved, are *de facto* excluded from the evaluation by being labelled as *Junk*.

The size issue of the datasets is partially addressed by the Oxford 100k *distractor set*. However, this contains false negative images, as well as images that are not challenging. State-of-the-art methods maintain near-perfect results even in the presence of these distractors. As a result, additional computational effort is spent with little benefit in drawing conclusions.

Contributions. As a first contribution, we generate new annotation for Oxford and Paris datasets, update the evaluation protocol, define new, more difficult queries, and create new set of challenging distractors. As an outcome we produce *Revisited Oxford*, *Revisited Paris*, and an accompanying distractor set of one million images. We refer to them as $\mathcal{R}$Oxford, $\mathcal{R}$Paris, and $\mathcal{R}$1M respectively.

As a second contribution, we provide extensive evaluation of image retrieval methods, ranging from local-feature based to CNN-descriptor based approaches, including various methods of re-ranking.

2. Revisiting the datasets

In this section we describe in detail why and how we revisit the annotation of Oxford and Paris datasets, present a new evaluation protocol and an accompanying challenging set of one million distractor images. The revisited benchmark is publicly available[2].

2.1. The original datasets

The original Oxford and Paris datasets consist of 5,063 and 6,392 high-resolution (1024×768) images, respectively. Each dataset contains 55 queries comprising 5 queries per landmark, coming from a total of 11 landmarks. Given a landmark query image, the goal is to retrieve all database images depicting the same landmark. The original annotation (labeling) is performed manually and consists of 11 ground truth lists since 5 images of the same landmark form a *query group*. Three labels are used, namely, *positive*, *junk*, and *negative* [3].

Positive images clearly depict more than 25% of the landmark, junk less than 25%, while the landmark is not shown in negative ones. The performance is measured via mean average precision (mAP) [33] over all 55 queries, while junk images are ignored, *i.e.* the evaluation is performed as if they were not present in the database.

2.2. Revisiting the annotation

The annotation is performed by five annotators, and it is performed in the following steps.

Query groups. Query groups share the same ground-truth list and simplify the labeling problem, but also cause some inaccuracies in the original annotation. *Balliol* and *Christ Church* landmarks are depicted from a different (not fully symmetric) side in the 2^{nd} and 4^{th} query, respectively. *Arc de Triomphe* has three day and two night queries, while day-night matching is considered a challenging problem [49, 35]. We alleviate this by splitting these cases into separate groups. As a result, we form 13 and 12 query groups on Oxford and Paris, respectively.

Additional queries. We introduce new and more challenging queries (see Figure 1) compared to the original ones. There are 15 new queries per dataset, originating from five out of the original 11 landmarks, with three queries per landmark. Along with the 55 original queries, they comprise the new set of 70 queries per dataset. The query groups, defined by visual similarity, are 26 and 25 for $\mathcal{R}$Oxford and $\mathcal{R}$Paris, respectively. As in the original datasets, the query object bounding boxes are simulating not only a user attempting to remove background clutter, but also cases of large occlusion.

Labeling step 1: Selection of potential positives. Each annotator manually inspects the whole dataset and marks images depicting any side or version of a landmark. The goal is to collect all images that are originally incorrectly labeled as negative. Even uncertain cases are included in this step and the process is repeated for each landmark. Apart from inspecting the whole dataset, an interactive retrieval tool is used to actively search for further possible positive images. All images marked in this phase are merged together with images originally annotated as positive or junk, creating a list of *potential positives* for each landmark.

Labeling step 2: Label assignment. In this step, each annotator manually inspects the list of potential positives for each query group and assigns labels. The possible labels are *Easy*, *Hard*, *Unclear*, and *Negative*. All images not in the list of potential positives are automatically marked negative. The instructions given to the annotators for each of the labels are as follows.

- *Easy:* The image clearly depicts the query landmark from the same side, with no large viewpoint change, no significant occlusion, no extreme illumination change, and no severe background clutter. In the case of fully symmetric sides, any side is valid.

$\mathcal{R}$Oxford				
Labels	Easy	Hard	Uncl.	Neg.
Positive	438	50	93	1
Junk	50	222	72	9
Negative	1	72	133	63768

$\mathcal{R}$Paris				
Labels	Easy	Hard	Uncl.	Neg.
Positive	1222	643	136	6
Junk	91	813	835	61
Negative	16	147	273	71621

Table 1. Number of images switching their labeling from the original annotation (positive, junk, negative) to the new one (easy, hard, unclear, negative).

[2] cmp.felk.cvut.cz/revisitop

[3] We rename the originally used labels {good, ok, junk, and absent} for the purpose of consistency with our terminology. Good and ok were always used as positives.

Figure 1. The newly added queries for $\mathcal{R}$Oxford(top) and $\mathcal{R}$Paris(bottom) datasets. Merged with the original queries, they comprise a new set of 70 queries in total.

Figure 2. Examples of *extreme* labeling mistakes in the original labeling. We show the **query (blue)** image and the associated database images that were originally marked as **negative (red)** or **positive (green)**. Best viewed in color.

- *Hard:* The image depicts the query landmark, but with viewing conditions that are difficult to match with the query. The depicted (side of the) landmark is recognizable without any contextual visual information.
- *Unclear:* (a) The image possibly depicts the landmark in question, but the content is not enough to make a certain guess about the overlap with the query region, or context is needed to clarify. (b) The image depicts a different side of a partially symmetric building, where the symmetry is significant and discriminative enough.
- *Negative:* The image is not satisfying any of the previous conditions. For instance, it depicts a different side of the landmark compared to that of the query, with no discriminative symmetries. If the image has any physical overlap with the query, it is never negative, but rather unclear, easy, or hard according to the above.

Labeling step 3: Refinement. For each query group, each image in the list of potential positives has been assigned a five-tuple of labels, one per annotator. We perform majority voting in two steps to define the final label. The first step is voting for {easy,hard}, {unclear}, or {negative}, grouping easy and hard together. In case majority goes to {easy,hard}, the second step is to decide which of the two. Draws of the first step are assigned to unclear, and of the second step to hard. Illustrative examples are (EEHUU) $\rightarrow$ E, (EHUUN) $\rightarrow$ U, and (HHUNN) $\rightarrow$ U. Finally, for each query group, we inspect images by descending label entropy to make sure there are no errors.

Revisited datasets: $\mathcal{R}$Oxford and $\mathcal{R}$Paris. Images from which the queries are cropped are excluded from the evaluation dataset. This way, unfair comparisons are avoided in the case of methods performing off-line preprocessing of the database [2, 14]; any preprocessing should not include any part of query images. The revisited datasets, namely, $\mathcal{R}$Oxford and $\mathcal{R}$Paris, comprise 4,993 and 6,322 images respectively, after removing the 70 queries.

In Table 1, we show statistics of label transitions from the old to the new annotations. Note that errors in the original annotation that affect the evaluation, *e.g.* negative moving to easy or hard, are not uncommon. The transitions from junk to easy or hard are reflecting the greater challenges of the new annotation. Representative examples of *extreme* labeling errors of the original annotation are shown in Figure 2. In Figure 3, representative examples of easy, hard, and unclear images are presented for several queries. This will help understanding the level of challenge of each evaluation protocol listed below.

2.3. Evaluation protocol

Only the cropped regions are to be used as queries; never the full image, since the ground-truth labeling strictly considers only the visual content inside the query region.

The standard practice of reporting mean average precision (mAP) [33] for performance evaluation is followed. Additionally, mean precision at rank K (mP@K) is reported. The former reflects the overall quality of the ranked list. The latter reflects the quality of the results of a search engine as they would be visually inspected by a user. More importantly, it is correlated to performance of subsequent processing steps [7, 21]. During the evaluation, positive images should be retrieved, while there is also an ignore list per query. Three evaluation setups of different difficulty are defined by treating labels (easy, hard, unclear) as positive or negative, or ignoring them:

- **Easy (E):** *Easy* images are treated as positive, while *Hard* and *Unclear* are ignored (same as *Junk* in [33]).
- **Medium (M):** *Easy* and *Hard* images are treated as positive, while *Unclear* are ignored.
- **Hard (H):** *Hard* images are treated as positive, while *Easy* and *Unclear* are ignored.

If there are no positive images for a query in a particular setting, then that query is excluded from the evaluation.

Figure 3. Sample **query (blue)** images and images that are respectively marked as **easy (dark green)**, **hard (light green)**, and **unclear (yellow)**. Best viewed in color.

Figure 4. Sample false negative images in Oxford100k.

The original annotation and evaluation protocol is closest to our **Easy** setup. Even though this setup is now trivial for the best performing methods, it can still be used for evaluation of *e.g.* near duplicate detection or retrieval with ultra short codes. The other setups, **Medium** and **Hard**, are challenging and even the best performing methods achieve relatively low scores. See Section 4 for details.

2.4. Distractor set $\mathcal{R}$1M

Large scale experiments on Oxford and Paris dataset are commonly performed with the accompanying distractor set of 100k images, namely Oxford100k [33]. Recent results [14, 13] show that the performance only slightly degrades by adding Oxford100k in the database compared to a small-scale setting. Moreover, it is not manually cleaned and, as a consequence, Oxford and Paris landmarks are depicted in some of the distractor images (see Figure 4), hence adding further noise to the evaluation procedure.

Larger distactor sets are used in the literature [33, 34, 16, 44] but none of them are standardized to provide a testbed for direct large scale comparison nor are they manually cleaned [16]. Some of the distractor sets are also biased, since they contain images of different resolution than the Oxford and Paris datasets.

We construct a new distractor set with exactly 1,001,001 high-resolution (1024×768) images, which we refer to as $\mathcal{R}$1M dataset. It is cleaned by a semi-automatic process. We automatically pick hard images for a number of state-of-the-art methods, resulting in a challenging large scale setup.

YFCC100M and semi-automatic cleaning. We randomly choose 5M images with GPS information from YFCC100M dataset [43]. Then, we exclude UK, France, and Las Vegas; the latter due to the *Eiffel Tower* and *Arc de Triomphe* replicas. We end up with roughly 4.1M images that are available for downloading in high resolution. We rank images with the same search tool as used in labeling step 1. Then, we manually inspect the top 2k images per landmark, and remove those depicting the query landmarks (faulty GPS, toy models, and paintings/photographs of landmarks). In total, we find 110 such images.

Un-biased mining of distracting images. We propose a way to keep the most challenging 1M out of the 4.1M images. We perform all 70 queries into the 4.1M database with a number of methods. For each query and for each distractor image we count the fraction of easy or hard images that are ranked after it. We sum these fractions over all queries of $\mathcal{R}$Oxford and $\mathcal{R}$Paris and over different methods, resulting in a measurement of how *distracting* each distractor image is. We choose the set of 1M most distracting images and refer to it as the $\mathcal{R}$1M *distractor set*.

Three complementary retrieval methods are chosen to compute this measurement. These are fine-tuned ResNet with GeM pooling [37], pre-trained (on ImageNet) AlexNet with MAC pooling [38], and ASMK [46]. More details on these methods are given in Section 3. Finally, we perform a sanity check to show that this selection process is not significantly biased to distract only those 3 methods. This includes two additional methods, VLAD [18] and fine-tuned ResNet with R-MAC pooling by Gordo *et al.* [10]. As shown in Table 2, the performance on the hardest 1M distractors is hardly affected whether one of those additional methods participates or not in the selection process. This suggests that the mining process is not biased towards particular methods.

