CENTER FOR
MACHINE PERCEPTION

CZECH TECHNICAL
UNIVERSITY IN PRAGUE

MASTER'S THESIS

ISSN 1213-2365

Large scale object detection

David Novotný

davnov134@gmail.com

CTU–CMP–2014–09

September 5, 2014

Thesis Advisor: Prof. Ing. Jiří Matas Dr.

Research Reports of CMP, Czech Technical University in Prague, No. 9, 2014

Published by

Center for Machine Perception, Department of Cybernetics
Faculty of Electrical Engineering, Czech Technical University
Technická 2, 166 27 Prague 6, Czech Republic
fax +420 2 2435 7385, phone +420 2 2435 7637, www: http://cmp.felk.cvut.cz

Large scale object detection

David Novotný

September 5, 2014

České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra kybernetiky

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student:	Bc. David N o v o t n ý
Studijní program:	Otevřená informatika (magisterský)
Obor:	Počítačové vidění a digitální obraz
Název tématu:	Detekce objektů z mnoha tříd

Pokyny pro vypracování:

Detekce objektů pocházejících z mnoha tříd je komplexní otevřený problém. Důraz je kladen na schopnost rozpoznávání velkého množství tříd.

1. Seznamte se s moderními metodami pro detekci objektů.
2. Vyberte metodu, zdůvodněte svou volbu a implementujte ji.
3. Vyhodnoťte její kvalitu z hlediska přesnosti a rychlosti na standartních datových sadách pro detekci objektů.
4. Navrhněte vylepšení referenční metody a ohodnoťte jeho přínos.

Seznam odborné literatury:

[1] Dean, Thomas, et al.: "Fast, Accurate Detection of 100,000 Object Classes on a Single Machine." IEEE Conference on Computer Vision and Pattern Recognition. 2013.
[2] Cinbis, Ramazan Gokberk, Verbeek, Jakob, and Schmid, Cordelia: "Segmentation Driven Object Detection with Fisher Vectors." International Conference on Computer Vision (ICCV). 2013.
[3] Wang, Xiaoyu, et al. "Regionlets for Generic Object Detection." International Conference on Computer Vision (ICCV). 2013.

Vedoucí diplomové práce: prof. Ing. Jiří Matas, Ph.D.

Platnost zadání: do konce letního semestru 2014/2015

L.S.

doc. Dr. Ing. Jan Kybic
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 10. 1. 2014

Czech Technical University in Prague
Faculty of Electrical Engineering

Department of Cybernetics

DIPLOMA THESIS ASSIGNMENT

Student:	Bc. David N o v o t n ý
Study programme:	Open Informatics
Specialisation:	Computer Vision and Image Processing
Title of Diploma Thesis:	Large Scale Object Detection

Guidelines:

Large scale generic object detection in images is a complex open problem. Focus on methods that are able to handle a large number of classes.

1. Familiarize yourself with the current state-of-the-art methods for object class detection.
2. Select a method, justify the choice and implement it.
3. Evaluate its performance in terms of the detection false positive rates and speed, using standard datasets.
4. Propose improvements of the reference method and evaluate their contribution.

Bibliography/Sources:

[1] Dean, Thomas, et al.: "Fast, Accurate Detection of 100,000 Object Classes on a Single Machine." IEEE Conference on Computer Vision and Pattern Recognition. 2013.
[2] Cinbis, Ramazan Gokberk, Verbeek, Jakob, and Schmid, Cordelia: "Segmentation Driven Object Detection with Fisher Vectors." International Conference on Computer Vision (ICCV). 2013.
[3] Wang, Xiaoyu, et al. "Regionlets for Generic Object Detection." International Conference on Computer Vision (ICCV). 2013.

Diploma Thesis Supervisor: prof. Ing. Jiří Matas, Ph.D.

Valid until: the end of the summer semester of academic year 2014/2015

L.S.

doc. Dr. Ing. Jan Kybic	prof. Ing. Pavel Ripka, CSc.
Head of Department	**Dean**

Prague, January 10, 2014

Firstly I would like to thank my supervisor Jiří Matas for his suggestions and guidance throughout my work on this thesis. I would also like to express appreciation to my former supervisors Andrea Vedaldi, Diane Larlus and Florent Perronnin for giving me an excelent intro to the field of object detection and image classification. Many thanks also belong to my family and friends for their support.

Prohlášení autora práce

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne ....

Podpis autora práce

Abstract

This thesis focuses on the problem of large scale visual object detection and classification in digital images. A new type of image features that are derived from state-of-the-art convolutional neural networks is proposed. It is further shown that the newly proposed image signatures bare a strong resemblance to the Fisher Kernel classifier, that recently became popular in the object category retrieval field. Because this new method suffers from having a large memory footprint, several feature compression / selection techniques are evaluated and their performance is reported. The result is an image classifier that is able to surpass the performance of the original convolutional neural network, from which it was derived. The new feature extraction method is also used for the object detection task with similar results.

Abstrakt

Tato práce se zabývá problémem detekce objektů z mnoha tříd a kategorizací v digitálních obrazech. Je navržen nový typ obrazových příznaků, které jsou odvozené od moderních konvolučních neuronových sítí. Dále je poukázáno na fakt, že se tato nová metoda podobá Fisher Kernel klasifikátoru, který byl v nedávné době úspěšně použit na klasifikaci obrazových dat. Nové příznaky vykazují velkou paměťovou náročnost, a proto je otestováno několik metod pro výběr a kompresi příznaků. Výsledkem je klasifikátor obrazů, jež je schopen překonat výsledky neuronové sítě, od které byl odvozen na úloze kategorizace obrazů. Tato nová metoda je také použita pro detekci objektů, přičemž bylo dosaženo podobných výsledků.

Contents

Used abbreviations

AP	Average Precision
BoW	Bag of Words
CNN	Convolutional Neural Network
CPU	Central Processing Unit
FGM	Feature Generation Machine
FK	Fisher Kernel
GMM	Gaussian Mixture Model
GPU	Graphics Processing Unit
HNM	Hard Negative Mining
HoG	Histogram of Oriented Gradients
mAP	Mean Average Precision
MI	Mutual Information
MKL	Multiple Kernel Learning
RBF	Radial Basis Function
PR	Precision-recall
ReLU	Rectified Linear Unit
SSR	Signed Square Rooting
SGD	Stochastic Gradient Descend
SIFT	Scale Invariant Feature Transform
SOTA	State of the Art
SSVM	Sparse Support Vector Machine
SVM	Support Vector Machine
tanh	hyperbolic tangent
VLAD	Vector of Linearly Aggregated Descriptors

1. Introduction

In this thesis a method for improving the performance of the state of the art convolutional neural networks is presented.

1.1. Motivation

Extracting higher level semantic information from images is one of the oldest and most commonly known computer vision tasks. In this thesis the problem of extracting the set of object categories present in a given image is studied. Finding an ultimate method that solves this problem has become a desired goal, mainly because of the huge amount of image data available, due to the increased popularity of various hand-held image devices. Searching in these large databases for an object of given category, by inspecting the the visual cues of individual images is as an extremely challenging task in the field of computer vision.

Every year the best computer vision labs submit their image classification and object detection pipelines to numerous contests that compare their system's performance (namely ImageNet Large Scale Visual Recognition Challenge [12], Pascal Visual Objects Classes challenge [15], Caltech-101 [16], etc.). In this challenging environment, even a slightest improvement of a state of the art image classification system is regarded as an interesting accomplishment.

In the past years, two main types of image classification systems managed to win the aforementioned competitions. The first were methods based on extracting orderless statistics of SIFT descriptors of image patches. Some notable methods are bag-of-words [10], VLAD [24] or Fisher Vectors [34]. The second and currently unsurpassed type of image classification systems are convolutional neural networks [18].

The methods proposed in this thesis are closely related to the Fisher Kernel (FK). The Fisher Kernel consists of deriving the loglikelyhood of a generative probability model w.r.t. its parameters and using this gradient as a feature vector. Sánchez, et.al. [32] have shown that by employing Fisher Kernel it is possible to obtain interesting results for image classification task[1].

However it seems that the computer vision community concentrates only on this one particular variant of the Fisher Kernel for image classification that uses Gaussian Mixture Model (GMM) as the underlying generative probabilistic model [32]. In this thesis a different approach is taken and instead of using GMM, a possibility of employing different underlying probability model is explored.

We propose a way how to extract image specific information from the state of the art convolutional neural network probability model (CNN), that is similar to the Fisher Kernel's derivatives of the loglikelyhood. Note that CNN is a discriminative model, so it is hard to obtain its generative loglikelyhood. We show that there is a possibility to obtain a different statistic, which is similar to the generative probability. After this is accomplished, the newly introduced statistic can be derived w.r.t. CNN's parameters, giving rise to features that bear a resemblance to original Fisher Vectors.

[1]Later Gokberk et. al. [9] have obtained compelling results on the object detection task.

The motivation here is that by combining the two recent SOTA image classification methods (CNN and FK), a superior classifier that picks the best the two methods is obtained.

1.2. Contribution

The following achievements were met in this thesis:

- A new image classification method, that **improves the performance of convolutional neural networks** on image classification task is proposed.
- Several feature selection / feature compression methods were tested and evaluated.
- The Multiple Kernel Learning based feature selection method [37] is extended such that it could be used in the multilabel classification task and its performance is evaluated with very positive results.
- A new object detection pipeline architecture is proposed and evaluated.

1.3. Thesis structure

The thesis is organized as follows. Chapter 2 contains brief introduction into the state of the art image classification and object detection techniques. Chapter 3 explains the proposed method for deriving Fisher Kernel based statistics from the CNN discriminative probability model. The section also shows how is the biggest problem of extremely high dimension of the introduced image features dealt with. Chapter 4 describes the architectures of the image classification and object detection systems used in this thesis. Chapter 5 contains the list of concluded experiments together with their discussion. Chapter 6 contains the conclusions of the work presented in this thesis.

2. State of the art image classification and object detection methods

In this part of the thesis, an introduction to the state of the art object detection and image classification methods is given.

2.1. Image classification

The image classification task consists of labeling input images with a probability of presence of a particular visual object class (dog, car, cat, ...). More precisely, given a set of images, $I = \{I_1, ..., I_n\}$ and a set of label vectors $Y = \{y_1, ..., y_K\}$, $y_i \in \{0, 1\}^{n \times 1}$ (where K stands for the number of classes), the task is to produce a set of predictions $\hat{Y} = \{\hat{y}_1, ..., \hat{y}_K\}$, that match Y as much as possible. Since this type of tasks is typically solved using machine learning techniques, the sets Y and I are split to testing and training subsets, while the performance of a given classification method is evaluated on the testing part.[1].

2.1.1. Datasets for image classification

Image classification is a very popular task in the field of computer vision and as such, many challenging benchmarks exist. The first and probably the most used is the Pascal VOC 2007 challenge benchmark [15]. Now community tends to use the ImageNet Large Scale Visual Recognition Challenge [12] which contains incomparably more object classes and images. Another notable dataset is Caltech 101 [16].

2.2. Object detection

The object detection task is closely related to the image classification one. The main difference lies in the fact that besides outputting the information about a presence or absence of a given class in a particular image, a position of an instance (or instances) of the class also has to be extracted (typically in form of a bounding box). A detection is regarded as true positive once the outputted bounding box has sufficiently large overlap with a ground truth object.

2.2.1. Datasets for object detection

The object detection data are typically included in the image classification benchmarks. This applies for already mentioned datasets [12] and [15].

[1]Note that for the purpose of generality, the problem is here defined as a multi-label classification task, which corresponds to the setting of the dataset [15] which is used in the experiments concluded in this thesis. However there exist datasets that actually define the image classification task as a multiclass problem (e.g. [16])

2.3. State of the art image classification pipelines

The first types of the image classification pipelines were mainly based on the orderless statistics such as bags of visual words [10] (BoW), which described the image as a histogram of quantized SIFT [30] keywords. Also [10] has presented a basic structure of image classification pipelines that continued to be used until today. The structure consists of two main stages - in the first one a raw image is converted to a different representation (feature extraction), which is in the second part used as the set of data vectors that is fed to a classifier (typically SVM [6][2]). The feature extraction stage tries to ease up the work of the classifier by embedding the raw image data to a space, where it is easier to perform classification.

The work of [10] was later improved by using so called *spatial pyramid* [28] which presented a way of incorporating spatial geometry into the BoW descriptors by counting the number visual words inside a set of image subregions.

Yang et.al. [42] defined the process of building a codebook of visual words as a sparse coding optimization problem and achieved state of the art results with this novel technique. They also employed a new feature pooling technique called *max-pooling*.

Later it has been shown that the bag of words strategy of accumulating just the zero order statistics (i.e. counts of visual words) discards a lot of valuable information about the image descriptors. A method that overcame this issue was introduced by Sánchez et.al. [32] and it consists of extracting higher order statistics by employing the Fisher Kernel [22]. Another notable feature encoding method called the Vector of Linearly Aggregated Descriptors [24] (VLAD) was proposed, however it has also been shown that VLAD is a simplification of the Fisher Kernel method [24].

In 2012 a major breakthrough in the field of image classification happened when Alex Krizhevsky el.al. managed to train a large convolutional neural network (CNN) [26] on the ImageNet database [12], thus giving a proof that CNNs could, besides the handwritten digit recognition [29], also perform well on the harder image classification task.

The presented CNN architecture (sometimes called ”Alexnet”) gave rise to a novel feature extraction technique, which consists of removing the top softmax layer of the CNN and using the lower activations (coming from the last fully connected layer) as discriminative image features. These could be used as generic image descriptors and in combination with a multiclass SVM classifier form a very powerful image categorization system [13] [7].

2.4. State of the art object detection pipelines

The first truly usable object detection system were sliding window classifiers computed on top of HoG [11] features. The original system from [11] was improved by Felzenschwalb et.al. [17] by introducing the Deformable Part Model, which consisted of discriminatively trained set of filters called ”parts”. Along with the appearance information contained in the set of HoG filters, the model was also able to capture geometrical information, thus being able to evaluate the score of a position of a given filter relative to the center of a bounding box.

[2]The convolutional neural networks (that are also described later in the text) use on the very top the softmax classification layer, however it has been shown that by replacing the last layer by a set of one-vs-rest SVM classifiers, the performance of the network is almost always better or stays roughly the same ([13], [7]).

Dalal et.al [11] have also described a heuristic for training object detectors called "hard negative mining" (HNM). Since in the object detection task, there is nearly infinite amount of negative training examples, it is convenient to pick a "representative" subset such that the classifier learned on top of these negatives gives high detection performance. The method of [11] alternates between adding the "hard negatives" (highest scoring false positive detections) as negative examples to the training set and retraining the object detector. This bootstrapping method [14] is now used in most of the state of the art object detection pipelines.

Another type of detectors was developed by utilizing the algorithms that are able to output a set of bounding boxes which are very likely to contain a generic visual object in an image ([38], [8], [1]). By reducing the number of evaluated bounding boxes per image to several hundreds, the object detection pipelines have now been brought close to the image classification ones. Both of them learn an SVM classifier on top of the set of descriptors extracted from positive and negative samples. The only difference lies in the construction of the set of negative training examples, where in the case of image classification it consists of all images that do not contain an instance of a positive class, whereas the object detection pipelines use the HNM procedure to obtain a subset of "hard" negative image subwindows.

The introduction of region proposals gave rise to some successful detection systems, based on the orderless bag of words like statistics. Some notable representatives are [40], [9], [38].

The current state of the art object detection systems [19], [20] use the window proposals from [38] together with the CNN image descriptors extracted using the ImageNet architecture from [26].

2.5. Convolutional neural networks

In image classification/ object detection field the convolutional neural networks [18] recently became very popular mainly because of the major success of the network created by Krizkevsky et.al. [26] that surpassed the previous state of the art method by almost 80 % on the ImageNet dataset [12].

The original idea of the convolutional neural network (CNN) was published by Fukushima [18]. Yann LeCun then improved the system [29] and achieved compelling results for the hand written digit recognition. His net consisted of stacked convolutional and subsampling layers. Convolutional layers improve the generalization properties by being connected only to a subpart of the neurons that reside in the next higher layer, thus the learned convolutional filters become invariant to translations. Furthermore the unwanted effect of vanishing gradients [4] is also partially eliminated [3]. The subsampling layers increase the capacity as well as the scale invariance of the CNN features.

The CNN that was the first of its kind that achieved results superior to bag-of-words like methods ([10], [32]) was created by Krizkevsky et.al. [26]. It uses, along with subsampling and convolutional layers, neurons wih ReLU activation functions [31] that speed up the convergence of the CNN training algorithm. Another addition are local response normalization layers that inhibit activities of adjacent highly active neurons. The last very important feature were dropout units (first introduced in [21]) which further reduced the degree of overfitting of the whole network.

The architecture of the state of the art ImageNet network is depicted in Figure 2.1. The lower layers consist of cascade of convolutional and maxpooling layers followed by two fully connected layers and the final softmax layer on the very top, which takes

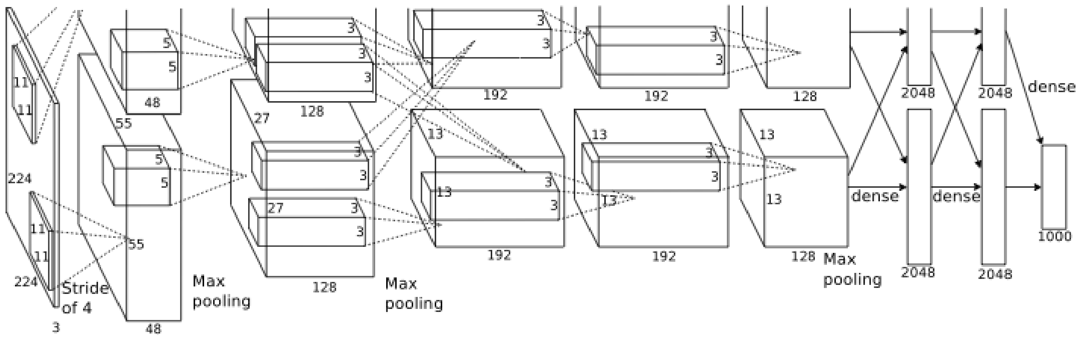


Figure 2.1. The state of the art ImageNet CNN architecture. The image was taken from [26].

care of making the final classification decisions. The whole network was trained on the ImageNet [12] dataset, which was the first very large dataset suitable for training such high-capacity architectures as deep convolutional neural networks.

Recently Chatfield et.al. [7] have made an exhaustive evaluation of different CNN architectures and achieved SOTA results on the Pascal VOC 2007 image classification database with a network who's architecture was only slightly different from the CNN described in [26].

2.6. Fisher Kernel

The *Fisher Vectors* [32] were the previous state of the art image classification framework that was later surpassed by the CNNs. Because the work in this thesis is closely related to the this method a short intro is given in this section.

To fully understand the *Fisher Vector* framework first the Fisher Kernel [22] has to be described.

Consider a class of generative models $P(X|\Phi)$, $\Phi \in \Theta$, where $X = \{x_1, ..., x_n\}$ denotes the set observed data samples x_i, Φ are the parameters of the model and Θ is the set of all possible settings of Φ.

The *Fisher score* U_X, is the gradient of the log-likelyhood of the generative model evaluated at the currently observed set of samples X_i

$$U_X = \nabla_\Phi log(P(X|\Phi))\Bigg|_{X_i} \tag{2.1}$$

this statistic could be seen as an indicator of how much one has to alter the model parameters Φ, such that the model better fits the currently observed data samples X_i. The *Fisher score* is used to measure similarity between two sets of samples, giving rise to the Fisher Kernel which is defined as:

$$K(X_i, X_j) = U_{X_i}^T I^{-1} U_{X_j} \tag{2.2}$$

where I is the *Fisher information matrix*, defined as

$$I = E_X[U_X U_X^T] \tag{2.3}$$

Because I is positive semidefinite, there is a possibility to perform the Cholesky decomposition of I and express the kernel function as a dot product of two column vectors

$$K(X_i, X_j) = \Upsilon_{X_i}^T \Upsilon_{X_j} \tag{2.4}$$

Where

$$\Upsilon_X = LU_X, \quad I = L'L \tag{2.5}$$

Such trick could be then used to train a linear classifier on top of the Υ_X data vectors, which is equivalent to learning the Fisher Kernel classifier on top of the sets of samples X.

Although in many image classification applications it is more convenient to train the linear classifiers, because of their superior convergence speed and simplicity of the implementation, it should be noted that in some of the upcoming experiments the dimensionality of Υ_X is so large that using the original kernel classifier is a necessity.

2.6.1. Improved Fisher Vectors

In [32], a Gaussian Mixture Model (GMM) , which models the generative process of SIFT [30] patches is used as an underlying generative model $P(X|\Phi)$. Here X stands for SIFT patches coming from images and Φ is a vector which consists of a concatenation of covariance matrices, means and priors of fitted Gaussians.

More precisely every SIFT patch x_i is represented by the *point-wise* Fisher Vector υ_i which is a concatenation of statistics $\upsilon_{i,k}^{(1)}$ and $\upsilon_{i,k}^{(2)}$.

$$\upsilon_i = \left[\upsilon_{i,1}^{(1)}\ \upsilon_{i,1}^{(2)}\ \upsilon_{i,2}^{(1)}\ \upsilon_{i,2}^{(2)}\ ...\ \upsilon_{i,K}^{(1)}\ \upsilon_{i,K}^{(2)}\right]^T \tag{2.6}$$

where $\upsilon_{i,k}^{(1)}$ stands for the derivative of the loglikelyhood function with respect to the k-th Gaussian mean μ_k evaluated at point x_i and $\upsilon_{i,k}^{(2)}$ is the derivative with respect to the k-th Gaussian covariance matrix σ_k.

$$\begin{aligned} \upsilon_{i,k}^{(1)} &= \frac{\partial\ logP(X|\Phi)}{\partial \mu_k}\bigg|_{x_i} = \frac{1}{\sqrt{\pi_k}} \frac{x_i - \mu_k}{\sigma_k} \\ \upsilon_{i,k}^{(2)} &= \frac{\partial\ logP(X|\Phi)}{\partial \sigma_k}\bigg|_{x_i} = \frac{1}{\sqrt{2\pi_k}} \left(\left(\frac{x_i - \mu_k}{\sigma_k} \right)^2 - 1 \right). \end{aligned} \tag{2.7}$$

Where π_k, σ_k, μ_k is estimated gaussian prior, covariance matrix and mean of the k-th gaussian in the GMM respectively.

Note that the derivatives with respect to the Gaussian priors are not expressed here, since these typically have no influence on the resulting classifier performance. Also the formulas (2.7) assume that the covariance matrices σ_k are diagonal, which is a common practice [34].

For the set of SIFT patches X coming from an image, it is desired to form a compact descriptor. The Fisher Vector Υ_X is a mean over all *point-wise* Fisher Vectors extracted from a given image.

$$\Upsilon_X = \frac{1}{n} \sum_{i=1}^{n} \upsilon_i \tag{2.8}$$

In [32] It has been shown that the performance of this Fisher Kernel classifier greatly improves if the Fisher Vectors Υ_X are further non-linearly transformed using the *signed square rooting* (SSR), i.e.:

$$\Upsilon_X^{SSR} = sgn(\Upsilon_X) \odot \sqrt{|\Upsilon_X|} \tag{2.9}$$

Also, because a linear classifier is typically used in combination with Fisher Vectors, it is further convenient to ℓ_2 normalize the data [32]:

$$\Upsilon_X^{SSR,\ell_2} = \frac{\Upsilon_X^{SSR}}{\|\Upsilon_X^{SSR}\|} \tag{2.10}$$

Vectors Υ_X^{SSR,ℓ_2} are then used as image descriptors and fed to a linear SVM solver.

3. Fisher Kernel based CNN features

In this work a combined approach of two recent major state of the art systems [32] and [26] is proposed.

In [32] the Fisher Kernel with Gaussian mixture model as an underlying generative model was proposed. In [26] a powerful multilayer discriminative model is trained to obtain state of the art results.

It is thus tempting to combine these two approaches to create a classification system that picks the best of the two hopefully giving a superior classifier as a result. In this work it is shown how to extract Fisher Kernel based features from arbitrary CNN, and how to use them as image descriptors.

3.1. Expressing generative likelyhood from a CNN

To be able to derive a Fisher Kernel from a probability model, first it is necessary to express its loglikelyhood function. As noted above the CNN is a discriminative model and therefore there should not be a way to express the function $P(X|\Phi)$ (recal that X are observed data variables, i.e. images and Φ is a set of CNN's parameters).

To show that there is a possibility to express the generative loglikelyhood function the topmost softmax layer of the CNN is inspected. Recall that the CNN's softmax function looks as follows:

$$p(C_k|x,\Phi) = \frac{exp(w_k^T\hat{x} + b_k)}{\sum_j exp(w_j^T\hat{x} + b_j)} \tag{3.1}$$

where C_k stands for the k-th class, w_k, b_k are tunable weights and bias respectively. In the case of CNN x is an image, and $\hat{x}$ are activations of the penultimate CNN layer. As it was shown in [5], the softmax function could be seen as an expression for the Bayes rule with tunable parameters w_k and b_k:

$$\frac{\exp(w_k^T\hat{x} + b_k)}{\sum_j \exp(w_j^T\hat{x} + b_j)} = \frac{p(x|\Phi, C_k)p(\Phi, C_k)}{\sum_j p(x|\Phi, C_j)p(\Phi, C_j)} = p(C_k|\Phi, x) \tag{3.2}$$

From there it is also possible to see that the joint probability $P(x, \Phi, C_k)$ (i.e. the nominator of Equation 3.2) is equal to:

$$P(x, \Phi, C_k) = p(x|\Phi, C_k)p(\Phi, C_k) = \exp(w_k^T\hat{x} + b_k) \tag{3.3}$$

To be able to derive a generative loglikelyhood function one has to be able to express the generative probability $P(X|\Omega)$, where Ω is a newly introduced symbol for the set of model parameters. In the case of CNN it is thus proposed to move the variables $C_1, ..., C_k$ into the set of model parameters (i.e. $\Omega = \{\Phi, C_1, ..., C_K\}$) such that it is possible to express the probability of set of images X conditioned on Ω.