Table 2 also shows that the distractor set we choose (version 1M (1,2,3) in the Table) is much harder than a random 1M subset and nearly as hard as all 4M distractor images. Example images from the set $\mathcal{R}$1M are shown in Figure 5.

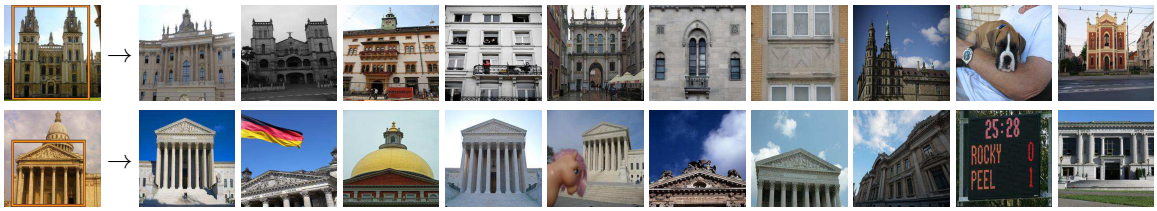


Figure 5. The most distracting images per query for two queries.

3. Extensive evaluation

We evaluate a number of state-of-the-art approaches on the new benchmark and offer a rich testbed for future comparisons. We list them in this section and they belong to two main categories, namely, classical retrieval approaches using local features and CNN-based methods producing global image descriptors.

3.1. Local-feature-based methods

Methods based on local invariant features [25, 26] and the Bag-of-Words (BoW) model [41, 33, 7, 34, 6, 27, 47, 4, 52, 54, 42] were dominating the field of image retrieval until the advent of CNN-based approaches [38, 3, 48, 20, 1, 10, 37, 28, 51]. A typical pipeline consists of invariant local feature detection [26], local descriptor extraction [25], quantization with a visual codebook [41], typically created with k-means, assignment of descriptors to visual words and finally descriptor aggregation in a single embedding [19, 32] or individual feature indexing with an inverted file structure [45, 33, 31]. We consider state-of-the-art methods from both categories. In particular, we use up-right hessian-affine (HesAff) features [31], RootSIFT (rSIFT) descriptors [2], and create the codebooks on the landmark dataset from [37], same as the one used for the whitening of CNN-based methods. Note that we always crop the queries according to the defined region and then perform any processing to be directly comparable to CNN-based methods.

Distractor set	$\mathcal{R}$Oxford					$\mathcal{R}$Paris				
	Method									
	(1)	(2)	(3)	(4)	(5)	(1)	(2)	(3)	(4)	(5)
4M	33.3	11.1	33.2	33.7	15.6	40.7	11.4	30.0	45.4	17.9
1M (1,2,3)	33.9	11.1	34.8	33.9	17.4	44.1	11.8	31.7	48.1	19.6
1M (1,2,3,4)	33.7	11.1	34.8	33.8	17.5	43.8	11.8	31.8	47.7	19.7
1M (1,2,3,5)	33.7	11.1	34.6	33.9	17.2	43.5	11.7	31.4	47.7	19.2
1M (random)	37.6	13.7	37.4	38.9	20.4	47.3	16.2	34.2	53.1	21.9

Table 2. Performance (mAP) evaluation with the Medium protocol for different distractor sets. The methods considered are (1) Fine-tuned ResNet101 with GeM pooling [37]; (2) Off-the-shelf AlexNet with MAC pooling [38]; (3) HesAff–rSIFT–ASMK* [46]; (4) Fine-tuned ResNet101 with R-MAC pooling [10]; (5) HesAff–rSIFT–VLAD [18]. The sanity check includes evaluation for different distractor sets, *i.e.* all, hardest subset chosen by method (1,2,3), (1,2,3,4), (1,2,4,5), and a random 1M sample.

We additionally follow the same BoW-based pipeline while replacing hessian-affine and RootSIFT with the deep local attentive features (DELF) [30]. The default extraction approach is followed (*i.e.* at most 1000 features per image), but we reduce the descriptor dimensionality to 128 and not to 40 to be comparable to RootSIFT. This variant is a bridge between classical approaches and deep learning.

VLAD. The Vector of Locally Aggregated Descriptors [18] (VLAD) is created by first-order statistics of the local descriptors. The residual vectors between descriptors and the closest centroid are aggregated w.r.t. a codebook whose size is 256 in our experiments. We reduce its dimensionality down to 2048 with PCA, while square-root normalization is also used [15].

SMK*. The binarized version of the Selective Match Kernel [46] (SMK*), a simple extension of the Hamming Embedding [16] (HE) technique, uses an inverted file structure to separately indexes binarized residual vectors while it performs the matching with a selective monomial kernel function. The codebook size is 65,536 in our experiments, while burstiness normalization [17] is always used. Multiple assignment to three nearest words is used on the query side, while the hamming distance threshold is set to 52 out of 128 bits. The rest are the default parameters.

ASMK*. The binarized version of the Aggregated Selective Match Kernel [46] (ASMK*) is an extension of SMK* that jointly encodes local descriptors that are assigned to the same visual word and handles the burstiness phenomenon. Same parametrization as SMK* is used.

SP. Spatial verification (SP) is known to be crucial for particular object retrieval [33] and is performed with the RANSAC algorithm [8]. It is applied on the 100 top-ranked images, as these are formed by a first filtering step, *e.g.* the SMK* or ASMK* method. Its result is the number of inlier correspondences, which is one of the most intuitive similarity measures and allows to detect true positive images. To assume that an image is spatially verified, we require 5 inliers with ASMK* and 10 with other methods.

HQE. Query expansion (QE), firstly introduced by Chum *et al.* [7] in the visual domain, typically uses spatial verification to select true positive among the top retrieved result and issues an enhanced query including the verified images. Hamming Query Expansion [47] (HQE) is combining QE with HE. We use same soft assignment as SMK* and the default parameters.

3.2. CNN-based global descriptor methods

We list different aspects of a CNN-based method for image retrieval, which we later combine to form different baselines that exist in the literature.

CNN architectures. We include 3 highly influential CNN architectures, namely AlexNet [22], VGG-16 [40], and ResNet101 [12]. They have different number of layers, complexity, and also produce descriptors of different dimensionality (256, 512, and 2048, respectively).

Pooling. A common practice is to consider a convolutional feature map and perform a pooling mechanism to construct a global image descriptor. We consider max-pooling (MAC) [38, 48], sum-pooling (SPoC) [3], weighted sum-pooling (CroW) [20], regional max-pooling (R-MAC) [48], generalized mean-pooling (GeM) [37], and NetVLAD pooling [1]. The pooling is always applied on the last convolutional feature map.

Multi-scale. The input image is resized to a maximum 1024×1024 size. Then, three re-scaled versions with scaling factor of 1, $1/\sqrt{2}$, and $1/2$ are fed to the network. Finally, the resulting descriptors are combined into a single descriptor by average pooling [10] for all methods, except for GeM where generalized-mean pooling is used [37]. This is shown to improve the performance of the CNN-based descriptors [10, 37].

Off-the-shelf *vs*. retrieval fine-tuning. Networks that are pre-trained on ImageNet [39] (off-the-shelf) are directly applicable on image retrieval. Moreover, we consider the following cases of fine-tuning for the task. Radenovic *et al.* [36] fine-tune a network with landmarks photos using contrastive loss [11]. This is available with MAC [36] and GeM pooling [37]. Similarly, Gordo *et al.* [10] fine-tune R-MAC pooling with landmark photos and triplet loss [50]. Finally, NetVLAD [1] is fine-tuned using street-view images and GPS information.

Descriptor whitening is known to be essential for such descriptors. We use the same landmark dataset [37] to learn the whitening for all methods. We use PCA whitening [15, 3] for all the off-the-shelf networks, and supervised whitening with SfM labels [24, 36] for all the fine-tuned ones. One exception is the tuning that includes the whitening in the network [10].

Query Expansion is directly applicable on top of global CNN-based descriptors. More specifically, we use α query expansion [37] (αQE) and diffusion [14] (DFS).

Method	Oxf	$\mathcal{R}$Oxford			Par	$\mathcal{R}$Paris		
		E	M	H		E	M	H
HesAff–rSIFT–SMK*	78.1	74.1	59.4	35.4	74.6	80.6	59.0	31.2
R–[O]–R-MAC	78.3	74.2	49.8	18.5	90.9	89.9	74.0	52.1
R–[37]–GeM	87.8	84.8	64.7	38.5	92.7	92.1	77.2	56.3
R–[37]–GeM+DFS	90.0	86.5	69.8	40.5	95.3	93.9	88.9	78.5

Table 3. Performance (mAP) on Oxford (Oxf) and Paris (Par) with the original annotation, and $\mathcal{R}$Oxford and $\mathcal{R}$Paris with the newly proposed annotation with three different protocol setups: Easy (E), Medium (M), Hard (H).

Method	Memory (GB)	Time (sec)		
		Extraction		Search
		GPU	CPU	
HesAff–rSIFT–ASMK*	62.0	n/a + 0.06	1.08 + 2.35	0.98
HesAff–rSIFT–ASMK*+SP				2.00
DELF–ASMK*+SP	10.3	0.41 + 0.01	n/a + 0.54	0.52
A–[37]–GeM	0.96	0.12	1.99	0.38
V–[37]–GeM	1.92	0.23	31.11	0.56
R–[37]–GeM	7.68	0.37	14.51	1.21

Table 4. Time and memory measurements. Extraction time on a single thread GPU (Tesla P100) / CPU (Intel Xeon CPU E5-2630 v2 @ 2.60GHz) per image of size 1024x768, the memory requirements and the search time (single thread CPU) reported for the database of $\mathcal{R}$Oxford+$\mathcal{R}$1M images. Feature extraction + visual word assignment is reported for ASMK*. SP: Geometry information is loaded from the disk and the loading time is included in search time. We did not consider geometry quantization [31].

4. Results

We report a performance comparison between the old and the revisited datasets. Additionally, we provide an extensive evaluation of the state-of-the-art methods on the revisited dataset, with and without the new large-scale distractor set, setting up a testbed for future comparisons.

The evaluation includes local feature-based approaches (see Section 3.1 for details and abbreviations), referred to by the combination of local feature type and representation method, *e.g.* HesAff–rSIFT–ASMK*. CNN-based global descriptors are denoted with the following abbreviations. Network architectures are AlexNet (A), VGG-16 (V), and ResNet101 (R). The fine-tuning options are triplet loss with GPS guided mining [1], triplet loss with spatially verified positive pairs [10], contrastive loss with mining from 3D models [36] and [37], and finally the off-the-shelf [O] networks. Pooling approaches are as listed in Section 3.2. For instance, ResNet101 with GeM pooling that is fine-tuned with contrastive loss and the training dataset by Radenovic *et al.* [37] is referred to as R–[37]–GeM.

Revisited *vs*. original. We compare the performance when evaluated on the original datasets, and the revisited annotation with the new protocols. The results for four representative methods are presented in Table 3. The old setup appears to be close to the new **Easy** setup, while **Medium** and **Hard** appear to be more challenging. We observe that the performance of the **Easy** setup is nearly saturated and, therefore, do not use it but only evaluate **Medium** and **Hard** setups in the subsequent experiments.