At this point $P(x|\Omega)$ is defined as:

$$P(x|\Phi, C_1, ..., C_K) = P(x|\Omega) = \frac{P(x, C_1, ..., C_K, \Phi)}{P(C_1, ..., C_K, \Phi)} = \frac{P(\Phi, x)\prod_{k=1}^{K} P(C_k|\Phi, x)}{P(C_1, ..., C_K, \Phi)} \tag{3.4}$$

Where the independence of the probabilities $P(C_1|\Phi, x), ..., P(C_K|\Phi, x)$ is assumed. Note that this assumption arises from the probabilistic interpretation of the softmax activation function who's formula is given in (3.2).

Assuming that samples $P(x|\Phi, C_1, ..., C_K)$ are independent $P(X|\Phi, C_1, ..., C_K)$ then becomes:

$$P(X|\Phi, C_1, ..., C_K) = \prod_{i=1}^{n} P(x_i|\Phi, C_1, ..., C_K) \tag{3.5}$$

If we were to follow Fisher Kernel framework, at this point the expression for the derivative of the loglikelyhood of $P(X|\Phi, C_1, ..., C_K)$ would follow. However, there are several caveats that make this step very challenging:

Unknown $P(\Phi, x)$ and $P(C_1, ..., C_K, \Phi)$ The probabilities $P(\Phi, x)$ and $P(C_1, ..., C_K, \Phi)$ from (3.4) are unknown. Note that it would be possible to assume uniform prior over $P(C_1, ..., C_K, \Phi)$, however $P(\Phi, x)$ depends on the data x, which is a property that cannot be omitted in the Fisher Kernel setting.

Loglikelyhood derivative Even if there was a possibility to overcome the issues with unknown probabilities, obtaining the derivatives of the loglikelyhood function with respect to the parameter set Ω would be a very challenging task.

Instead of formulating unrealistic assumptions, that would help us with obtaining the final evaluable formulation of $P(X|\Phi, C_1, ..., C_K)$ we opt for defining our own function $\Lambda(x, \Phi, C_1, ..., C_K)$ that has similar properties as the probability $P(x|\Phi, C_1, ..., C_K)$:

$$\Lambda(x, \Phi, C_1, ..., C_K) = \prod_{k=1}^{K} P(x, \Phi, C_k) \tag{3.6}$$

Function Λ in our formulation of Fisher Kernel based features retakes the role of probabilities $P(x|\Phi, C_1, ..., C_K)$.

The FK classifier uses derivatives of the generative loglikelyhood function with respect to its parameters. Here since we use our own defined function $\Lambda(x, \Phi, C_1, ..., C_K)$, we call the expression, that is the equivalent to the generative likelyhood $\mathcal{L}(X|\Phi, C_1, ..., C_K)$ as *pseudo-likelyhood* $\hat{\mathcal{L}}_\Lambda$. It is defined as:

$$\hat{\mathcal{L}}_\Lambda(X, \Phi, C_1, ..., C_K) = \prod_{i=1}^{n} \Lambda(x_i, \Phi, C_1, ..., C_K) \tag{3.7}$$

At this point it is important to note that in the case of CNNs, the set of samples X actually consists of only one observation, which is the image x_i, i.e. in our case $n = 1$.

If the contents of (3.3) are plugged into the pseudo-likelyhood formula (3.7) the following is obtained:

$$\hat{\mathcal{L}}_\Lambda(X, \Phi, C_1, ..., C_K) = \prod_{i=1}^{n} \prod_{k=1}^{K} \exp(w_k^T \hat{x} + b_k) \tag{3.8}$$

Taking the logarithm of 3.8 the corresponding *pseudo-loglikelyhood* function is formed.

$$\boxed{\log \hat{\mathcal{L}}_\Lambda(X, \Phi, C_1, ..., C_K) = \sum_{i=1}^{n} \sum_{k=1}^{K} w_k^T \hat{x}_i + b_k} \tag{3.9}$$

After taking a derivative of the function $\log \hat{\mathcal{L}}_\Lambda(X, \Phi, C_1, ..., C_K)$ with respect to its parameters $\Phi, C_1, ..., C_K$, *Fisher Kernel based features* could be obtained.

Recall that the derivatives of (3.9) are not the proper Fisher Kernel features, because of our decision to replace probabilities $P(x|\Phi, C_1, ..., C_K)$ with $\Lambda(x, \Phi, C_1, ..., C_K)$ which cannot be regarded as generative probability measures. However our selection of $\Lambda(x, \Phi, C_1, ..., C_K)$ could be justified.

The purpose of the generative probability is to assign higher values of $P(x|\Phi, C_1, ..., C_K)$ to images x that are more likely to be observed. Our function Λ computes the product of unnormalized class posteriors $P(x, \Phi, C_k)$. Λ thus reaches high values once $P(x, \Phi, C_k)$ are also elevated. This means that from the view of Λ the images that are likely to appear are those that contain objects from classes $C_1, ..., C_K$. We hope that the fact that Λ assigns high values to the images that contain actual visual objects could be regarded as a justification of our choice of Λ function as a suitable replacement of $P(x|\Phi, C_1, ..., C_K)$.

From the Fisher Kernel point of view, the gradients of the loglikelyhood should have a "meaningful" form. This means that their directions should be designed such that it is possible to perform linear classification in this space. In the case of Fisher Kernel, the gradients of the models that are trained to maximize generative loglikelyhoods are used. The fact that the loglikelyhood of the model reaches its peak guarantees this property of the gradient directions. However it is not obvious whether the gradients of the aforementioned *pseudo-loglikelyhood* also exhibit this characteristic. While not giving any theoretical explanations, the empirical observations reported in Section 5 show that our Fisher Kernel based features are suitable for linear classification.

Another positive property of Λ is the simplicity of the resulting pseudo-loglikelyhood formula (3.9). Because the exponential terms got eliminated, the expression takes form of a simple sum of linear functions. Obtaining its derivative is then an easy task.

The fact that $n = 1$ in formula 3.7 could also be regarded as a theoretical issue, since the Fisher Kernel was originally defined to compare the sets of samples X which generally have more than one element. This issue could be for instance resolved by picking random crops or flips of the original image x and adding them to the set X. This extension would be another step for improving our proposed method, which is not covered in this thesis. Also note that in our particular case variables X and x are ambiguous and both represent image x.

To conclude, the gradients of Λ cannot be regarded as Fisher Kernel features, because of the reasons mentioned in the previous paragraphs. However the substitution of probability $P(x|\Phi, C_1, ..., C_K)$ by our own function Λ is the only difference between the Fisher Kernel and our proposed method. Thus in this thesis we choose to term the gradients of Λ *Fisher Kernel based features*, because of the evident resemblance to the original method.

3.2. Deriving Fisher Kernel based features from CNN

In the following section it is shown how to use the gradients of the pseudo-loglikelyhood derived from a CNN in combination with a SVM solver and use it for classifying images. (the pseudo-loglikelyhood formula is given in equation 3.9). The kernel function K_Λ that uses gradients of the pseudo-loglikelyhood and compares two sample sets X_i and X_j is the same as in the case of Fisher Kernel:

$$K_\Lambda(X_i, X_j) = U_{X_i}^T I^{-1} U_{X_j} \tag{3.10}$$

in this particular case U_X stands for the derivative of the pseudo-loglikelihood of the CNN (3.9) w.r.t. its parameters Ω evaluated at particular point (image) X_i:

$$U_X = \nabla_\Omega \log \hat{\mathcal{L}}_\Lambda \Big|_{X_i} \tag{3.11}$$

Furthermore it is again possible to utilize the cholesky decomposition of the matrix I and express kernel function $K(X_i, X_j)$ as a scalar product of two column vectors Υ_{X_i} and Υ_{X_j}.

$$K_\Lambda(X_i, X_j) = \Upsilon_{X_i}^T \Upsilon_{X_j} \tag{3.12}$$

Where

$$\Upsilon_X = L U_X, \quad I = L'L \tag{3.13}$$

After obtaining Υ_X the ℓ_2 normalization follows:

$$\Upsilon_X^{\ell_2} = \frac{\Upsilon_X}{\|\Upsilon_X\|_2} \tag{3.14}$$

Note that for brevity the vector $\Upsilon_X^{\ell_2}$ will be simply denoted as Υ_X. Also the kernel classifier formed by using derivatives of the pseudo-loglikelyhood of CNN will be termed *CNN-FK classifier*, the vectors Υ_X *Fisher Kernel based features* or shortly *CNN-FK features*.

From the implementation point of view, writing an algorithm that evaluates U_X seems like a complex task, because this amounts to compute a gradient of a very complex CNN mapping w.r.t. every parameter of the given CNN. However, since CNNs are learned using the stochastic gradient descend, these derivatives are always available, because these are the gradient updates of the CNN's parameters which are used during the SGD learning phase.

3.3. Memory issues

The CNN who's pseudo-loglikelyhood is derived has typically from 50 to 100 million parameters, which means that the **dimensionality of the data vectors Υ_X is extremely large** leading to mainly memory related issues that have to be dealt with.

The first apparent problem is the size of the matrix I, which scales quadratically with the number of dimensions of U_X (leading to $\sim 10^{16}$ elements). Thus in the experiments concluded in this thesis the approach of [32] is followed and the matrix I is assumed to be diagonal which means that the number of elements in I becomes the same as the dimensionality of U_X. Also obtaining an analytical formula for matrix I would be a complex task, thus an empirical estimate is used. More precisely the estimate of I is computed as follows:

$$I = \frac{1}{N} \sum_{i=1}^{N} U_{X_i} \odot U_{X_i} \tag{3.15}$$

Where $\{X_1, ..., X_N\}$ is a training set of images. The estimate of L is thus defined as:

$$L = \sqrt{\frac{1}{N} \sum_{i=1}^{N} U_{X_i} \odot U_{X_i}} \tag{3.16}$$

Another issue that makes the learning problem hard is that a larger portion of these extremely high dimensional features cannot be fit into the memory (5000 training vectors - which is the amount of training data used in the Pascal VOC 2007 classification challenge - would occupy 2 TB of memory), thus it is problematic to learn a SVM classifier in its primal formulation. The following paragraph explains the memory issues and shows how they are dealt with.

Perhaps the most crucial issue is that impossibility of training a SVM linear classifier in its primal form on top of the raw Υ_X vectors. In this thesis the problem is solved by utilizing the following four feature selection / compression methods:

Feature binarization The size of vectors Υ_X is significantly reduced using a lossy compression technique (binarization).

Mutual information based feature selection By employing the mutual information measure between individual dimensions of Υ_X and the set of training labels y, one can remove dimensions with the lowest values of this metric.

SVM dual formulation The SVM learning problem is optimized in its dual form leading to a more memory efficient representation.

MKL based feature selection A Multiple Kernel Learning (MKL) solver is used to obtain the most discriminative subset of feature dimensions.

The five aforementioned methods are described in the ongoing subsections.

3.3.1. Feature binarization

Recently Zhang et.al. [43] have proposed a feature compression scheme for Fisher Vectors [32] which consists of binarizing the features and then removing the unimportant dimensions based on a mutual information measure. They have also shown that just by binarizing the features the classifier performance could be actually improved.

In this thesis the binarization technique is tested as a potential way to compress the Υ_X vectors. More precisely denote the vector Υ_X^{BIN} as the binarized version of Υ_X. The binarization process is expressed in the following formula:

$$\Upsilon_{X_d}^{BIN} = \begin{cases} 1 & \Upsilon_{X_d} \geq 0 \\ -1 & \Upsilon_{X_d} < 0 \end{cases} \tag{3.17}$$

Where $\Upsilon_{X_d}^{BIN}$ is the d-th dimension of the feature vector Υ_X^{BIN}. This function corresponds to the quantization of each feature dimension into two bins with the bin decision threshold set to zero.

It is then easy to represent each dimension of Υ_X^{BIN} by a single bit value and decrease the memory usage by the factor of 32 (assuming that uncompressed data are 32-bit floating point values). By utilizing the SVM solver tricks from [43], a standard SGD solver (e.g [35]) could then be used to learn a linear classifier on top of Υ_X^{BIN} training vectors.

3.3.2. Mutual information based feature selection

Furthermore the feature selection technique from [43] was also experimented with. The method uses a mutual information based approach to select the most relevant dimensions given the training data labels. The mutual information measure is computed between a joint probability distribution of each binarized descriptor dimension and the set of training labels. More precisely the mutual information is defined as:

$$I(\Upsilon_{X_d}^{BIN}, y) = H(y) + H(\Upsilon_{X_d}^{BIN}) - H(\Upsilon_{X_d}^{BIN}, y) \tag{3.18}$$

Where y is a set of training labels. Because $H(y)$ remains unchanged while varying dimension d it could be removed from the above formula giving:

$$I(\Upsilon_{X_d}^{BIN}, y) = H(\Upsilon_{X_d}^{BIN}) - H(\Upsilon_{X_d}^{BIN}, y) \tag{3.19}$$

Note that entropy H is defined for probability distributions and not for arbitrary real values. This is the reason why the binarized fisher vectors $\Upsilon_{X_d}^{BIN}$ are figuring in Equation (3.19). Once the features are binarized it is then easy to estimate the probabilities of individual bins on a training set. This enables the use of entropy measures. The same applies for the set of labels y, with the difference that no binarization is needed, because y is already discrete.

After the computation of the mutual information measure for each of the dimensions of Υ_{X_i}, it is then possible to discard given number of dimensions that have the lowest magnitude of $I(\Upsilon_{X_d}^{BIN}, y)$, thus performing feature selection. Note that although the mutual information measures are computed on binarized features Υ_X^{BIN}, in this thesis these measures are used to remove dimensions of raw descriptors Υ_X. Thus in this case the binarization is just a proxy for obtaining probabilities and mutual information measures.

The described algorithm that uses mutual information to select features will be termed **MI-FS** in the next lines.

3.3.3. SVM dual formulation

Later in this thesis, it is possible to see that the feature compression technique described in section 3.3.1 discards a lot of information and can lead to inferior results. It is thus convenient to learn a SVM solver on the raw uncompressed Υ_X vectors and optimize the SVM objective in its dual representation who's memory usage scales quadratically with the number of training samples and as such, it is not dependent on the dimensionality of the features Υ_X.

For completeness, the dual of the SVM objective function that is optimized is defined as follows [6]:

$$\begin{aligned} &\arg\max_{\alpha_i} \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \Upsilon_{X_i}^T \Upsilon_{X_j} \\ s.t.:\ &\alpha_i \geq 0, \\ &\sum_i \alpha_i y_i = 0 \end{aligned} \tag{3.20}$$

where α_i are support vector coefficients and y_i are labels of the training vectors.

Here it is possible to see that training vectors Υ_X appear only in the form of dot products $\Upsilon_{X_i}^T \Upsilon_{X_j}$ thus there is no need to keep them in the memory in their explicit form.

3.3.4. Feature selection via MKL

Because a SVM classifier is learned on top of the high dimensional vectors Υ_X it is tempting to use a feature selection method that is strongly related to the original SVM optimization problem. In this thesis a slightly modified SVM solver is used that performs feature selection by inducing sparsity into the vector of weights w. Once the sparse vector is obtained, the zeroed dimensions of w could then be removed from feature vectors Υ_X.

In this thesis the feature selection problem is regarded as a task, who's goal is to train a Sparse SVM classifier (SSVM) [41] which is defined as:

$$\begin{aligned}
&\arg\min_{d\in\mathcal{D}}\min_{w}\frac{1}{2}\|w\|^2+\frac{C}{2}\sum\max(1-y_i w^T(x_i\odot d),0)\\
&w\in R^{\delta},\quad x_i\in R^{\delta},\quad \mathcal{D}=\{d|\sum_i^{\delta}d_i\leq B, d_i\in\{0,1\}\}
\end{aligned}\tag{3.21}$$

Where B stands for the number of dimensions that are required to have non-zero weights and δ is dimensionality of the features x_i[1] . $\mathcal{D}$ is the set of all possible configurations of binary vectors d.

Luckily Tan et.al. [37] have developed a solver for this optimization problem which they call FGM. It optimizes the convex relaxation of the Mixed Integer Programming problem from Equation (3.21). A brief description of the method is given in the following paragraphs and we refer to [37] for additional details.

First the SSVM problem is converted to its dual representation and then a multiple kernel learning [27] (MKL) optimization problem is defined as follows:

$$\begin{aligned}
&\min_{\mu\in\mathcal{M}}\max_{\alpha\in\mathcal{A}}-\frac{1}{2}(\alpha\odot y)^T(\sum_{d_t\in\mathcal{D}}\mu_t X_t X_t^T)(\alpha\odot y)\\
&\mathcal{M}=\{\mu|\sum\mu_t=1,\mu_t\geq 0\}\\
&\mathcal{A}=\{\alpha|\sum_{i=1}^{n}\alpha_i y_i=0,\alpha_i\geq 0\}\\
&X_t=[x_1\odot d^t,...,x_n\odot d^t]
\end{aligned}\tag{3.22}$$

Note that instead of presenting the problem with squared empirical loss as it happens in [37], the hinge empirical loss function is used in Equations 3.21 and 3.22.

[1] To avoid confusion, we note that for this particular thesis section x_i will stand for the SVM features

The objective function from Equation 3.22 is then optimized using a cutting plane method [25] which alternates between two main steps:

1. Adding constraints corresponding to the most violated d_t, i.e.:

$$d_{t+1} = \max_{d \in \mathcal{D}} \frac{1}{2} \| \sum_{i=1}^{n} \alpha_i y_i (x_i \odot d) \|^2 \tag{3.23}$$

   This problem can be solved trivially by sorting the dimensions of the vector $c = \sum_{i=1}^{n} (\alpha_i y_i x_i)^2$ in a descending order and setting first B numbers to 1, and the rest to 0.
2. Solving an MKL subproblem defined by Equation 3.22 with individual d_ts fixed to obtain the kernel weights μ_t and dual coefficients α_t (μ_t and α_t are then fixed and used in step 1).

These two steps are cyclically repeated until the convergence is reached.

3.3.5. Extending FGM to multilabel classification

Recall that the FGM solver from [37] does a binary classification, thus in its original form it cannot be used in our multilabel task of image classification. A simple modification to the objective function in Equation (3.22) is proposed in this thesis to enable learning a sparse SVM model on a multi-label classification task:

$$\min_{\mu \in \mathcal{M}} \sum_{k=1}^{K} \max_{\alpha_k \in \mathcal{A}} -\frac{1}{2} (\alpha_k \odot y_k)^T (\sum_{d_t \in \mathcal{D}} \mu_t X_t X_t^T)(\alpha_k \odot y_k) \tag{3.24}$$

The above written objective simply sums over all the loss functions and sets the optimal d_t vector to the one that gives the best value of the objective function accumulated over all the classes (the number of classes is denoted by K).

The original FGM algorithm needs two minor modifications:

1. The step of finding the most violated d_t from Equation 3.25 becomes:

$$d_{t+1} = \max_{d \in \mathcal{D}} \frac{1}{2} \sum_{k=1}^{K} \| \sum_{i=1}^{n} \alpha_{i,k} y_{i,k} (x_i \odot d) \|^2 \tag{3.25}$$

   which again has a simple solution of sorting the values of the vector $c = \sum_{k=1}^{K} \sum_{i=1}^{n} (\alpha_{i,k} y_{i,k} x_i)^2$ in the descending order and regarding the first B dimensions as the new d_{t+1}.
2. Once the most violated d_ts are fixed and new μ and α parameters are required the problem (3.24) can be solved by the multiclass version of the SimpleMKL algorithm [33].

In the following lines the above described multilabel extension of the FGM algorithm will be termed Multilabel Feature Generation Machine (ML-FGM). Also the feature vector Υ_X consisting only of the features selected by the ML-FGM will be termed Υ_X^{FGM}.

There are two remarks of the proposed approach that are worth noting:

Memory efficiency The main property of the proposed ML-FGM supervised feature selection method is the fact that during learning only kernel matrices are used, thus the algorithm is memory efficient, solving the problem with extremely high feature dimension.

Universality of selected features The proposed improvement of extending the original FGM binary classification task to a multilabel classification task makes it possible to obtain a set of universal features that could be used in image classification, regardless of the object class that is being recognized. This is somehow similar to the CNN models, that are also learned in the supervised way on the ImageNet database to recognize a set of particular object categories. However because the object categories have similar properties, the learned features are able to capture generic image patterns and could then be used for recognizing different unseen objects. In this thesis the transfer of the selected features from the image classification to object detection task is demonstrated.

3.4. Combining CNN-FK with CNN neuron activities

When obtaining the gradients of the pseudo-loglikelyhood function of the CNN U_X (3.11) that are after normalization used as feature vectors Υ_X, the standard bottom-up pass through the layers of the CNN has to be performed, producing the activations of the neurons in the penultimate fully connected layer as a byproduct. It is thus convenient to use these activations (which, as has been written above, are current state of the art image features in several image recognition tasks [19], [2], [13], [7]) in combination with the proposed CNN-FK classifier. In this thesis two approaches to this problem are proposed:

Late fusion Another classifier on top of the SVM scores outputted from the CNN-FK classifier and the SVM classifier trained on top of the CNN activations is trained. Its classification output is then used as a final measure of probability of a class being present in an image.

Early fusion The vector of neuron activities of the penultimate CNN layer is appended to the CNN-FK features, and the set of resulting features is then fed to the linear SVM classifier.

The following sections give more insight on the two proposed classifier combination methods.

Late fusion

Denote $s_{i,k}^{CNN}$ the score that a SVM classifier learned on top of the CNN activation vectors to recognize class k assigns to an image X_i[2]:

$$s_{i,k}^{CNN} = \langle w_k^{CNN}, \hat{x}_i \rangle + b_k^{CNN} \tag{3.26}$$

and also denote the score of the corresponding CNN-FK classifier $s_{i,k}^{CNN-FK}$:

$$s_{i,k}^{CNN-FK} = \langle w_k^{CNN-FK}, \Upsilon_{X_i} \rangle + b_k^{CNN-FK} \tag{3.27}$$

[2]Recall that $\hat{x}_i$ stands for the set of the neuron activations of the penultimate CNN layer.

the set of features $\{z_{i,k}\}$ that is fed to the final combined classifier are 2-dimensional concatenations of $s_{i,k}^{CNN}$ and $s_{i,k}^{CNN-FK}$:

$$z_{i,k} = \begin{bmatrix} s_{i,k}^{CNN} & s_{i,k}^{CNN-FK} \end{bmatrix}^T \tag{3.28}$$

Once all the $z_{i,k}$ values are obtained, K one-versus-rest SVM classifiers are learned for each set of features $Z_k = \{z_{i,k} | i = \{1...n\}\}$ and labels y_k. Because the used features are only 2-dimensional, there is a possibility to employ a nonlinear kernel SVM. In this thesis RBF, hyperbolic tangent (tanh) and polynomial kernels were tried.

Early fusion

Denote the vector of neuron activities coming from the penultimate CNN layer as ϕ_X (we will continue to use this also symbol in the following sections). The set of final features that are fed to the SVM training algorithm is $\{\mathcal{F}_1, ..., \mathcal{F}_n\}$, where $\mathcal{F}_i$ stands for:

$$\mathcal{F}_i = \begin{bmatrix} \Upsilon_{X_i} \\ \phi_{X_i} \end{bmatrix} \tag{3.29}$$

Note that Υ_X here stands for an arbitrary CNN-FK feature vector, i.e. instead of raw vectors Υ_X their compressed version (by utilizing methods from Section 3.3) could be used.

4. Description of used pipelines

To evaluate the performance of the CNN-FK features, image classification and object detection systems were implemented and used as benchmarks.

4.1. Image classification pipelines

This section describes individual steps of the image classification pipelines that are used in the experiments later in this thesis. Figure 4.1 contains a sketch of the architectures of the three used pipelines.

The first part of the pipelines is common and consists of extracting features from each image X. This step involves either obtaining Fisher Kernel based features Υ_X or activities of neurons denoted as ϕ_X that reside in the last fully connected layer of the CNN.

After extracting the features, optional compression / feature selection steps follow. The compressed or uncompressed features are then either appended with neuron activities ϕ_X (early fusion) or stay unchanged. A multiclass SVM classifier is then learned on top of these features, which are extracted from the training set of images. The multiclass SVM classifier is trained in the one-vs-rest fashion. The SVMs can be optimized in dual or primal depending on the dimensionality of the features that enter them.

At this point the scores outputted by the classifiers could be regarded as a final classification posterior probabilities or their outputs could be fed to another non-linear kernel SVM classifier (late fusion).

On the following lines, the three main classification pipeline architectures are described. Each is designed to compare different methods proposed throughout this thesis.

4.1.1. Late fusion pipeline

The first pipeline is designed to compare the performance of various late fusion approaches and the binarization compression explained in Section 3.3.1.