Method	Medium								Hard							
	$\mathcal{R}$Oxf		$\mathcal{R}$Oxf+$\mathcal{R}$1M		$\mathcal{R}$Par		$\mathcal{R}$Par+$\mathcal{R}$1M		$\mathcal{R}$Oxf		$\mathcal{R}$Oxf+$\mathcal{R}$1M		$\mathcal{R}$Par		$\mathcal{R}$Par+$\mathcal{R}$1M	
	mAP	mP@10	mAP	mP@10	mAP	mP@10	mAP	mP@10	mAP	mP@10	mAP	mP@10	mAP	mP@10	mAP	mP@10
HesAff–rSIFT–VLAD	33.9	54.9	17.4	34.8	43.6	90.9	19.6	76.1	13.2	18.1	5.6	7.0	17.5	50.7	3.3	21.1
HesAff–rSIFT–SMK*	59.4	83.6	35.8	64.6	59.0	97.4	34.1	89.1	35.4	53.7	16.4	27.7	31.2	72.6	10.5	47.6
HesAff–rSIFT–ASMK*	60.4	85.6	45.0	76.0	61.2	97.9	42.0	95.3	36.4	56.7	25.7	42.1	34.5	80.6	16.5	63.4
HesAff–rSIFT–SMK*+SP	59.8	84.3	38.1	67.1	59.2	97.4	34.5	89.3	35.8	54.0	17.7	30.3	31.3	73.6	11.0	49.1
HesAff–rSIFT–ASMK*+SP	60.6	86.1	46.8	79.6	61.4	97.9	42.3	95.3	36.7	57.0	26.9	45.3	35.0	81.7	16.8	65.3
DELF–ASMK*+SP	67.8	87.9	53.8	81.1	76.9	99.3	57.3	98.3	43.1	62.4	31.2	50.7	55.4	93.4	26.4	75.7
A – [O] – MAC	28.3	44.7	14.1	28.3	47.3	88.6	18.7	69.4	8.8	15.5	3.5	5.1	23.1	61.6	4.1	29.0
A – [O] – GeM	33.8	51.2	16.3	32.4	52.7	90.1	23.8	78.1	10.4	16.7	3.9	6.3	26.0	68.0	5.5	31.6
A – [36] – MAC	41.3	62.1	23.9	43.0	56.4	92.9	29.6	85.4	17.8	28.2	8.4	11.9	28.7	69.3	8.5	40.9
A – [37] – GeM	43.3	62.1	24.2	42.8	58.0	91.6	29.9	84.6	17.1	26.2	9.4	11.9	29.7	67.6	8.4	39.6
V – [O] – MAC	37.8	57.8	21.8	39.7	59.2	93.3	33.6	87.1	14.6	27.0	7.4	11.9	35.9	78.4	13.2	54.7
V – [O] – SPoC	38.0	54.6	17.1	33.3	59.8	93.0	30.3	83.0	11.4	20.9	0.9	2.9	32.4	69.7	7.6	30.6
V – [O] – CroW	41.4	58.8	22.5	40.5	62.9	94.4	34.1	87.1	13.9	25.7	3.0	6.6	36.9	77.9	10.3	45.1
V – [O] – GeM	40.5	60.3	25.4	45.6	63.2	94.6	37.5	88.6	15.7	28.6	7.6	12.1	38.8	79.0	14.2	55.9
V – [O] – R-MAC	42.5	62.8	21.7	40.3	66.2	95.4	39.9	88.9	12.0	26.1	1.7	5.8	40.9	77.1	14.8	54.0
V – [1] – NetVLAD	37.1	56.5	20.7	37.1	59.8	94.0	31.8	85.7	13.8	23.3	6.0	8.4	35.0	73.7	11.5	46.6
V – [36] – MAC	58.4	81.1	39.7	68.6	66.8	97.7	42.4	92.6	30.5	48.0	17.9	27.9	42.0	82.9	17.7	63.7
V – [37] – GeM	61.9	82.7	42.6	68.1	69.3	97.9	45.4	94.1	33.7	51.0	19.0	29.4	44.3	83.7	19.1	64.9
R – [O] – MAC	41.7	65.0	24.2	43.7	66.2	96.4	40.8	93.0	18.0	32.9	5.7	14.4	44.1	86.3	18.2	67.7
R – [O] – SPoC	39.8	61.0	21.5	40.4	69.2	96.7	41.6	92.0	12.4	23.8	2.8	5.6	44.7	78.0	15.3	54.4
R – [O] – CroW	42.4	61.9	21.2	39.4	70.4	97.1	42.7	92.9	13.3	27.7	3.3	9.3	47.2	83.6	16.3	61.6
R – [O] – GeM	45.0	66.2	25.6	45.1	70.7	97.0	46.2	94.0	17.7	32.6	4.7	13.4	48.7	88.0	20.3	70.4
R – [O] – R-MAC	49.8	68.9	29.2	48.9	74.0	97.7	49.3	93.7	18.5	32.2	4.5	13.0	52.1	87.1	21.3	67.4
R – [37] – GeM	64.7	84.7	45.2	71.7	77.2	98.1	52.3	95.3	38.5	53.0	19.9	34.9	56.3	89.1	24.7	73.3
R – [10] – R-MAC	60.9	78.1	39.3	62.1	78.9	96.9	54.8	93.9	32.4	50.0	12.5	24.9	59.4	86.1	28.0	70.0
Query expansion (QE) and diffusion (DFS)																
HesAff–rSIFT–HQE	66.3	85.6	42.7	67.4	68.9	97.3	44.2	90.1	41.3	60.0	23.2	37.6	44.7	79.9	20.3	51.4
HesAff–rSIFT–HQE+SP	71.3	88.1	52.0	76.7	70.2	98.6	46.8	93.0	49.7	69.6	29.8	50.1	45.1	83.9	21.8	61.9
DELF–HQE+SP	73.4	88.2	60.6	79.7	84.0	98.3	65.2	96.1	50.3	67.2	37.9	56.1	69.3	93.7	35.8	69.1
R – [O] – R-MAC+αQE	51.9	70.3	30.8	49.7	77.3	97.9	55.3	94.7	21.8	35.2	5.2	15.9	57.0	87.6	28.0	76.1
V – [37] – GeM+αQE	66.6	85.7	47.0	72.0	74.0	98.4	52.9	95.9	38.9	57.3	21.1	34.6	51.0	88.4	25.6	75.0
R – [37] – GeM+αQE	67.2	86.0	49.0	74.7	80.7	98.9	58.0	95.9	40.8	54.9	24.2	40.3	61.8	90.6	31.0	80.4
R – [10] – R-MAC+αQE	64.8	78.5	45.7	66.5	82.7	97.3	61.0	94.3	36.8	53.3	19.5	36.6	65.7	90.1	35.0	76.9
V – [37] – GeM+DFS	69.6	84.7	60.4	79.4	85.6	97.1	80.7	97.1	41.1	51.1	33.1	49.6	73.9	93.7	65.3	93.1
R – [37] – GeM+DFS	69.8	84.0	61.5	77.1	88.9	96.9	84.9	95.9	40.5	54.4	33.1	48.2	78.5	94.6	71.6	93.7
R – [10] – R-MAC+DFS	69.0	82.3	56.6	68.6	89.5	96.7	83.2	93.3	44.7	60.5	28.4	43.6	80.0	94.1	70.4	89.1
HesAff–rSIFT–ASMK*+SP → R– [37]–GeM+DFS	79.1	92.6	74.3	87.9	91.0	98.3	85.9	97.1	52.7	66.1	48.7	65.9	81.0	97.9	73.2	96.6
HesAff–rSIFT–ASMK*+SP → R– [10]–R-MAC+DFS	80.2	93.7	74.9	87.9	92.5	98.7	87.5	97.1	54.8	70.6	47.5	62.4	84.0	98.3	76.0	96.3
DELF–ASMK*+SP → R– [10]–R-MAC+DFS	75.0	87.9	68.7	83.6	90.5	98.0	86.6	98.1	48.3	64.0	39.4	55.7	81.2	95.6	74.2	94.6

Table 5. Performance evaluation (mAP, mP@10) on $\mathcal{R}$Oxford ($\mathcal{R}$Oxf) and $\mathcal{R}$Paris ($\mathcal{R}$Par) without and with $\mathcal{R}$1M distractors. We report results with the revisited annotation, using Medium and Hard evaluation protocols. We use a color-map that is normalized according to the minimum (white) and maximum (**green** / **orange**) value per column.

State of the art evaluation. We perform an extensive evaluation of the state-of-the-art methods for image retrieval. We present time/memory measurements in Table 4 and performance results in Table 5. We additionally show the average precision (AP) per query for a set of representative methods in Figures 6 and 7, for $\mathcal{R}$Oxford and $\mathcal{R}$Paris, respectively. The representative set covers the progress of methods over time in the task of image retrieval. In the evaluation, we observe that there is no single method achieving the highest score on every protocol per dataset. Local-feature-based methods perform very well on $\mathcal{R}$Oxford, especially at large scale, achieving state-of-the-art performance, while CNN-based methods seem to dominate on $\mathcal{R}$Paris. We observe that BoW-based classical approaches are still not obsolete, but their improvement typically comes at significant additional cost. Recent CNN-based local features, *i.e.* DELF, reduce the number of features and improve the performance at the same time.

CNN fine-tuning consistently brings improvements over the off-the-shelf networks. The new protocols make it clear that improvements are needed at larger scale and the hard setup. Many images are not retrieved, while the top 10 results mostly contain false positives. Interestingly, we observe that query expansion approaches (*e.g.* diffusion) degrade the performance of queries with few relevant images (see Figures 6 and 7). This phenomenon is more pronounced in the revisited datasets, where the the query images are removed from the preprocessing. We did not include separate regional representation and indexing [38], which is previously shown to be beneficial. Preliminary experiments with ResNet and GeM pooling show that it does not deliver improvements that are significant enough to justify the additional memory and complexity cost.

The best of both worlds. The new dataset and protocols reveal space for improvement by CNN-based global descriptors in cases where local features are still better. Diffu-

Figure 6. Performance (AP) per query on $\mathcal{R}$Oxford + $\mathcal{R}$1M with Medium setup. AP is shown with a bar for 8 methods. The methods, from left to right, are HesAff–rSIFT–ASMK*+SP, DELF–ASMK*+SP, DELF–HQE+SP, V–[O]–R-MAC, R–[O]–GeM, R–[37]–GeM, R–[37]–GeM+DFS, HesAff–rSIFT–ASMK*+SP → R–[37]–GeM+DFS. The total number of easy and hard images is printed on each histogram. Best viewed in color.