The system is further divided to five subtypes depending on the features that are used and whether the binarization is employed:

CNN-ϕ_X A standard architecture first introduced in [13]. It learns a linear SVM on top of the ϕ_X neuron activities.

CNN-Υ_X The pipeline that learns SVM in dual formulation on the raw Υ_X CNN-FK feature vectors.

CNN-Υ_X^{BIN} This architecture learns a linear SVM on top of the binarized Υ_X features. No late fusion method is used.

CNN-$\Upsilon_X + \phi_X$ Late fusion utilizing the scores outputted by CNN-ϕ_X and CNN-Υ_X.

CNN-$\Upsilon_X^{BIN} + \phi_X$ Late fusion approach that uses scores of linear SVM learned on binarized Υ_X and the scores of the CNN-ϕ_X classifier.

Note that both late fusion systems used non-linear kernel classifier to learn the combination of the input linear SVM scores.

4.1.2. CNN-Υ_X^{FGM} and CNN-Υ_X^{MI}

To compare the performance of the MKL based feature selection algorithm (ML-FGM explained in Section 3.3.5) and the feature selection that uses mutual information measures (MI-FS algorithm from Section 3.3.2), two more pipelines are used - CNN-Υ_X^{FGM} which is designed to test ML-FGM and CNN-Υ_X^{MI} that uses MI-FS.

The pipeline first extracts raw Υ_X features. Utilizing one of the two aforementioned feature selection methods the descriptor dimensionality is reduced. We tried different values of the target dimensionality. In our experiments the sizes of descriptors were reduced by the factors of $10^4, 10^3, 10^2$, 10^1 and 1 (no reduction).

Afterwards a standard one-vs-rest SVM classifier is learned and its outputs are used as final classification scores.

4.1.3. CNN-$\Upsilon_X^{FGM}\&\phi_X$ early fusion pipeline

For comparing the performance of the early-fused vectors the last image classification system CNN-$\Upsilon_X^{FGM}\&\phi_{x_i}$ was created.

At the beginning CNN-FK features together with the neuron activations are extracted from each image. The Fisher Kernel based statistics are then compressed using ML-FGM algorithm (again several different dimensionality reduction factors were tried). The Υ_X^{FGM} and ϕ_X are then concatenated and fed to a linear SVM solver which performs classification.

4.2. Object detection pipeline

In this thesis the performance of the CNN-FK features on the object detection task is also evaluated. The proposed detection system is described in this section.

Because CNN-FK features are expensive to compute, it is a necessity to decrease the amount of these that is extracted from each image. Thus the detection pipeline uses a multi-stage cascade, where at earlier levels, the features that are easier to obtain are used and with later stages the amount of bounding boxes that have to be evaluated decreases. The CNN-FK features are used at the latest stage of the system.

The architecture of the detection pipeline that is used in this thesis is a three stage cascade system. The first and second step of the cascade is basically the method proposed by Girshick in [19], i.e. it uses the CNN neuron activations as descriptors for bounding boxes extracted using the method from [38].

4.2.1. First stage

The first stage is based on the tentative object proposals from [38]. Using this algorithm 2000 bounding boxes that have high probability of containing an object of interest are obtained from each image.

4.2.2. Second stage

Then similar to [19] the image is cropped around each bounding box and every crop is fed to the CNN to obtain the activities of neurons in the first fully connected layer. These are then ℓ_2 normalized and scored by each of the K (K denotes number of classes) linear SVM classifiers, each of them trained specifically for a given class. The non-maxima suppression is then performed on each of the K sets of scored detections separately.

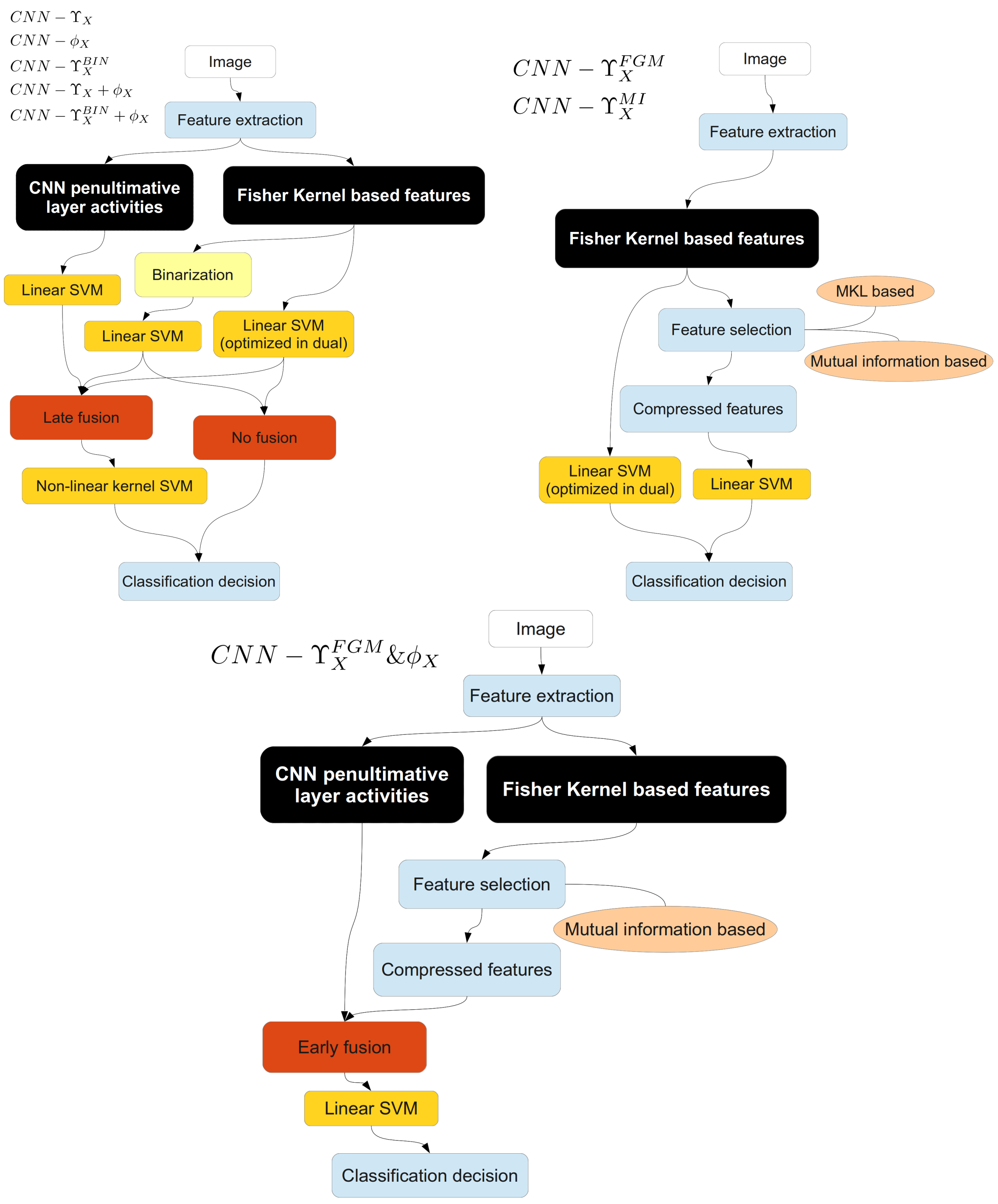


Figure 4.1. Diagrams of proposed image classification systems. The scheme of the early fusion and binarization pipeline from Section 4.1.1 is located in the upper left part. The upper right part shows pipelines CNN-Υ_X^{FGM} and CNN-Υ_X^{MI} described in Section 4.1.2. The layout of the CNN-$\Upsilon_X^{FGM}\&\phi_X$ early fusion pipeline (Section 4.1.3) is placed in the in the lower region.

At this point each of the 2000 bounding boxes is assigned K scores that define the probability of every class being present in the given bounding box. Thanks to the fact that the classifiers are learned using the logistic empirical loss function and the value of the C parameter is the same for each one-vs-rest SVM, their scores are roughly calibrated across classes. Denote $s_{B_i,k}^{CNN}$ the score that is assigned to the bounding box B_i by the SVM classifier that is learned to recognize class k. The property of calibrated scores enables the labeling of each bounding box by the the maximum score across all K classes. More precisely denote the final score of the bounding box B_i as $S_{B_i}^{CNN}$. It is then defined as:

$$S_{B_i}^{CNN} = \max_k s_{B_i,k}^{CNN} \tag{4.1}$$

Once each bounding box is labeled with $S_{B_i}^{CNN}$ it is possible to use these values to decrease the amount of the bounding boxes that have to be evaluated in the third stage of the cascade. The decrease in size is performed by simply sorting the set of bounding boxes in descending order of $S_{B_i}^{CNN}$ and retaining only top D detections[1].

Note that the labeling of each bounding box with a class according to the highest posterior probability (explained in the previous paragraph) could lead to a higher amount of false positive detections, due to class confusion errors (dog/cat, bus/car, etc.). This is not a problem because in the next stage this class label information is forgotten. The only purpose of the second stage is to decrease the amount of windows that need to be classified in the third stage as much as possible.

4.2.3. Third stage

The third stage operates solely on D windows that come from the second stage. Here each CNN descriptor corresponding to each of the top scoring D windows (note that these descriptors were computed during the second step of the cascade) is appended with the corresponding ℓ_2 normalized CNN-FK feature vector (early fusion approach). A second group of K linear SVM classifiers is then learned on top of these concatenated descriptors. The classifiers are then used to label every bounding box that made it to the third stage with K scores, that correspond to the probability of each class being present inside. Note that at this point, no max-pooling approach is employed and the K sets of scored detections are first separately filtered using another non-maxima suppression and evaluated according to the Pascal VOC 2007 detection challenge rules.

Figure 4.2 contains the diagram of the aforementioned detection pipeline architecture.

CNN-FK descriptor compression

Because detection pipelines are much more memory demanding than the classification ones, the CNN-FK descriptor that is appended to the CNN features at the third stage of the detection process has to be compressed. In the presented system the MKL feature selection is used. The ML-FGM algorithm was used, because it performed better on the classification task (as it could be observed from experiments in Section 5.3.4). Note that the set of dimensions selected by the MKL feature selection algorithm was obtained via the classification task. The reason why the feature selection algorithm was not run using the CNN-FK features coming from the detection task is, that obtaining sufficiently large set of positive and negative detection CNN-FK features for each class would be extremely memory and time consuming task. In our experiments the dimensionality was decreased by the factor of 1000.

[1] $D = 100$ for all detection experiments in this thesis

Non-maxima suppression method

The used non-maxima suppression method is the one that is most commonly employed in standard detection pipelines. It consists of sorting the set of detections by their score in descending order and then retaining only those windows that do not have their overlap with any of the higher scoring detections higher than 30% (intersection over union measure - see Appendix A.2.1 for further details).

4.2.4. Detection pipeline learning process

The two groups of linear SVM classifiers are learned using the following method. The first bunch of K linear SVMs is trained using the hard negative mining process first used in [11]. The method consists of alternating between adding "hard" negative samples to the set of training features and retraining the classifier on this augmented set. The "hard" samples are coming from the training set and are assigned high scores by the classifier while being labeled as negative. A bounding box is labeled as negative if its overlap with any of the ground truth objects in the image is lower than 30 %.

The second group of SVM classifiers that are trained to recognize the appended feature vectors in the third stage of the cascade is also trained using hard negative mining. The difference is that the "hard" samples are picked solely from the set of bounding boxes that "survive" to the third stage of the cascade. Because there is no need for calibrated SVM outputs, the classifiers are learned using the hinge empirical loss function.