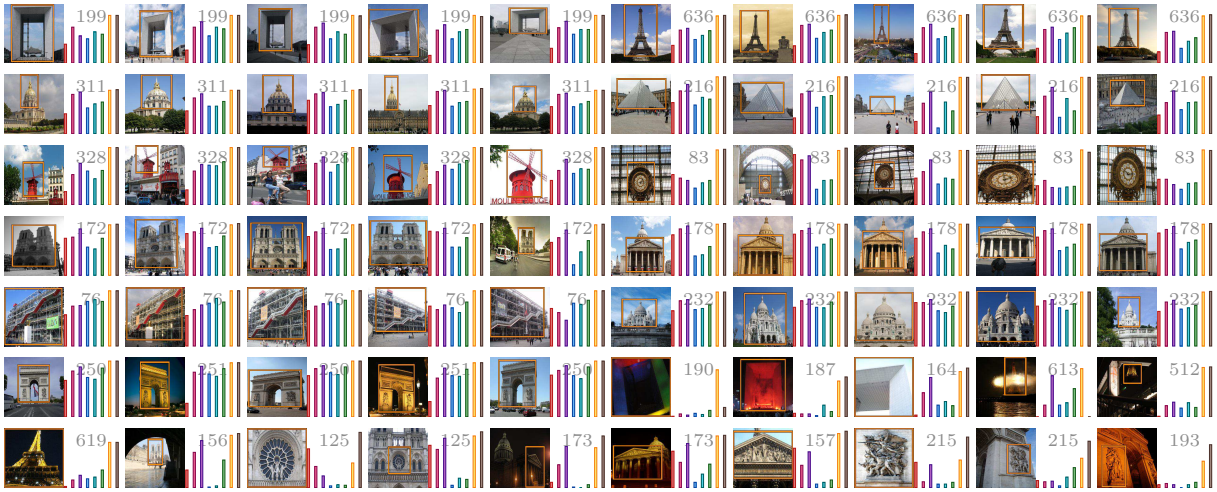


Figure 7. Performance (AP) per query on $\mathcal{R}$Paris + $\mathcal{R}$1M with Medium setup. AP is shown with a bar for 8 methods. The methods, from left to right, are HesAff–rSIFT–ASMK*+SP, DELF–ASMK*+SP, DELF–HQE+SP, V–[O]–R-MAC, R–[O]–GeM, R–[37]–GeM, R–[37]–GeM+DFS, HesAff–rSIFT–ASMK*+SP → R–[37]–GeM+DFS. The total number of easy and hard images is printed on each histogram. Best viewed in color.

sion performs similarity propagation by starting from the query's nearest neighbors according to the CNN global descriptor. This inevitably includes false positives, especially in the case of few relevant images. On the other hand, local features, *e.g.* with ASMK*+SP, offer a verified list of relevant images. Starting the diffusion process from geometrically verified images obtained by BoW methods combines the benefits of the two worlds. This combined approach, shown at the bottom part of Table 5, improves the performance and supports the message that both worlds have their own benefits. Of course this experiment is expensive and we perform it to merely show a possible direction to improve CNN global descriptors. There are more methods that combine CNNs and local features [53], but we focus on the results related to methods included in our evaluation.

5. Conclusions

We have revisited two of the most established image retrieval datasets, that were perceived as performance saturated. To make it suitable for modern image retrieval benchmarking, we address drawbacks of the original annotation. This includes new annotation for both datasets that was created with an extra attention to the reliability of the ground truth, and an introduction of 1M hard distractor set.

An extensive evaluation provides a testbed for future comparisons and concludes that image retrieval is still an open problem, especially at large scale and under difficult viewing conditions.

References

[1] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. In *CVPR*, 2016. 1, 5, 6, 7

[2] R. Arandjelovic and A. Zisserman. Three things everyone should know to improve object retrieval. In *CVPR*, 2012. 3, 5

[3] A. Babenko and V. Lempitsky. Aggregating deep convolutional features for image retrieval. In *ICCV*, 2015. 1, 5, 6

[4] Y. Cao, C. Wang, Z. Li, L. Zhang, and L. Zhang. Spatial-bag-of-features. In *CVPR*, 2010. 5

[5] O. Chum, A. Mikulik, M. Perdoch, and J. Matas. Total recall II: Query expansion revisited. In *CVPR*, 2011. 1

[6] O. Chum, M. Perdoch, and J. Matas. Geometric min-hashing: Finding a (thick) needle in a haystack. In *CVPR*, 2009. 5

[7] O. Chum, J. Philbin, J. Sivic, M. Isard, and A. Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *ICCV*, 2007. 1, 3, 5

[8] M. A. Fischler and R. C. Bolles. Random sample consensus. *Communications of ACM*, 1981. 5

[9] S. Gammeter, L. Bossard, T. Quack, and L. Van Gool. I know what you did last summer: object-level auto-annotation of holiday snaps. In *CVPR*, 2009. 1

[10] A. Gordo, J. Almazan, J. Revaud, and D. Larlus. End-to-end learning of deep visual representations for image retrieval. *IJCV*, 2017. 1, 4, 5, 6, 7

[11] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *CVPR*, 2006. 6

[12] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 6

[13] A. Iscen, Y. Avrithis, G. Tolias, T. Furon, and O. Chum. Fast spectral ranking for similarity search. In *CVPR*, 2018. 4

[14] A. Iscen, G. Tolias, Y. Avrithis, T. Furon, and O. Chum. Efficient diffusion on region manifolds: Recovering small objects with compact cnn representations. In *CVPR*, 2017. 1, 3, 4, 6

[15] H. Jégou and O. Chum. Negative evidences and cooccurences in image retrieval: The benefit of PCA and whitening. In *ECCV*, 2012. 5, 6

[16] H. Jégou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *ECCV*, 2008. 1, 4, 5

[17] H. Jégou, M. Douze, and C. Schmid. On the burstiness of visual elements. In *CVPR*, 2009. 5

[18] H. Jégou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *CVPR*, 2010. 1, 4, 5

[19] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid. Aggregating local descriptors into compact codes. *PAMI*, 2012. 5

[20] Y. Kalantidis, C. Mellina, and S. Osindero. Cross-dimensional weighting for aggregated deep convolutional features. In *ECCVW*, 2016. 1, 5, 6

[21] Y. Kalantidis, G. Tolias, Y. Avrithis, M. Phinikettos, E. Spyrou, P. Mylonas, and S. Kollias. Viral: Visual image retrieval and localization. *Multimedia Tools and Applications*, 2011. 3

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 6

[23] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998. 1

[24] K. Mikolajczyk and J. Matas. Improving descriptors for fast tree matching by optimal linear projection. In *ICCV*, 2007. 6

[25] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *PAMI*, 2005. 5

[26] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, T. Schaffalitzky, F.and Kadir, and L. Van Gool. A comparison of affine region detectors. *IJCV*, 2005. 5

[27] A. Mikulik, M. Perdoch, O. Chum, and J. Matas. Learning vocabularies over a fine quantization. *IJCV*, 2013. 1, 5

[28] E. Mohedano, K. McGuinness, N. E. O'Connor, A. Salvador, F. Marques, and X. Giro-i Nieto. Bags of local convolutional features for scalable instance search. In *ICMR*, 2016. 5

[29] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *CVPR*, 2006. 1

[30] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han. Large-scale image retrieval with attentive deep local features. In *ICCV*, 2017. 1, 5

[31] M. Perdoch, O. Chum, and J. Matas. Efficient representation of local geometry for large scale object retrieval. In *CVPR*, 2009. 1, 5, 6

[32] F. Perronnin, Y. Liu, J. Sanchez, and H. Poirier. Large-scale image retrieval with compressed Fisher vectors. In *CVPR*, 2010. 5

[33] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, 2007. 1, 2, 3, 4, 5

[34] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *CVPR*, 2008. 1, 4, 5

[35] F. Radenović, J. L. Schönberger, D. Ji, J.-M. Frahm, O. Chum, and J. Matas. From dusk till dawn: Modeling in the dark. In *CVPR*, 2016. 2

[36] F. Radenović, G. Tolias, and O. Chum. CNN image retrieval learns from BoW: Unsupervised fine-tuning with hard examples. In *ECCV*, 2016. 6, 7

[37] F. Radenović, G. Tolias, and O. Chum. Fine-tuning CNN image retrieval with no human annotation. In *arXiv*, 2017. 1, 4, 5, 6, 7, 8

[38] A. S. Razavian, J. Sullivan, S. Carlsson, and A. Maki. Visual instance retrieval with deep convolutional networks. *ITE Trans. MTA*, 2016. 4, 5, 6, 7

[39] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 2015. 6

[40] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *arXiv*, 2014. 6

[41] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *ICCV*, 2003. 1, 5

[42] R. Tao, E. Gavves, C. G. Snoek, and A. W. Smeulders. Locality in generic instance search from one example. In *CVPR*, 2014. 5

[43] B. Thomee, D. A. Shamma, G. Friedland, B. Elizalde, K. Ni, D. Poland, D. Borth, and L.-J. Li. YFCC100M: The new data in multimedia research. *Communications of the ACM*, 2016. 4

[44] G. Tolias and Y. Avrithis. Speeded-up, relaxed spatial matching. In *ICCV*, 2011. 4

[45] G. Tolias, Y. Avrithis, and H. Jégou. To aggregate or not to aggregate: Selective match kernels for image search. In *ICCV*, 2013. 5

[46] G. Tolias, Y. Avrithis, and H. Jégou. Image search with selective match kernels: aggregation across single and multiple images. *IJCV*, 2015. 4, 5

[47] G. Tolias and H. Jégou. Visual query expansion with or without geometry: refining local descriptors by feature aggregation. *Pattern Recognition*, 2014. 1, 5

[48] G. Tolias, R. Sicre, and H. Jégou. Particular object retrieval with integral max-pooling of CNN activations. In *ICLR*, 2016. 1, 5, 6

[49] Y. Verdie, K. Yi, P. Fua, and V. Lepetit. TILDE: a temporally invariant learned detector. In *CVPR*, 2015. 2

[50] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu. Learning fine-grained image similarity with deep ranking. In *CVPR*, 2014. 6

[51] J. Yue-Hei Ng, F. Yang, and L. S. Davis. Exploiting local features from deep networks for image retrieval. In *CVPR*, 2015. 5

[52] L. Zheng, S. Wang, Z. Liu, and Q. Tian. Lp-norm idf for large scale image search. In *CVPR*, 2013. 5

[53] L. Zheng, S. Wang, J. Wang, and Q. Tian. Accurate image search with multi-scale contextual evidences. *IJCV*, 2016. 8

[54] W. Zhou, Y. Lu, H. Li, Y. Song, and Q. Tian. Spatial coding for large scale partial-duplicate web image search. In *ACM Multimedia*, 2010. 5

XXVIII The Met Dataset: Instance-level Recognition for Artworks

Title: The Met Dataset: Instance-level Recognition for Artworks
Authors: NA. Ypsilantis, N. Garcia, G. Han, S. Ibrahimi, N. Van Noord, G. Tolias
Published at: NeurIPS 2021

The Met Dataset: Instance-level Recognition for Artworks

Nikolaos-Antonios Ypsilantis
VRG, Faculty of Electrical Engineering
Czech Technical University in Prague

Noa Garcia
Institute for Datability Science
Osaka University

Guangxing Han
DVMM Lab
Columbia University

Sarah Ibrahimi
Multimedia Analytics Lab
University of Amsterdam

Nanne van Noord
Multimedia Analytics Lab
University of Amsterdam

Giorgos Tolias
VRG, Faculty of Electrical Engineering
Czech Technical University in Prague

Abstract

This work introduces a dataset for large-scale instance-level recognition in the domain of artworks. The proposed benchmark exhibits a number of different challenges such as large inter-class similarity, long tail distribution, and many classes. We rely on the open access collection of The Met museum to form a large training set of about 224k classes, where each class corresponds to a museum exhibit with photos taken under studio conditions. Testing is primarily performed on photos taken by museum guests depicting exhibits, which introduces a distribution shift between training and testing. Testing is additionally performed on a set of images not related to Met exhibits making the task resemble an out-of-distribution detection problem. The proposed benchmark follows the paradigm of other recent datasets for instance-level recognition on different domains to encourage research on domain independent approaches. A number of suitable approaches are evaluated to offer a testbed for future comparisons. Self-supervised and supervised contrastive learning are effectively combined to train the backbone which is used for non-parametric classification that is shown as a promising direction. Dataset webpage: `http://cmp.felk.cvut.cz/met/`.

1 Introduction

Classification of objects can be done with categories defined at different levels of granularity. For instance, a particular piece of art is classified as the "Blue Poles" by Jackson Pollock, as painting, or artwork, from the point of view of instance-level recognition [8], fine-grained recognition [24], or generic category-level recognition [32], respectively. Instance-level recognition (ILR) is applied to a variety of domains such as products, landmarks, urban locations, and artworks. Representative examples of real world applications are place recognition [1, 22], landmark recognition and retrieval [39], image-based localization [33, 3], street-to-shop product matching [2, 17, 26], and artwork recognition [11]. There are several factors that make ILR a challenging task. It is typically required to deal with a large category set, whose size reaches the order of 10^6, with many classes represented by only a few or a single example, while the small between class variability further increases the hardness. Due to these difficulties the choice is often made to handle instance-level classification as an instance-level retrieval task [37]. Particular applications, *e.g.* in the product or art

35th Conference on Neural Information Processing Systems (NeurIPS 2021) Track on Datasets and Benchmarks.

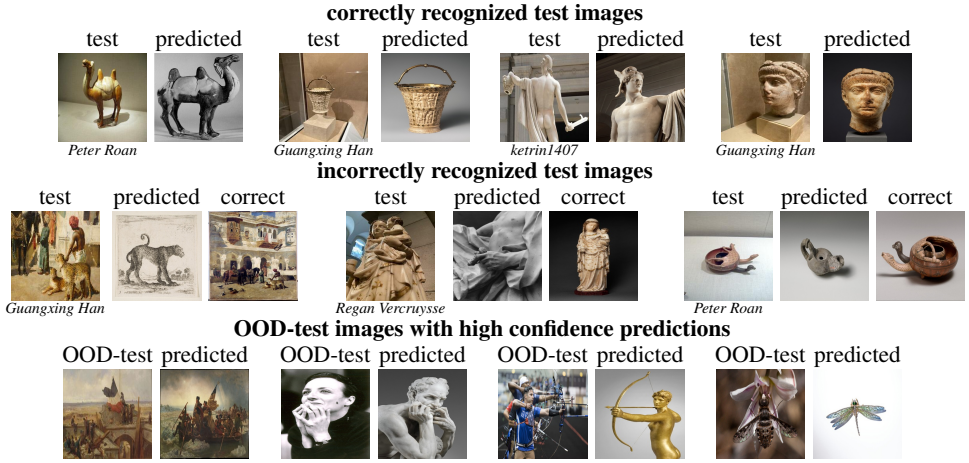


Figure 1: Challenging examples from the Met dataset for the top performing approach. Test images are shown next to their nearest neighbor from the Met exhibits that generated the prediction of the corresponding class. Top row: correct predictions. Middle row: incorrect predictions; an image of the ground truth class is also shown. Bottom row: high confidence predictions for OOD-test images; the goal is to obtain low confidence for these.

domain require dynamic updates of the category set; images from new categories are continuously added. Therefore, ILR is a form of open set recognition [16].

Despite the many real-world applications and challenging aspects of the task, ILR has attracted less attention than category-level recognition (CLR) tasks, which are accompanied by large and popular benchmarks, such as ImageNet [31], that serve as a testbed even for approaches applicable beyond classification tasks. A major cause for this is the lack of large-scale datasets. Creating datasets with accurate ground truth at large scale for ILR is a tedious process. As a consequence, many datasets include noise in their labels [8, 11, 39]. In this work, we fill this gap by introducing a dataset for instance-level classification in the artwork domain.

The art domain has attracted a lot of attention in computer vision research. A popular line of research focuses on a specific flavor of classification, namely attribute prediction [21, 27, 28, 36, 40]. In this case, attributes correspond to various kinds of metadata for a piece of art, such as style, genre, period, artist and more. The metadata for attribute prediction is obtained from museums and archives that make this information freely available. This makes the dataset creation process convenient, but the resulting datasets are often highly noisy due to the sparseness of this information [27, 36]. Another known task is domain generalization or adaptation where object recognition or detection models are trained on natural images and their generalization is tested on artworks [10]. A very challenging task is motif discovery [34, 35] which is intended as a tool for art historians, and aims to find shared motifs between artworks. In this work we focus on ILR for artworks which combines the aforementioned challenges of ILR, is related to applications with positive impact, such as educational applications, and has not yet attracted attention in the research community.

We introduce a new large-scale dataset (see Figure 1 for examples) for instance-level classification by relying on the open access collection from the Metropolitan Museum of Art (The Met) in New York. The training set consists of about 400k images from more than 224k classes, with artworks of world-level geographic coverage and chronological periods dating back to the Paleolithic period. Each museum exhibit corresponds to a unique artwork, and defines its own class. The training set exhibits a long-tail distribution with more than half of the classes represented by a single image, making it a special case of few-shot learning. We have established ground-truth for more than $1,100$ images from museum visitors, which form the query set. Note that there is a distribution shift between this query set and the training images which are created in studio-like conditions. We additionally include a large set of distractor images not related to The Met, which form an Out-Of-Distribution (OOD) [25, 30] query set. The dataset follows the paradigm and evaluation protocol of the recent Google Landmarks Dataset (GLD) [39] to encourage universal ILR approaches that are applicable in a wider range of domains. Nevertheless, in contrast to GLD, the established ground-

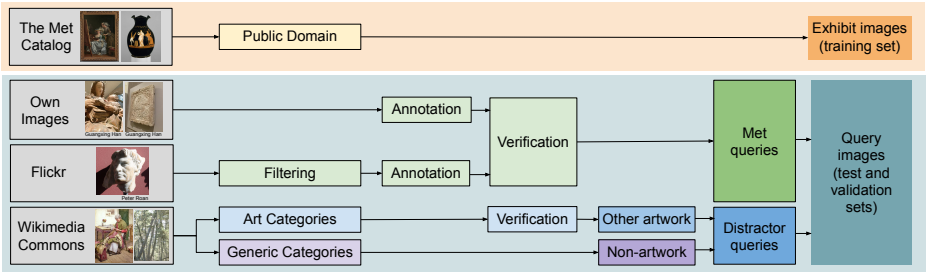


Figure 2: The Met dataset collection and annotation process.

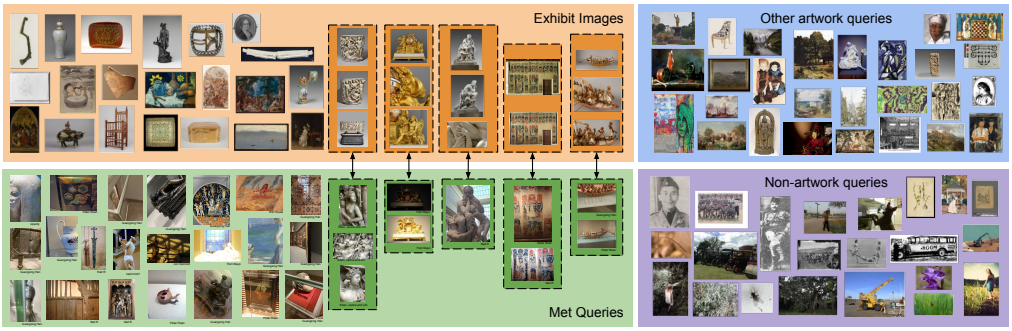


Figure 3: Samples from the Met dataset of exhibit and query (Met and distractor) images, demonstrating the diversity in viewpoint, lighting, and subject matter of the images. Exhibit images and queries from the same Met class are indicated by dashed lines.

truth does not include noise. To our knowledge this the only ILR dataset at this scale, that includes no noise in the ground-truth and is fully publicly available.

The introduced dataset is accompanied by performance evaluation of relevant approaches. We show that non parametric classifiers perform much better than parametric ones. Improving the visual representation becomes essential with the use of non-parametric classifiers. To this end, we show that the recent self-supervised learning methods that rely only on image augmentations are beneficial, but the available ILR labels should not be discarded. A combined self-supervised and supervised contrastive learning approach is the top performer in our benchmark indicating promising future directions.

2 The Met dataset

The *Met dataset* for ILR contains two types of images, namely *exhibit images* and *query images*. Exhibit images are photographs of artworks in The Met collection taken by The Met organization under studio conditions, capturing multiple views of objects featured in the exhibits. These images form the training set for classification and are interchangeably called exhibit or training images in the following. We collect about 397k exhibit images corresponding to about 224k unique exhibits, *i.e*. classes, also called *Met classes*.

Query images are images that need to be labeled by the recognition system, essentially forming the evaluation set. They are collected from multiple online sources for which ground-truth is established by labeling them according to the Met classes. The Met dataset contains about 20k query images, that are divided into the following three types: 1) *Met queries*, which are images taken at The Met museum by visitors and labeled with the exhibit depicted, 2) *other-artwork queries*, which are images of artworks from collections that do not belong to The Met, and 3) *non-artwork queries*, which are images that do not depict artworks. The last two types of queries are referred to as *distractor queries* and are labeled as "distractor" class which denotes out-of-distribution queries.

Dataset collection. The dataset collection and annotation process is described in the following and summarized in Figure 2, while sample images from the dataset are shown in Figure 3.

Split	Type	# Images			# Classes
		Met	other-art	non-art	
Train	Exhibit	397,121	-	-	224,408
Val	Query	129	1,168	868	111 + 1
Test	Query	1,003	10,352	7,964	734 + 1

Table 1: Number of images and classes in the Met dataset per split. Met exhibits images are from the museum's open collection, while Met query images are from museum visitors. Query images contain distractor images too (denoted by the +1 class) while the rest of val/test classes are subset of the train classes.

Image sources: Exhibit images are obtained from The Met collection.[1] Only exhibits labeled as open access are considered. A maximum of 10 images per exhibit is included in the dataset, images with very skewed aspect ratios are excluded, and image deduplication is performed. Query images are collected from different sources according to the type of query. Met queries are taken on site by museum visitors. Part of them are collected by our team, and the rest are Creative Commons (CC) images crawled from Flickr. We use Flickr groups[2] related to The Met to collect candidate images. Distractor queries are downloaded from Wikimedia Commons[3] by crawling public domain images according to the Wikimedia assigned categories. Generic categories, such as people, nature, or music, are used for non-artwork queries, and art-related categories, *e.g.* art, sculptures, painting, architecture, for other-artwork queries.