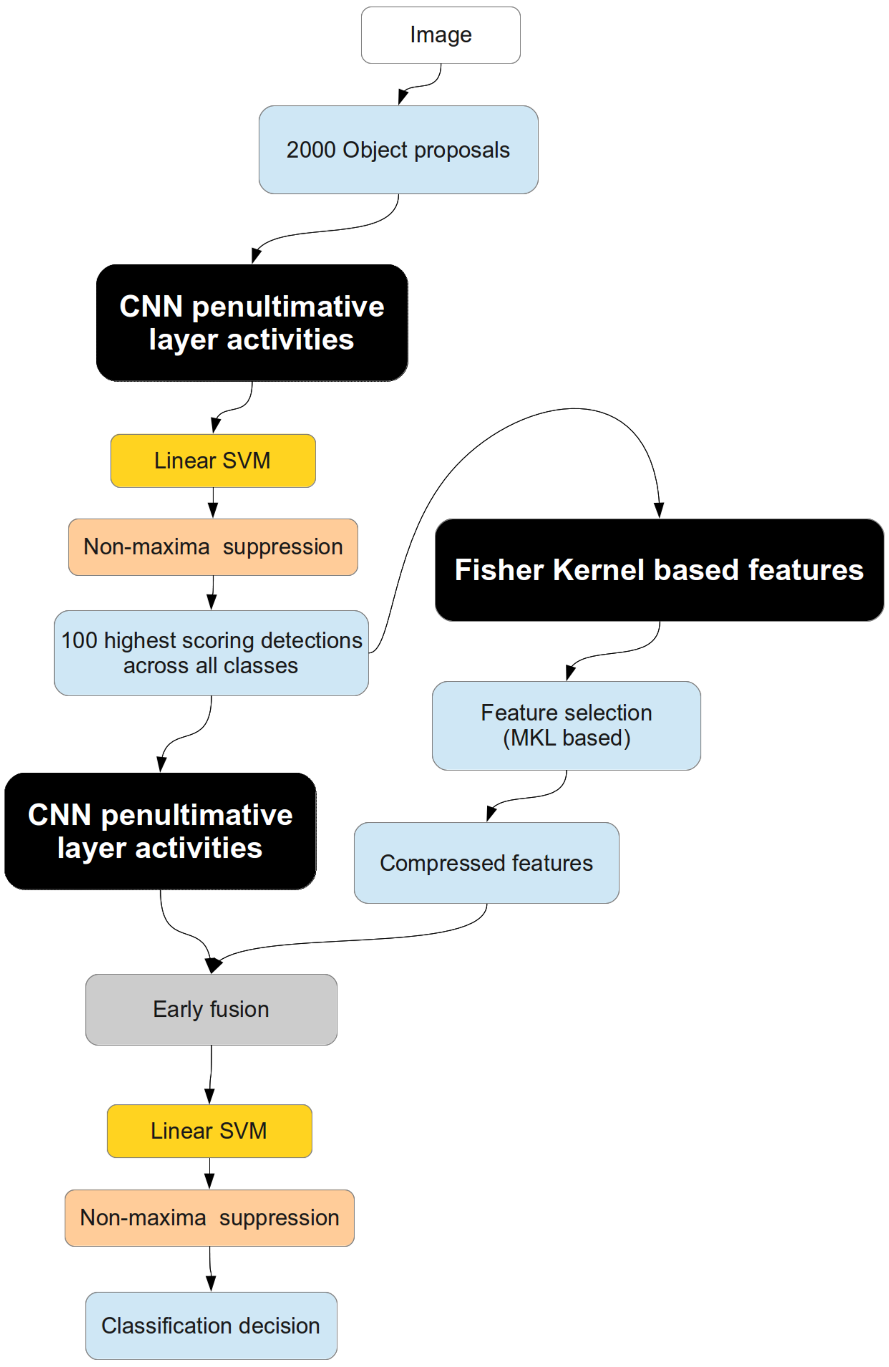


Image
2000 Object proposals
CNN penultimative layer activities
Linear SVM
Non-maxima suppression
100 highest scoring detections across all classes
Fisher Kernel based features
Feature selection (MKL based)
CNN penultimative layer activities
Compressed features
Early fusion
Linear SVM
Non-maxima suppression
Classification decision

Figure 4.2. Detection pipeline diagram explaining individual steps of scoring 2000 object proposals coming from each image.

5. Experiments

5.1. Datasets and evaluation protocol

All of the experiments in this thesis were concluded on the Pascal VOC 2007 dataset [15]. It is commonly used standard benchmark for image classification and object detection pipelines. It contains images with annotations of 20 object categories.

All of the SVMs were trained on the ”train” and ”val” sets and the performance on the ”test” set is reported. Hyperparameters (such as SVM's C regularization constant) were optimized using the ”val” set.

The reported average precision measures (AP) are the averages of the precision observed each time a new positive sample is recalled [39], i.e. these are not the TREC AP metrics that were originally used for the Pascal VOC 2007 evaluation. The standard AP metric was chosen, because these are the numbers that are currently being reported in the recently published research articles. The mean average precision (mAP) is the mean over average precisions of each of all the classes 20 classes in the VOC 2007 dataset.

For the description of the average precision evaluation process on the image classification task see Appendix A.1. Appendix A.2 then contains the description of reported object detection metrics.

5.2. Used convolutional nets

Instead of building and training a CNN architecture from scratch, the pretrained networks from [7] and [13] were used.

5.3. Image classification experiments

The following subsection describes the set of experiments concluded on the Pascal VOC 2007 image classification task.

5.3.1. Experiments with the Caffe CNN

The first batch of experiments was performed using the ImageNet reference CNN model from [13]. The performance of the five pipelines introduced in Section 3.4 was compared. For completeness the systems are CNN-ϕ_X (reimplementation of [13]), CNN-Υ_X, CNN-Υ_X^{BIN}, CNN-$\Upsilon_X + \phi_X$ and CNN-$\Upsilon_X^{BIN} + \phi_X$. The summary of the experiment is captured in Table 5.1.

From the results it is apparent that the proposed gradients of the pseudo-loglikelyhood contain relevant information and are suitable for use in image classifiers. Already the basic version of the CNN-FK classifier CNN-Υ_X gives results comparable to the state of the art. With 74.3 mAP It actually beats the best Fisher Kernel method which used GMM of SIFT patches as an underlying generative model (68.0 mAP [7]).

It is apparent that the binarization of the features (CNN-Υ_X^{BIN}) discards a lot of information and a performance drop of 2 mAP points over the CNN-Υ_X version which

Class	CNN-Υ_X^{BIN}	CNN-Υ_X	CNN-ϕ_X [13]	CNN-$\Upsilon_X^{BIN} + \phi_X$	CNN-$\Upsilon_X + \phi_X$
aero	87.7	90.3	**92.6**	91.9	92.5
bicycle	77.9	80.0	82.2	82	**82.5**
bird	83.4	84.5	86.6	86.6	**86.9**
boat	83.5	84.4	87.1	87.1	**87.5**
bottle	34.2	36.4	40.0	**41.1**	40.8
bus	70.6	73.0	74.2	74.5	**74.7**
car	86.2	87.1	88.0	88.0	**88.3**
cat	79.8	79.9	82.8	83.0	**83.2**
chair	59.6	59.3	59.5	60.7	**61.1**
cow	54.5	61.3	65.3	63.5	**65.4**
dtable	67.5	71.1	70.8	72.4	**72.9**
dog	74.9	77.9	79.5	79.5	**80.3**
horse	85.6	87.6	88.3	88.3	**89.1**
mbike	74.9	77.4	78.4	78.0	**78.8**
person	93.1	93.4	93.4	93.6	**93.8**
pplant	49.4	52.1	53.0	54.6	**55.0**
sheep	68.5	71.5	73.0	73.4	**74.1**
sofa	60.9	62.2	61.4	62.3	**63.4**
train	88.7	89.5	**91.9**	91.6	91.7
tv	64.9	66.8	**70.5**	70.3	70.1
mAP	72.3	74.3	75.9	75.6	**76.6**

Table 5.1. Comparison between CNN-FK classifier with binarized features (CNN-Υ_X^{BIN}), standard CNN-FK classifier CNN-Υ_X and the CNN-ϕ_X system from [13], which classifies the activities of the topmost fully connected layer. The classifiers formed by combining scores of CNN-Υ_X with CNN-ϕ_X (CNN-$\Upsilon_X+\phi_X$) and CNN-Υ_X^{BIN} with CNN-ϕ_X (CNN-$\Upsilon_X^{BIN}+\phi_X$) are also included.

does not use feature compression is observed. However the advantage of CNN-Υ_X^{BIN} is that the descriptors are 32 times smaller than in the case of CNN-Υ_X, thus a linear SVM classifier could be used. Another benefit of binarization is, that after the learning is finished the features can be scored using table lookups which speeds up the application of model at test time by the factor of eight. Needless to say, from the memory perspective, the decrease in size of the descriptors is also advantageous.

The CNN-Υ_X method actually does worse in comparison with the standard CNN-ϕ_X classifier - a performance drop of 1.6 mAP points can be observed, although not that severe, the proposed method alone does not seem to surpass the original CNN-ϕ_X pipeline.

However the results of combined classifiers actually show that the information contained in the gradients of the loglikelyhood is not redundant and seems to be useful for making classification decisions. The improvement of almost 0.7 mAP of the CNN-$\Upsilon_X+\phi_X$ classifier combination over CNN-ϕ_X supports this claim. The second combined classifier CNN-$\Upsilon_X^{BIN} + \phi_X$ does not produce improved results, which is expected concerning the inferior performance of CNN-Υ_X^{BIN}.

5.3.2. Classifier combination analysis

To show that the CNN-FK classifier contains useful information a series of plots in Figure 5.1 shows the features that enter the final combined classifier, i.e. the scores of CNN-Υ_X and CNN-ϕ_X that are assigned to individual examples of the test set are plotted.

As noted above each point in the Figure 5.1 stands for one test image. Its "x"

coordinate is the score assigned by the CNN-ϕ_X classifier trained for a particular class and "y" coordinate is the CNN-Υ_X's score. Each point is plotted either as a "+" sign (positive example according to the ground truth label) or a circle (negative example). The color of each point encodes the score of the final combined classifier CNN-$\Upsilon_X + \phi_X$.

After taking a look at the plots in Figure 5.1 it is possible to observe that the scores of the CNN-Υ_X and CNN-ϕ_X are very correlated. However mainly next to the combined classifier decision boundary it is possible to observe that there are positive samples that receive higher score from one classifier, while being ranked lower by the second one. This inconsistency between the scores seems to help the final combined classifier to improve the final decision by giving higher weight to the classifier that assigns higher score to a given positive sample. It is interesting to observe that although the final classifier is trained using polynomial kernel the decision boundary has the shape of a (almost) straight line.

5.3.3. Experiments with the state of the art CNN

The next round of experiments was concluded using the CNN-S network from [7]. This network achieved currently highest reported mAP on the Pascal VOC 2007 image classification challenge.

Note that the network CNN-S TUNE-RNK, which is a fine-tuned version of CNN-S trained on an augmented training set was not publicly available at the time this thesis was being written, thus the experiments use the "vanilla" non-finetuned network CNN-S. Also in this thesis the training set augmentation from [7] was not implemented, which is the reason why the reimplementation of the classification pipeline from [7] does not produce the same performance as reported in [7].

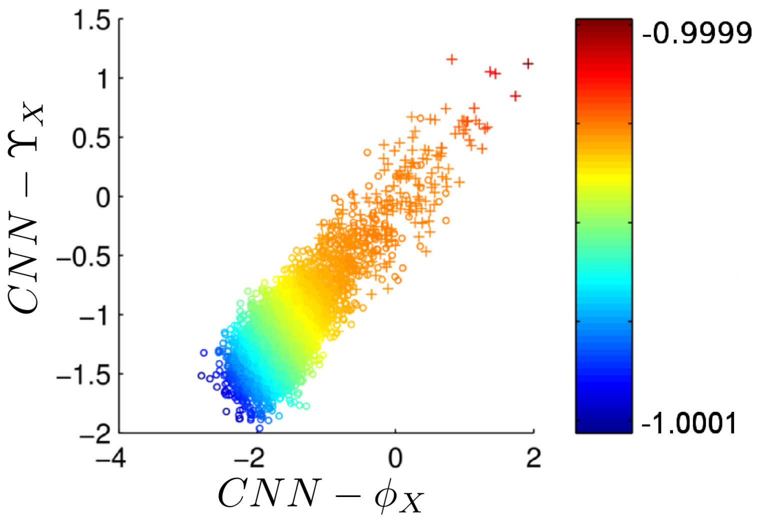
Apart from the classic CNN-S architecture [7] denoted as CNN-S-ϕ_X[1] in this thesis, the classifier with Fisher Kernel based features derived from CNN-S was tested (CNN-S-Υ_X) together with the late fusion classifier that combines the scores of CNN-S-ϕ_X and CNN-S-Υ_X using a nonlinear kernel SVM (polynomial, rbf and tanh kernels were tried). The combined classifier is named CNN-S-$\Upsilon_X + \phi_X$ The results of these experiments are reported in Table 5.2.

The first thing that is apparent is that the CNN-S-ϕ does perform worse than is reported in [7]. As noted above, the training set augmentation is not implemented in this thesis which is the most likely reason why the CNN-S-ϕ_X pipeline in this thesis produces results inferior to the one from [7] - the difference is 0.84 mAP points.