Annotation: We label query images with their corresponding Met class, if any. Met queries taken by our team are annotated based on exhibit information, whereas Met queries downloaded from Flickr are annotated in three phases, namely filtering, annotation, and verification. In the filtering phase, invalid images are discarded, *i.e.* images containing visitor faces, images not depicting exhibits, or images with more than one exhibit. In the annotation phase, queries are labeled with the corresponding Met class. To ease the task, the title and description fields on Flickr are used for text-based search in the list of titles from The Met exhibits included in the corresponding metadata. Queries whose depicted Met exhibit is not in the public domain are discarded. Finally, in the verification phase, two different annotators verify the correctness of the labeling per query. We additionally verify that distractor queries, especially other-artwork queries, are true distractors and do not belong to The Met collection. This is done in a semi-automatic manner supported by (i) text-based filtering of the Wikimedia image titles and (ii) visual search using a pre-trained deep network. Top matches are manually inspected and images corresponding to Met exhibits are removed.

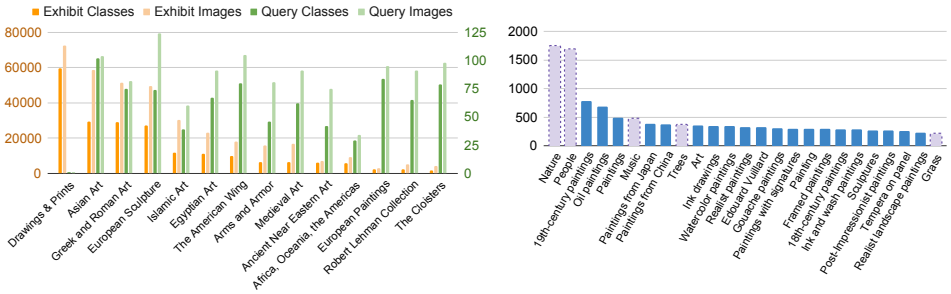


Figure 4: Left: Number of images and classes by department. Met queries are assigned to the department of their ground-truth class. Some departments that do not contain queries but contain exhibit images are not shown. Right: Number of distractor images by Wikimedia category. Top categories shown: art-related categories in solid blue and generic categories in dash purple.

Benchmark and evaluation protocol. The structure and evaluation protocol for the Met dataset follows that of the Google Landmarks Dataset (GLD) [39]. All Met exhibit images form the training

[1] https://www.metmuseum.org/

[2] https://www.flickr.com/groups/metmuseum/, https://www.flickr.com/groups/themet/, https://www.flickr.com/groups/mma_aaoa/

[3] https://commons.wikimedia.org/wiki/Main_Page

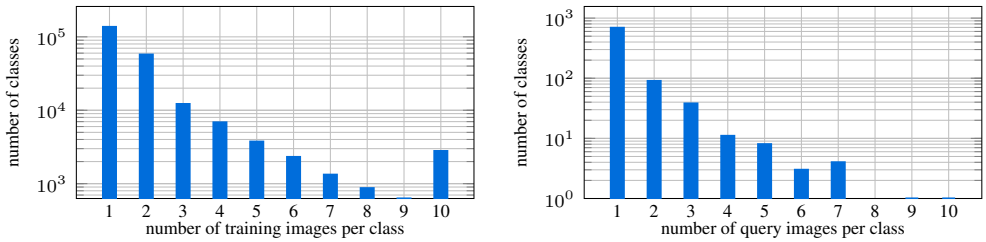


Figure 5: Left: number of Met classes versus number of training images per class. Right: number of Met classes versus number of query images per class.

set, while the query images are split into test and validation sets. The test set is composed of roughly 90% of the query images, and the rest is used to form the validation set. To ensure no leakage between the validation and test split, all Met queries are first grouped by user and then assigned to a split. Additionally, we enforce that there is no class overlap between the splits. As a result, 25 (14) users appear only in the test (validation) split, respectively. Image and class statistics for the train, val, and test parts are summarized in Table 1. The intended use of the validation split is for hyper-parameter tuning. All images are resized to have maximum resolution 500×500.

For evaluation we measure the classification performance with two standard ILR metrics, namely average classification accuracy (ACC), and Global Average Precision (GAP). The average classification accuracy is measured only on the Met queries, whereas the GAP, also known as Micro Average Precision (μAP), is measured on all queries taking into account both the predicted label and the prediction confidence. All queries are ranked according to the confidence of the prediction in descending order, and then average precision is estimated on this ranked list; predicted labels and ground-truth labels are used to infer correctness of the prediction, while distractors are always considered to have incorrect predictions. GAP is given by $\frac{1}{M}\sum_{i=1}^{T} p(i)r(i)$, where $p(i)$ is the precision at position i, $r(i)$ is a binary indicator function denoting the correctness of prediction at position i, M is the number of the Met queries, and T is the total number of queries. The GAP score is equal to the area-under-the-curve of the precision-recall curve whilst jointly taking all queries into account. We measure this for the Met queries only, denoted by GAP$^-$, and for all queries, denoted by GAP. In contrast to accuracy, this metric reflects the quality of the prediction confidence as a way to detect out-of-distribution (distractor) queries and incorrectly classified queries. It allows for inclusion of distractor queries in the evaluation without the need for distractors in the learning; the classifier never predicts "out-of-Met" (distractor) class. Optimal GAP requires, other than correct predictions for all Met queries, that all distractor queries get smaller prediction confidence than all the Met queries.

Dataset statistics. The Met dataset contains artworks spanning from as far back as $240,000$ BC to the current day. Figure 4 (left) shows the distribution of classes and images according to The Met department. Whereas there is an imbalance for exhibits across The Met departments, queries are collected to be evenly distributed to the best of our capabilities. In this way, we aim to ensure models are not biased towards a specific type of art, i.e., developing models that only produce good results for, e.g., European paintings, will not necessarily ensure good results on the overall benchmark. Finally, Figure 4 (right) shows the number of distractor query images by Wikimedia Commons categories.

The class frequency for exhibit images ranges from 1 to 10, with 60.8% and 1.2% classes containing a single and 10 images, respectively (see Figure 5 left). Met queries are obtained from 39 visitors in total, while the maximum number of query images per class is, coincidentally, also 10. In total, 81.5% of the Met query images are the sole Met queries that depict a particular Met class (see Figure 5 right).

Comparison to other datasets. We compare the Met dataset with existing datasets that are relevant in terms of domain or task.

Artwork datasets: Table 2 summarizes datasets in the artwork domain for various tasks. Most of the artwork datasets [21, 27, 28, 36, 40] focus on attribute prediction (AP), containing multiple types of annotations, such as author, material, or year of creation, usually obtained directly from the museum collections. Other datasets [5, 10, 40, 42] are focused on CLR, aiming to recognize object

Art datasets	Year	Domain	# Images	# Classes	Type of annotations	Task	Image source
PrintArt [5]	2012	Prints	988	75	Art theme	CLR	Artstor
VGG Paintings [10]	2014	Paintings	8,629	10	Object category	CLR	Art UK
WikiPaintings [21]	2014	Paintings	85,000	25	Style	AP	WikiArt
Rijksmuseum [28]	2014	Artwork	112,039	†6,629	Art attributes	AP	Rijksmuseum
BAM [40]	2017	Digital art	65M	†9	Media, content, emotion	AP, CLR	Enhance
Art500k [27]	2017	Artwork	554,198	†1,000	Art attributes	AP	*Various*
SemArt [14]	2018	Paintings	21,383	21,383	Art attributes, descriptions	Text-image	Web Gallery of Art
OmniArt [36]	2018	Artwork	1,348,017	†100,433	Art attributes	AP	*Various*
Open MIC [23]	2018	Artwork	16,156	866	Instance	ILR (DA)	*Authors*
iMET [42]	2019	Artwork	155,531	1,103	Concepts	CLR	The Met
NoisyArt [11]	2019	Artwork	89,095	3,120	Instance (noisy)	ILR	*Various*
The Met (Ours)	2021	Artwork	418,605	224,408	Instance	ILR	*Various*

Table 2: Comparison to art datasets. † For datasets with multiple annotations, the task with the largest number of classes is reported.

ILR datasets	Year	Domain	# Images	# Classes	Type of annotations	Image source
Street2Shop [17]	2015	Clothes	425,040	204,795	Category, instance	*Various*
DeepFashion [26]	2016	Clothes	800, 000	33,881	Attributes, landmarks, instance	*Various*
GLD v2 [39]	2019	Landmarks	4.98M	200,000	Instance (noisy)	Wikimedia
AliProducts [8]	2020	Products	3M	50,030	Instance (noisy)	Alibaba
Products-10K [2]	2020	Products	150,000	10,000	Category, instance	JD.com
The Met (Ours)	2021	Artwork	418,605	224,408	Instance	*Various*

Table 3: Comparison to instance-level recognition datasets.

categories, such as animals and vehicles, in paintings. From the artwork datasets, Open MIC [23] and NoisyArt [11] are the only ones with instance-level labels. Compared to the Met dataset, the Open MIC is smaller, with significantly less classes and mostly focuses on domain adaptation (DA) tasks. NoisyArt has a similar focus to ours, but is significantly smaller, and has noisy labels.

ILR datasets: In Table 3 we compare the Met dataset with existing ILR datasets in multiple domains. ILR is widely studied for clothing [17, 26], landmarks [39], and products [2, 8]. The Met dataset resembles ILR datasets in those domains in that the training and query images are from different scenarios. For example, in Street2Shop [17] and DeepFashion [26] queries are taken by customers in real-life environments, whereas training images are studio shots. Getting annotations for ILR, however, is not easy, and some datasets contain a significant number of noisy annotations from crawling from the web without verification [8, 11, 39]. In that sense, the Met is the largest ILR dataset in terms of number of classes, which have been manually verified. Overall, the Met dataset proposes a large-scale challenge in a new domain, encouraging future research on generic ILR approaches that are applicable in a universal way to multiple domains.

3 Baseline approaches

This section presents the approaches considered as baselines, *i.e*. existing methods that are applicable to this dataset, in the experimental evaluation.

Representation. Consider an embedding function $f_\theta : \mathcal{X} \to \mathbb{R}^d$ that takes an input image $x \in \mathcal{X}$ and maps it to a vector $f_\theta(x) \in \mathbb{R}^d$, equivalently denoted by $f(x)$. Function $f(\cdot)$ comprises a fully convolutional network (the backbone network), a global pooling operation that maps a 3D tensor to a vector, vector ℓ_2 normalization, and an optional fully-connected layer (also seen as 1×1 convolution), and a final vector ℓ_2 normalization. The backbone is parametrized by the parameter set θ. ResNet18 (R18) and ResNet50 (R50) [18] are the backbones used in this work, while global pooling is performed by Generalized-Mean (GeM) pooling [29], shown to be effective for representation in instance-level tasks [4].

Representation of image x, denoted by vector embedding $\mathbf{v}(x) \in \mathbb{R}^d$, is a result of aggregation of multi-resolution embeddings given by

$$\mathbf{v}(x) = \frac{\sum_{r \in R} f(x_r)}{|| \sum_{r \in R} f(x_r) ||}, \tag{1}$$

where x_r denotes image x down-sampled by relative factor r. We set $R = \{1, 2^{-0.5}, 2^{-1}\}$ and $R = \{1\}$ in the *multi-scale* (MS) and *single-scale* (SS) case, respectively. Following the standard practice in instance-level search, the image representation space is whitened with PCA whitening

(PCAw) [20] learned on the representation vectors of all Met training images. Optionally, dimensionality reduction is performed by keeping the dimensions corresponding to the top components. PCAw is always performed in the rest of the paper, unless stated otherwise; for simplicity we reuse notation $\mathbf{v}(x)$ for the whitened image embeddings. Given a trained backbone (fixed θ), the image representation is consequently used in combination with a k-Nearest-Neighbor (kNN) classifier.

kNN classifier. The label of image x is denoted by $y(x)$ and q is a query image. The similarity between query and a training image is given by $\mathbf{v}(x)^\top \mathbf{v}(q)$, coinciding with the cosine similarity. The confidence of class c for query q is given by

$$s_c(q) = \max_{x \in \mathrm{NN}_k(q)} (\mathbf{v}(x)^\top \mathbf{v}(q)) \mathbb{1}_{y(x)=c}, \tag{2}$$

where $\mathrm{NN}_k(q)$ is the set of k nearest-neighbors of q in the d-dimensional representation space. The vector of class confidences is $\mathbf{s}(q) \in \mathbb{R}^N$ with elements $s_c(q), c \in [1, \ldots, N]$, where N is the number of training classes. Classes without any example in the top-k neighbors have zero confidence. The predicted label $\hat{y}(q) = \arg\max_c s_c(q)$ is, according to (2), equivalent to the label of the closest training image. Despite label prediction requiring only $k = 1$, confidence estimation for more classes is essential for normalization and handling of OOD (distractor) queries. The normalized confidence is given by the soft-max of vector $\tau \mathbf{s}(q)$, where τ is the temperature. This is a non-parametric classifier that does not necessarily require training on The Met dataset; it only requires an existing backbone network. Hyper-parameters k and τ are tuned with grid search according to GAP on the validation set.

Training on the Met. We use the Met training set and perform either training of a classifier for the Met classes or training of the backbone to obtain image embeddings for the kNN classifier. During all variants of training the backbone the optional FC layer is included in the architecture and initialized with the result of PCA whitening [29].

Deep network (DNet) classifier with instance-level labels: The backbone is trained jointly with a cosine similarity (linear) classifier [38], used previously for training with imbalanced datasets [19], combined with one of the two following losses. Cross-Entropy (CE) loss with soft-max, where the input to the soft-max is equal to the cosine similarity between the backbone output and the learnable class vectors (prototypes) multiplied by temperature γ. Alternatively, we use the ArcFace (AF) loss [12], which is also used in the work of Cao *et al*. [4] for instance-level recognition of landmarks. During inference two options are considered. First, use the whole deep network classifier and consider its $\arg\max$ and $\max$ as class prediction and confidence score, respectively. Second, discard the linear classifier and use the backbone $f_\theta(\cdot)$ to obtain the image representation $v(x)$ and make predictions with the kNN classifier.

Simple-siamese (SimSiam) instance discrimination: We apply the recent self-supervised approach by Chen and He [7] to train the backbone. Each training image is augmented twice resulting in a positive pair, while no negative pairs and no Met labels are used in this approach.

Contrastive loss with synthetic/real positives and hard negatives: The backbone is trained with contrastive loss [9], where each training image is used as an anchor to form one positive and one hard negative pair per epoch. A hard-negative pair is formed by randomly choosing an image among the 10 most similar images from a different class, as these are computed according to embeddings obtained with the current backbone before each epoch. Three different ways of forming the positive pair are tested. *Syn*: the positive is an augmented (synthesized) version of the anchor image. *Syn+Real*: the selected positive is another randomly chosen image of the same class as the anchor, or an augmented version of the anchor image. Synthetic positive or one of the real (all images in the class but the anchor) positives is chosen with equal probability which is equal to one over the number of images in the class. If the class has a single image, then augmentation is performed; note that many classes contain a single image. *Syn+Real-closest*: same as *Syn+Real* but the real positive counterpart is chosen to be the one with the most similar embedding to the anchor. This is used to avoid images that depict completely different views of the object and has previously been used in location estimation [1]. Synthetic or real positive is chosen with equal probability in this case.

Pretrained models. We consider networks pretrained on other tasks and use them to obtain the image embeddings for the kNN classifier. None of these variants includes the optional FC layer in the architecture.

ID	Net	PCAw	MS	k	τ	GAP	GAP$^-$	ACC
1	R18IN			3	15	3.7	16.7	26.8
2	R18IN	✓		7	100	10.9	28.0	33.7
3	R18IN		✓	50	10	10.5	23.8	33.5
4	R18IN	✓	✓	3	50	**15.9**	**37.5**	**42.3**
5	R18IN	✓	✓	1	-	2.9	33.6	42.3
6	R18IN†	✓	✓	3	100	14.1	36.9	42.3

Table 4: Recognition performance for kNN classifier on representation obtained from ResNet18 pretrained on ImageNet. MS: multi-scale representation. †: tuning k, τ only with Met queries, and without distractor queries in the validation set.

Net	GAP	GAP$^-$	ACC
R18IN [18]	15.9 (+0.0)	37.5 (+0.0)	42.3 (+0.0)
R18SFM [29]	23.2 (+7.3)	41.5 (+4.0)	45.7 (+3.4)
R18SWSL [41]	**24.7 (+8.8)**	**47.0 (+9.5)**	**50.9 (+8.6)**
R50IN [18]	22.2 (+0.0)	41.8 (+0.0)	46.4 (+0.0)
R50SFM [29]	26.6 (+4.4)	44.8 (+3.0)	48.6 (+2.2)
R50SemArt (author) [13]	1.8 (-20.4)	12.2 (-29.6)	18.0 (-28.4)
R50SemArt (type) [13]	7.9 (-14.3)	26.8 (-15.0)	31.9 (-14.5)
R50SIN [15]	15.5 (-6.7)	36.4 (-5.4)	41.7 (-4.7)
R50SwAV [6]	22.8 (+0.6)	45.0 (+3.2)	49.6 (+3.2)
R50SWSL [41]	**30.4 (+8.2)**	**52.9 (+11.1)**	**56.3 (+9.9)**

Table 5: Comparison of recognition performance for kNN classifier with representation from backbone networks pretrained for different tasks. Relative improvements compared to the corresponding network trained on ImageNet are shown in parentheses.

ImageNet (IN) - classification: approach for training on ImageNet with cross-entropy loss [18]. *Landmarks (SfM) - metric learning:* approach for metric learning with contrastive loss on image pairs obtained from Structure-from-Motion on landmarks [29]. *Artwork attributes (SemArt):* networks trained on the SemArt dataset [14] by Garcia *et al*. [13] for artwork attribute prediction. In particular, we consider variants for painting type (10 classes) or author (350 classes). *StylizedImageNet (SIN):* network trained by Geirhos *et al*. [15] on a stylized version of ImageNet to improve the texture bias of deep networks. *SwAV on ImageNet (IN) - self supervision:* representation learning on ImageNet with self-supervision by instance discrimination. The resulting network has achieved good results in concept generalization [6]. *Semi-weakly supervised (SWSL) on Instagram 1G + ImageNet:* teacher-student approach [41] with teacher pretrained on about 1 billion images with hashtags and student trained with teacher-generated pseudo-labels, eventually fine-tuned on ImageNet.

4 Experiments

We perform performance evaluation of the baseline approaches using GAP and accuracy on the test queries of the Met dataset. Training, if any, is performed on the training part of the Met, while the validation queries are either used as validation set during the training or to tune the hyper-parameters of the kNN classifier. Multi-scale representation and PCA whitening with dimensionality reduction to 512D are used unless otherwise stated.

Image representation and kNN classifier components. ResNet18 trained on ImageNet is used as backbone to perform recognition with a kNN classifier. Hyper-parameters k and τ are tuned and reported separately per experiment in Table 4 which shows the impact of different components. The multi-scale representation and the use of whitening are essential parts of main approach (ID4 vs ID1,ID2, and ID3). Fixing $k = 1$ (ID5) is equivalent to no use of soft-max normalization and has significantly lower GAP on all queries, slightly lower GAP on Met queries, and identical accuracy by definition. Confidence normalization is therefore very important for handling distractors and high GAP performance. Finally, we show that having distractors in the validation set is boosting GAP by better kNN classifier hyper-parameter tuning (ID6 vs ID4).

Method	GAP	GAP$^-$	ACC
Parametric classification			
R18IN DNet CE	9.6	24.7	30.6
R18IN DNet AF	16.9	32.0	36.6
kNN classification			
R18IN (baseline)	15.9	37.5	42.3
R18IN DNet CE	21.6	40.4	44.7
R18IN DNet AF	23.7	43.9	47.4
R18IN SimSiam	26.8	42.3	45.6
R18IN Con-Syn	30.4	46.6	49.4
R18IN Con-Syn+Real	29.8	46.0	48.8
R18IN Con-Syn+Real-closest	**32.5**	**47.5**	**50.0**
R18SWSL (baseline)	24.7	47.0	50.9
R18SWSL Con-Syn+Real-closest	**36.1**	**52.4**	**55.0**

Table 6: Performance comparison for different types of training on the Met dataset. Training starts from the result of pretraining on ImageNet or that of SWSL. Baseline: not trained on the Met.

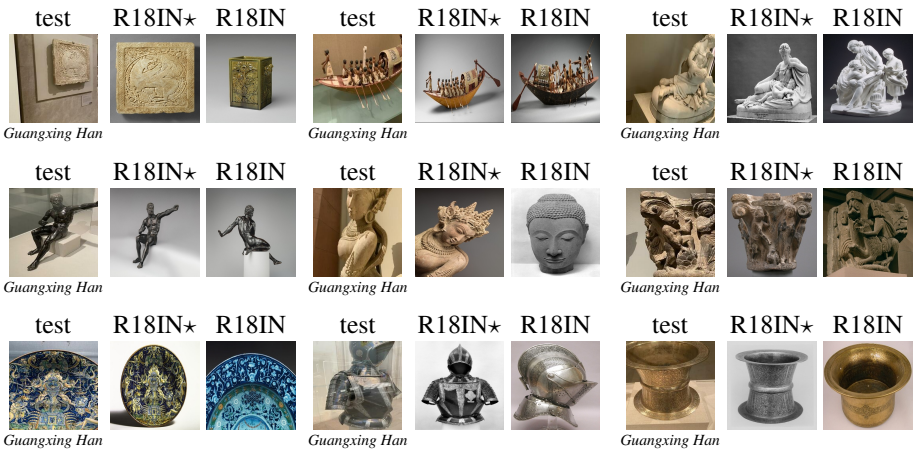


Figure 6: Examples of incorrect and correct classification of test images for R18IN (baseline) and R18IN Con-Syn+Real-closest (R18IN⋆), respectively. The test images are shown next to their nearest neighbor from the Met exhibits that produced the respective prediction per method.

Pretrained backbones and kNN classifier. Table 5 summarizes results of recognition performance with a kNN classifier for backbones pretrained on different tasks. Networks for art attribute prediction perform worse than the ImageNet ones, verifying that the task of art attribute prediction is far from that of ILR. The network for metric learning on landmarks provides improvements; despite the domain difference (artwork vs landmarks), training for metric learning well reflects the objectives of ILR. SwAV provides a performance boost, verifying the usefulness of unsupervised representation learning for better generalization. Finally, SWSL is the best performing variant demonstrating the benefits of learning on a very large image corpus despite the noisy labels; we expect the training set to include many artworks too.