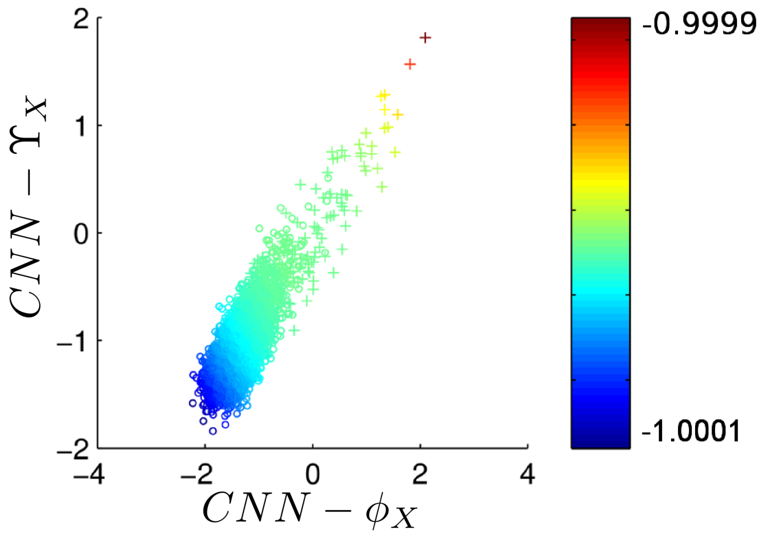
The comparison between CNN-S-ϕ_X and CNN-S-Υ_X is again in favor of CNN-S-ϕ_X. However the combined classifier CNN-S-$\phi_X + \Upsilon_X$ beats the CNN-S-ϕ_X by 0.7 mAP points, proving again that by utilizing the Fisher Kernel based classifier scores the performance of the final classification system improves.

The conclusion from the experiments in Sections 5.3.1 and 5.3.3 is that it seems very likely that if the CNN-S TUNE-RNK network from [7] was available the similar performance increase of ~ 1 mAP point, when using the Fisher Kernel based features in combination with the neuron activities would very likely occur again. However since these experiments were not concluded, the only verifiable result is that the network CNN-S-$\phi_X + \Upsilon_X$ achieves state of the art performance on the Pascal VOC 2007 classification challenge when fine tuning is not employed and if the training set augmentation is not used.

[1] Note the extra "S" letter appearing in the abbreviation. It emphasizes the fact that the CNN-S network is used instead of the Caffe reference ImageNet model, which was utilized in the previous experiments.



(a)



(b)

Figure 5.1. Visualization of scores outputted by the CNN-Υ_X classifier and the CNN-ϕ_X classifier. The score of the combined classifier CNN-$\Upsilon_X + \phi_X$ is encoded in color. Every point in the graph is one testing instance (i.e. image from a test set of Pascal VOC 2007 image classification challenge). The positive instances are plotted as "+" signs and negative ones as circles. The top plot (a) shows the situation for the classifiers learned for the "diningtable" class the plot (b) shows the "pottedplant" class.

Class	CNN-S-ϕ_X [7]	CNN-S-Υ_X	CNN-S-Υ_X+ϕ_X		
			poly	rbf	tanh
aero	92.3	87.1	92.5	**92.6**	**92.6**
bicycle	86.1	84.7	86.2	**86.3**	**86.3**
bird	88.3	87.5	88.8	**88.9**	**88.9**
boat	**88.5**	84.7	88.1	88.3	88.3
bottle	42.5	41.3	42.7	**43.7**	**43.7**
bus	**78.9**	76.2	78.6	78.7	78.7
car	89.7	88.7	**89.9**	**89.9**	**89.9**
cat	88.5	86.7	88.8	**88.9**	**88.9**
chair	62.6	63.4	63.5	**64.0**	**64.0**
dtable	71.6	72.9	73.7	**74.6**	**74.6**
cow	67.9	65.8	**68.1**	**68.1**	**68.1**
dog	85.1	83.7	85.3	**85.8**	**85.8**
horse	89.4	88.5	89.9	**90.2**	**90.2**
mbike	**82.6**	80.0	82.3	82.5	82.5
person	93.8	94.2	94.2	**94.4**	94.3
pplant	54.7	54.9	56.5	**56.8**	**56.8**
sheep	79.2	77.4	79.6	**80.0**	**80.0**
sofa	68.5	66.3	68.9	**69.2**	**69.2**
train	93.5	92.5	**94.0**	**94.0**	**94.0**
tv	74.0	71.4	**74.7**	**74.7**	**74.7**
mAP	78.9	77.4	79.3	**79.6**	**79.6**

Table 5.2. Comparison between the CNN-S-ϕ_X and CNN-S-Υ_X classifiers, which is a Fisher Kernel based classifier derived from the CNN-S network. The classifier that combines scores of CNN-S-ϕ_X and CNN-S-Υ_X is denoted CNN-S-$\phi_X + \Upsilon_X$ and its results are also in the table. Three nonlinear kernels (poly, rbf, tanh) were compared in the case of CNN-S-$\phi_X + \Upsilon_X$.

5.3.4. Feature selection experiments

This subsection contains the results of the experiments that were comparing the performance of the both feature selection techniques proposed in Section 3.3, i.e. the *MKL based supervised feature selection* (ML-FGM) and the *Mutual information based feature selection* (MI-FS).

The experiments were again concluded using the state of the art CNN model from [7] - CNN-S. This time the pipeline that uses solely Fisher Kernel based features was tested (i.e. no combined classifier was employed).

To compare the quality of selected features the performance of the pipeline that uses the Υ_X compressed by both of the methods is evaluated. For each method four feature selection experiments were concluded, such that every time the dimensionality of the features is reduced by the factor of 10^1, 10^2, 10^3 and 10^4 (the original dimension of the Υ_X features is $\sim 103 \times 10^6$). After performing the dimensionality reduction the compressed features were fed to the SVM solver. The results of the aforementioned experiments are presented in Table 5.3.

The first apparent conclusion from this set of experiments is that the mutual information based feature selection approach performs much worse than the multiple kernel learning method. This observation is quite expected, since the mutual information based approach does not take into account correlations between individual features and treats them independently. Also the MKL based method optimizes a objective function which is very close to the one that is used in the original SVM learning algorithm, thus giving the final SVM classifier a set of features that are tailored for the problem which is solved.

Class	CNN-S-ϕ_X	CNN-S-Υ_X^{FGM} dimensionality decrease factor								
		1	10^4		10^3		10^2		10	
		no compr.	MI	MKL	MI	MKL	MI	MKL	MI	MKL
aero	92.3	87.1	68.5	89.1	90.9	91.1	91.8	91.9	**92.8**	92.6
bicycle	86.1	84.7	15.4	80.6	73.0	83.5	84.4	85.1	**86.2**	86.1
bird	88.3	87.5	70.4	86.4	86.5	87.4	88.0	88.1	**89.1**	88.6
boat	88.5	84.7	58.3	82.4	84.7	85.8	88.0	87.7	**89.0**	88.4
bottle	42.5	41.3	3.8	38.8	38.3	42.3	41.4	44.6	43.3	**45.2**
bus	78.9	76.2	62.0	72.9	71.2	76.5	79.1	78.8	**80.0**	79.7
car	89.7	88.7	80.1	87.4	85.5	89.2	89.2	89.7	**90.2**	**90.2**
cat	**88.5**	86.7	53.2	84.8	83.1	87.7	87.7	87.9	88.3	88.3
chair	62.6	63.4	37.6	59.4	51.4	62.2	60.6	62.6	63.1	**63.9**
cow	71.6	**72.9**	13.7	57.2	54.7	65.0	67.4	67.2	69.1	68.7
dtable	67.9	65.8	5.0	68.9	56.7	73.8	68.9	74.9	73.5	**75.5**
dog	85.1	83.7	17.1	81.4	74.9	84.2	83.1	85.1	**85.9**	85.8
horse	89.4	88.5	42.7	85.5	83.2	88.6	88.4	89.6	**90.4**	90.1
mbike	82.6	80.0	53.7	76.1	73.7	81.0	82.6	82.7	83.2	**83.3**
person	93.8	94.2	74.3	92.9	90.0	93.9	93.4	94.1	**94.4**	**94.4**
pplant	54.7	54.9	15.5	47.8	35.7	53.1	52.9	54.9	56.2	**56.8**
sheep	79.2	77.4	20.7	73.4	69.3	77.9	77.5	78.8	**79.8**	79.6
sofa	68.5	66.3	5.0	64.2	53.9	68.4	64.6	69.0	69.3	**70.1**
train	93.5	92.5	66.7	91.0	88.7	92.7	93.0	93.2	**93.6**	**93.6**
tv	74.0	71.4	53.3	71.0	59.7	74.8	73.1	**75.7**	74.9	75.3
mAP	78.9	77.4	40.9	74.6	70.3	78.0	77.8	79.1	79.6	**79.8**

Table 5.3. The results of the comparison between the ML-FGM and MI-FS feature selection methods.

One very interesting observation is the fact, that MKL based feature selection actually improves the performance of the CNN-Υ_X by a substantial amount of 2.4 mAP points (CNN-S-Υ_X with no compression vs. CNN-S-Υ_X^{FGM} with 10 times compressed features). This could be the result of removing noisy features from the training set. Note that the result of **79.8 mAP** points is actually better than the performance the original CNN-ϕ_X network, which uses neuron activations as features.

The conclusion of the feature selection experiments is that the MKL based feature selection method gives surprisingly good results. From Figure 5.2 it is possible to see that the **dimensionality of the Fisher Kernel based features Υ_X could be decreased by the factor of 10^3 while obtaining performance superior to the pipeline that uses uncompressed Υ_X features**. Also when the dimensionality of the Υ_X features is decreased 10 times, **the performance of the CNN-S-Υ_X^{FGM} pipeline is actually superior to the original CNN-S-ϕ_X, which uses neuron activities as image features, with the difference of almost 1mAP point**.

Late fusion with MKL compressed features

The observation from the previous section motivated the experiment where the classifier scores of the Υ_X^{FGM} features 10 times compressed using ML-FGM algorithm are used in combination with the scores outputted by the CNN-S-ϕ_X classifier. Similar to Section 5.3.3 the scores were combined using the non-linear polynomial kernel.

The final result was **79.8** mAP which is slightly better than the CNN-S-$\phi_X + \Upsilon_X$ classifier's 79.6 mAP. However the performance is the same as the best result from the previous section (10 times compressed Υ_X^{FGM} features using ML-FGM).

The intuition that the improved CNN-S-Υ_X^{FGM} classifier would also improve the results of the combined classifier is thus not confirmed by this experiment.

Analysis of selected features

Because each dimension of a Fisher Kernel based feature vector corresponds to a derivative of a parameter coming from a particular layer of the CNN architecture, it is interesting to analyze from which layers the selected features come from.

The CNN-S network consists of 5 convolutional layers that are denoted conv1, ... conv5, three fully connected layers above them fc6, ..., fc8 and one layer on the very top that outputs the value corresponding to the pseudo-loglikelyhood evaluated at given input image X. All these layers contain parameters, who's derivatives evaluated at point X form the final Fisher Kernel based feature vector. The series of pie charts in Figure 5.3 and Figure 5.4 depicts how many features were selected by ML-FGM and MI-FS from each layer for different settings of the dimensionality decrease factor. Note that because each layer contains different amount of parameters the number of selected features is always normalized per layer by the total number of parameters in that particular layer.

The charts show that the pseudo-loglikelyhood layer seems to be the most important one. This is quite expected, because the topmost layer typically contains the most abstract information that is the most suitable for making final classification decision. It is interesting that the lower fully connected layers are not as important as the topmost one. Also there is a not negligible portion of derivatives with respect to the parameters of the convolutional layers present in the set of selected features. This seems unexpected, because the lower convolutional layers typically contain simple gabor-like filters [23] which do not carry much information about the complex structure of object instances that are being detected by the pipeline.

The comparison between the sets of selected features by MI-FS and ML-FGM shows that MI-FS typically goes for all the features in the topmost layer, which carry the most complex information. However because MI-FS neglects the dependencies between features and treats each feature dimension independently, the lower layers that typically do not contain enough information for making classification decision are not selected by MI-FS. This seems like the main reason why MI-FS method is so inferior to ML-FGM, since smaller perturbations in the lower layers in combination with the higher level semantic information from the top CNN layer seems to improve the resulting classifier performance.

The important thing to mention here is that the experiment in this section assumes that the number of selected features that come from a given layer is proportional to the importance of the derivatives of the parameters located in that layer. This does not have to be necessarily true for single dimensional features that contain a lot of information by themselves and their sole values are sufficient to make complex decisions, thus their amount does not say anything about their importance.

False positive / true positive images

Figure 5.5 contains the set of some highest ranked false positive images. Figure 5.6 on the other hand contains some examples of the highest scoring true positive images. The classification pipeline that was used to output these examples was the CNN-S-Υ_X^{FGM} classifier with the dimension of the feature vectors decreased by the factor of 10 using the MKL feature selection method.