Training on the Met dataset. Results from training on the Met dataset are shown in Table 6 with a parametric deep network classifier and with a kNN classifier. The latter is shown to be superior, while carrying the extra cost of storing a 512-D vector per training image. AF is shown to be better than CE, verifying prior results on ILR [4]. SimSiam improves the performance over the baseline without the use of any supervision indicating that self-supervised learning is a promising direction for ILR. Con-Syn uses the same positives as SimSiam but further boosts the performance by the use of negatives. Including real positives too with constrastive loss achieves the best performance but only if the positive pair is properly disambiguated (Real-closest vs Real). Improvements by training on the Met are confirmed starting from R18SWSL too. Examples where R18IN Con-Syn+Real-

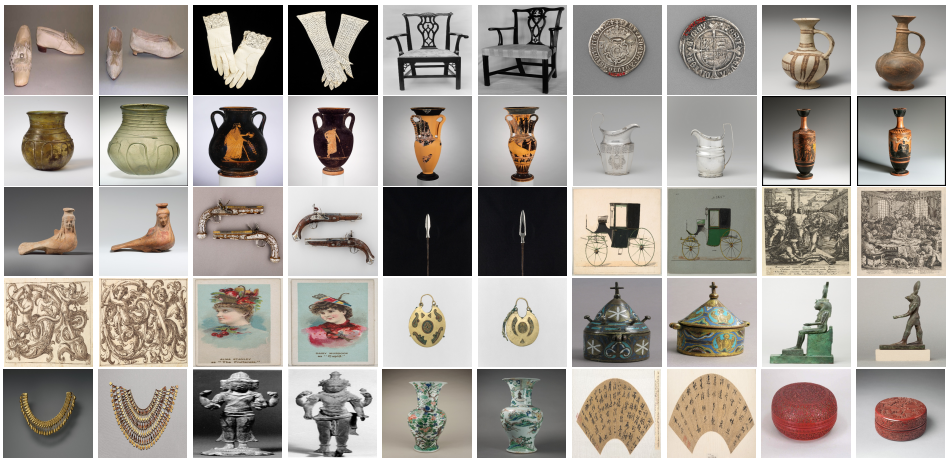


Figure 7: Examples of hard negative pairs formed by the approaches that use the Contrastive loss on the Met training set. These examples additionally demonstrate the large inter-class similarity of the dataset. Images are shown as squares only for the purposes of this figure.

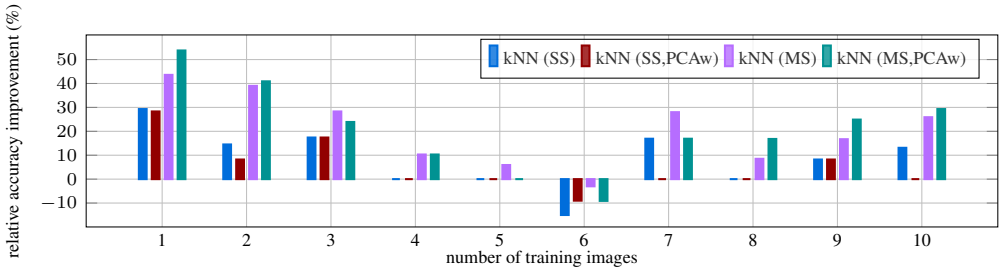


Figure 8: Accuracy improvement of the kNN classifier over the parametric one for varying number of training images per class. DNet is trained with AF loss for the parametric classifier, while the embeddings learned with this setup are used for the kNN classifier. Relative improvements are reported in percentage for the different embedding variants.

closest succeeds in prediction but the R18IN baseline fails are shown in Figure 6. These cases include challenges such as large view points changes and high inter-class similarity. Examples of hard negative pairs used in the contrastive variants are shown in Figure 7.

Few training examples and kNN classifier. We train a parametric classifier and additionally use the resulting embeddings for the kNN classifier. A comparison is shown in Figure 8, where performance is reported separately according to the number of training examples per ground-truth class of each query. The kNN classifier does not only perform better than the parametric one, but is shown to be more suitable for long tail recognition, as it achieves increasingly higher gains for more underrepresented classes.

5 Conclusions

This work introduces a new large-scale dataset for ILR on artworks. It is the first dataset on artworks to focus on this task, the only large-scale ILR dataset with clean annotations, and it poses a number of different challenges. The considered task is closer to ILR and deep representation learning than it is to popular computer vision tasks in the artwork domain, whilst including many of the same challenges. Fine-tuning the representation on The Met exhibits appears essential but also challenging due to the training set statistics. We expect this dataset to foster research not only on ILR for artworks but also for ILR across multiple domains, when combined with other existing datasets.

6 Acknowledgements

The authors would like to thank The Met employees Jennie Choi and Maria Kessler for their support and help, Andre Araujo, Tobias Weyand, and Xu Zhang for valuable discussions during the earlier stages of this work, and all the Flickr photographers whose photos are included in this dataset. This work was supported by JSPS KAKENHI No. JP20K19822, Junior Star GACR grant No. GM 21-28830M, and MSMT LL1901 ERC-CZ grant.

References

[1] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. In *CVPR*, 2016.

[2] Yalong Bai, Yuxiang Chen, Wei Yu, Linfang Wang, and Wei Zhang. Products-10k: A large-scale product recognition dataset. In *arXiv*, 2020.

[3] Vassileios Balntas, Shuda Li, and Victor Prisacariu. Relocnet: Continuous metric learning relocalisation using neural nets. In *ECCV*, 2018.

[4] Bingyi Cao, André Araujo, and Jack Sim. Unifying deep local and global features for image search. In *ECCV*, 2020.

[5] Gustavo Carneiro, Nuno Pinho Da Silva, Alessio Del Bue, and João Paulo Costeira. Artistic image classification: An analysis on the PRINTART database. In *ECCV*. Springer, 2012.

[6] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *NIPS*, 2020.

[7] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *CVPR*, 2021.

[8] Lele Cheng, Xiangzeng Zhou, Liming Zhao, Dangwei Li, Hong Shang, Yun Zheng, Pan Pan, and Yinghui Xu. Weakly supervised learning with side information for noisy labeled images. In *ECCV*, 2020.

[9] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *CVPR*, 2005.

[10] Elliot J Crowley and Andrew Zisserman. The state of the art: Object retrieval in paintings using discriminative regions. In *BMVC*, 2014.

[11] Riccardo Del Chiaro, Andrew D Bagdanov, and Alberto Del Bimbo. Noisyart: A dataset for webly-supervised artwork recognition. In *VISIGRAPP (4: VISAPP)*, 2019.

[12] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *CVPR*, 2019.

[13] Noa Garcia, Benjamin Renoust, and Yuta Nakashima. Context-aware embeddings for automatic art analysis. In *Proceedings of the ACM International Conference on Multimedia Retrieval*, 2019.

[14] Noa Garcia and George Vogiatzis. How to read paintings: semantic art understanding with multi-modal retrieval. In *ECCV Workshops*, 2018.

[15] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *ICLR*, 2019.

[16] Chuanxing Geng, Sheng-jun Huang, and Songcan Chen. Recent advances in open set recognition: A survey. *PAMI*, 2020.

[17] M Hadi Kiapour, Xufeng Han, Svetlana Lazebnik, Alexander C Berg, and Tamara L Berg. Where to buy it: Matching street clothing photos in online shops. In *ICCV*, 2015.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[19] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *CVPR*, 2019.

[20] Hervé Jégou and Ondřej Chum. Negative evidences and co-occurences in image retrieval: The benefit of pca and whitening. In *European conference on computer vision*. Springer, 2012.

[21] Sergey Karayev, Matthew Trentacoste, Helen Han, Aseem Agarwala, Trevor Darrell, Aaron Hertzmann, and Holger Winnemoeller. Recognizing image style. In *BMVC*, 2014.

[22] Jan Knopp, Josef Sivic, and Tomas Pajdla. Avoiding confusing features in place recognition. In *ECCV*, 2010.

[23] Piotr Koniusz, Yusuf Tas, Hongguang Zhang, Mehrtash Harandi, Fatih Porikli, and Rui Zhang. Museum exhibit identification challenge for the supervised domain adaptation and beyond. In *ECCV*, 2018.

[24] Jonathan Krause, Hailin Jin, Jianchao Yang, and Li Fei-Fei. Fine-grained recognition without part annotations. In *CVPR*, 2015.

[25] Shiyu Liang, Yixuan Li, and R Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *ICLR*, 2018.

[26] Ziwei Liu, Ping Luo, Shi Qiu, Xiaogang Wang, and Xiaoou Tang. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *CVPR*, 2016.

[27] Hui Mao, Ming Cheung, and James She. Deepart: Learning joint representations of visual arts. In *ACM Multimedia*, 2017.

[28] Thomas Mensink and Jan Van Gemert. The Rijksmuseum challenge: Museum-centered visual recognition. In *ICMR*, 2014.

[29] Filip Radenović, Giorgos Tolias, and Ondřej Chum. Fine-tuning cnn image retrieval with no human annotation. *PAMI*, 41(7):1655–1668, 2019.

[30] Jie Ren, Peter J Liu, Emily Fertig, Jasper Snoek, Ryan Poplin, Mark A DePristo, Joshua V Dillon, and Balaji Lakshminarayanan. Likelihood ratios for out-of-distribution detection. In *NIPS*, 2019.

[31] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 115(3):211–252, 2015.

[32] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 2015.

[33] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Fast image-based localization using direct 2d-to-3d matching. In *ICCV*, 2011.

[34] Benoît Laurent Auguste Seguin, Carlota Striolo, Isabella di Lenardo, and Frédéric Kaplan. Visual Link Retrieval in a Database of Paintings. In *ECCVW*, 2016.

[35] Xi Shen, Alexei A. Efros, and Mathieu Aubry. Discovering Visual Patterns in Art Collections With Spatially-Consistent Feature Learning. In *CVPR*, 2019.

[36] Gjorgji Strezoski and Marcel Worring. Omniart: a large-scale artistic benchmark. *TOMM*, 14(4):1–21, 2018.

[37] Giorgos Tolias, Tomas Jenicek, and Ondřej Chum. Learning and aggregating deep local descriptors for instance-level recognition. In *ECCV*, 2020.

[38] Feng Wang, Xiang Xiang, Jian Cheng, and Alan Loddon Yuille. Normface: L2 hypersphere embedding for face verification. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 1041–1049, 2017.

[39] Tobias Weyand, André Araujo, Bingyi Cao, and Jack Sim. Google landmarks dataset v2 - A large-scale benchmark for instance-level recognition and retrieval. In *CVPR*, 2020.

[40] Michael J Wilber, Chen Fang, Hailin Jin, Aaron Hertzmann, John Collomosse, and Serge Belongie. BAM! the behance artistic media dataset for recognition beyond photography. In *ICCV*, 2017.

[41] I. Zeki Yalniz, Hervé Jégou, Kan Chen, Manohar Paluri, and Dhruv Mahajan. Billion-scale semi-supervised learning for image classification. In *arXiv*, 2019.

[42] Chenyang Zhang, Christine Kaeser-Chen, Grace Vesom, Jennie Choi, Maria Kessler, and Serge Belongie. The iMet collection 2019 challenge dataset. In *arXiv*, 2019.