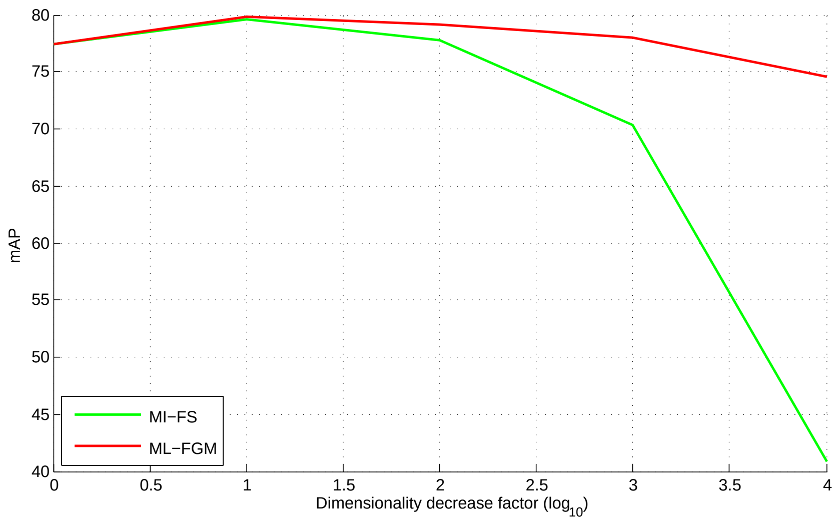


Figure 5.2. The plot that shows the performance of the ML-FGM and MI-FS feature selection methods as a function of the dimensionality decrease factor. Note the logarithmic scale of the "x" axis.

5.3.5. Early fusion experiments

Another set of experiments was evaluating the performance of the pipeline that classifies the concatenated ϕ_X and Υ_X descriptors (i.e. the *early fusion* combination explained in Section 4.1.3). Furthermore the size of Υ_X descriptors derived from CNN-S was again decreased by utilizing the better of the two feature selection methods according to the experiments concluded in the previous section (i.e. using the MKL based feature selection). The summary of the experiment is in Table 5.4.

The classifier which learns the concatenations of the CNN-S neuron activities with the Fisher Kernel based features extracted from the CNN-S network is termed CNN-S-Υ_X^{FGM}&ϕ_X.

After taking a look at the results, it is obvious that the early fusion also improves over the classic CNN-S-ϕ_X network, however the performance is inferior to the CNN-FK classifier that uses 10 times compressed Fisher Kernel based features. The best result of **79.7 mAP** is actually obtained once the CNN-FK features that are appended to the CNN-S neuron activations are not compressed at all.

5.3.6. Classification pipeline run-time analysis

The following section discusses the speed of the best performing image classification pipeline which is CNN-S-Υ_X^{FGM} with dimensionality reduction factor equal to 10 (79.8 mAP).

The classification times were measured on a cluster node equipped with Intel Xeon E5-2630 CPU. Note that although it is a common practice, we did not use a GPU for obtaining the CNN neuron activities or FK based features.

Mean image processing time measured on a subset of test images is 1.7 sec. This includes obtaining the CNN-FK features together with their scoring using the K sets of learned SVM weights. The whole time is in fact spent on extracting the image signatures. The overall SVM scoring time is below 0.1 seconds. Note that the feature extraction time is independent of the number of classes K, thus it is expected that the method scales very well with increasing K. More exactly the SVM scoring time has complexity of $O(KD)$, where D stands for the descriptor dimension. The time that is

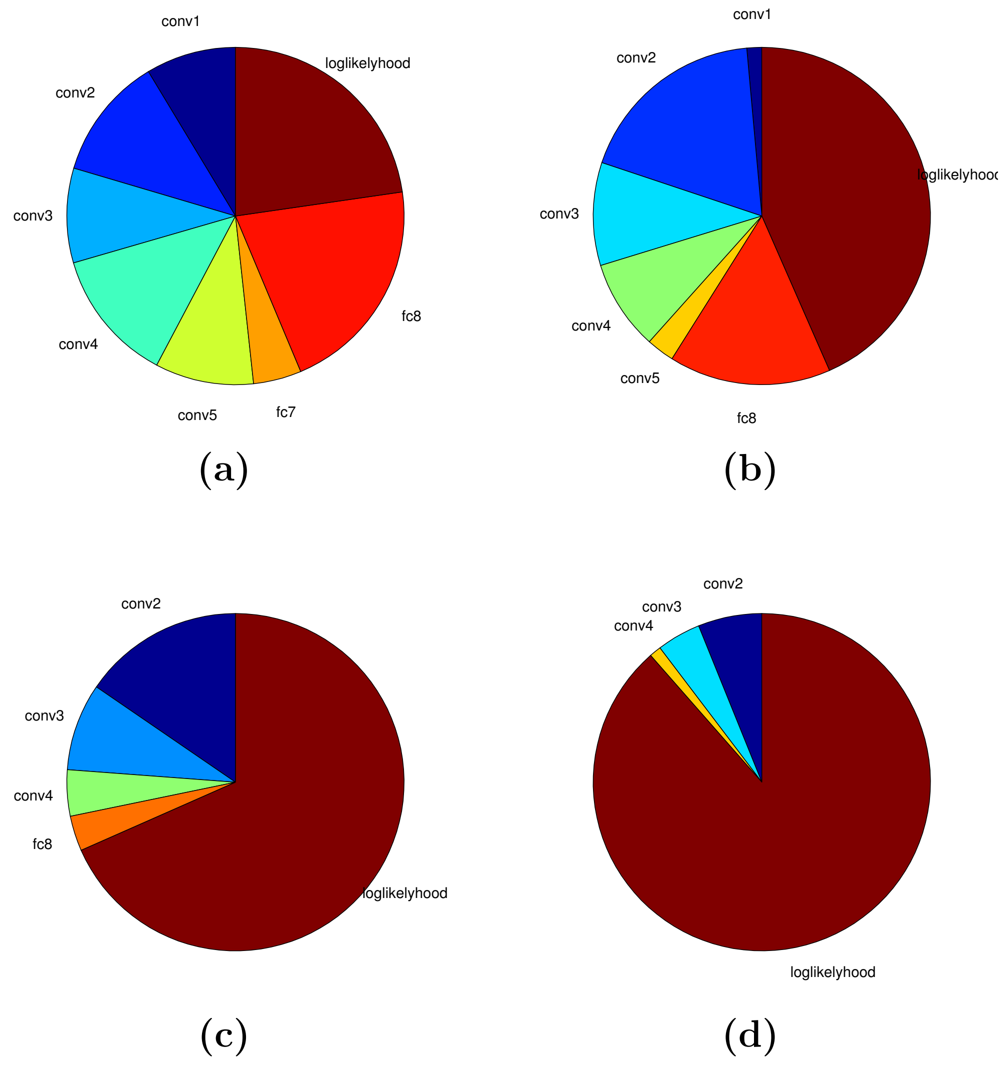


Figure 5.3. *Features selected by ML-FGM.* Pie charts showing the distribution of selected features across the individual layers of the CNN-S network. Charts (a), (b), (c), (d) correspond to the setting with the dimensionality decrease factor equal to 10, 100, 1000 and 10 000 respectively. The features were selected by the **ML-FGM algorithm**. Note that the layers who's portion was below 1% were not included in the charts.

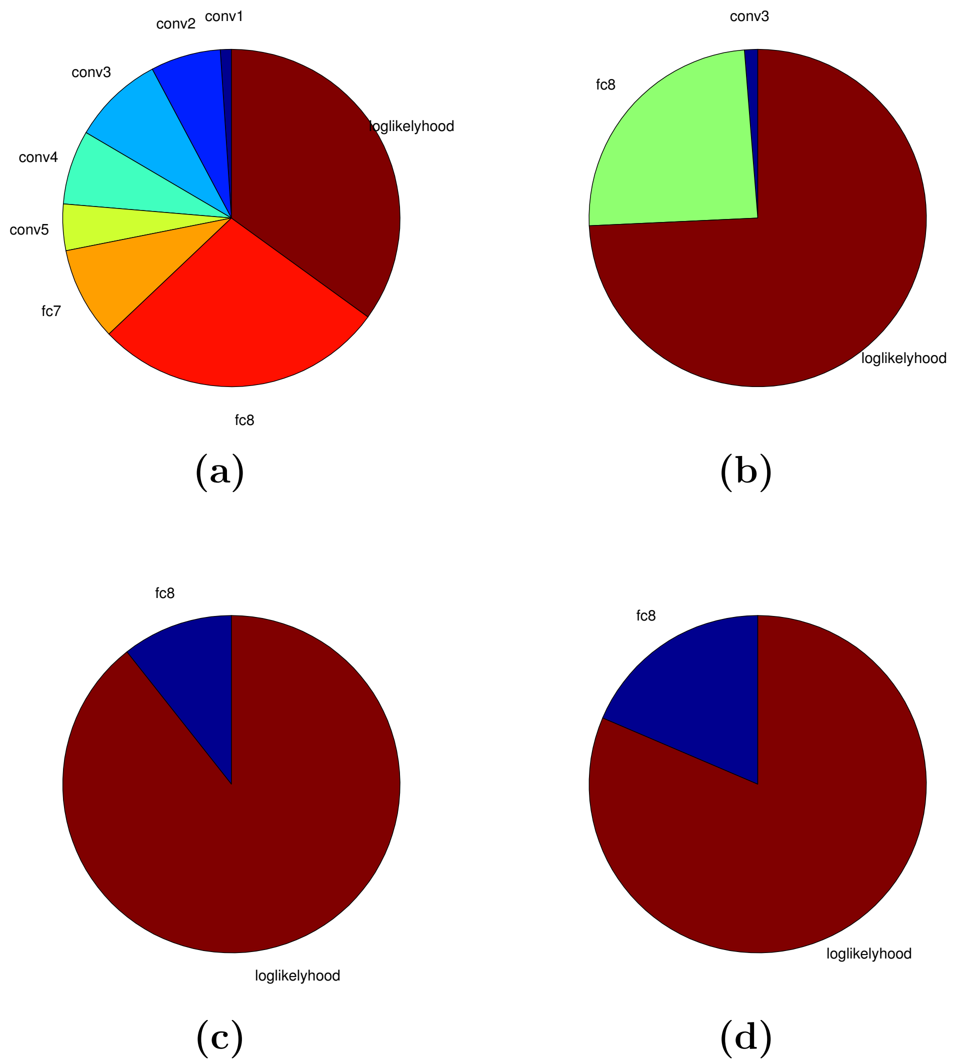


Figure 5.4. *Features selected by MI-FS.* Pie charts showing the distribution of selected features across the individual layers of the CNN-S network. Charts (a), (b), (c), (d) correspond to the setting with the dimensionality decrease factor equal to 10, 100, 1000 and 10 000 respectively. The features were selected by the **MI-FS algorithm**. Note that the layers who's portion was below 1% were not included in the charts.

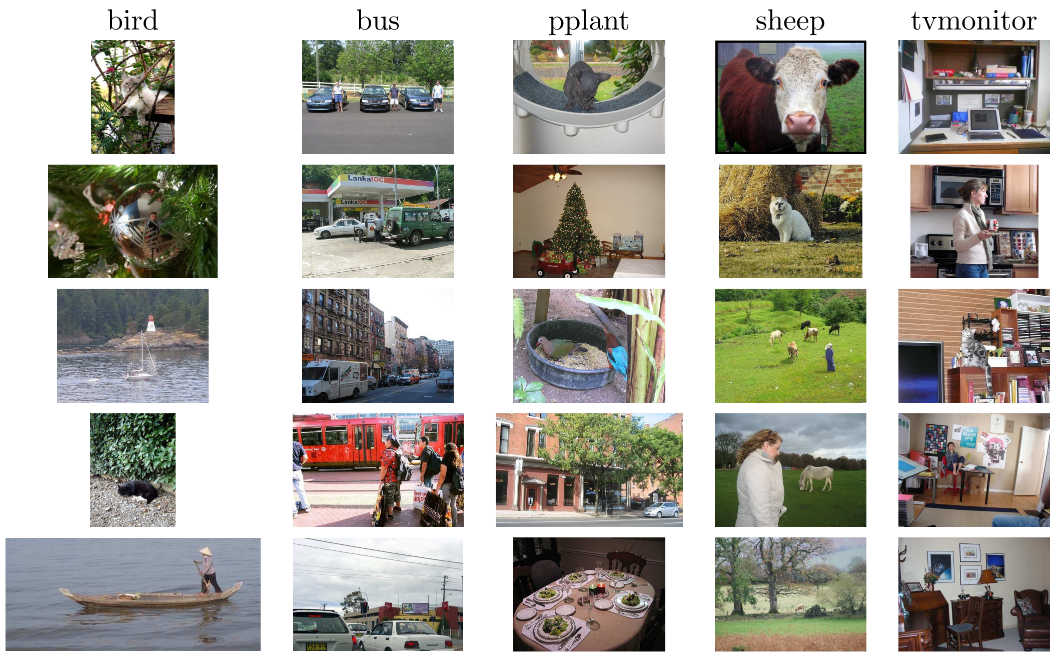


Figure 5.5. Images of highest ranked **false positive** images for some classes from the Pascal VOC 2007 classification challenge test set. The images were scored using the CNN-S-Υ_X^{FGM} classifier with the features compressed by the factor of 10 using the MKL based feature selection method.

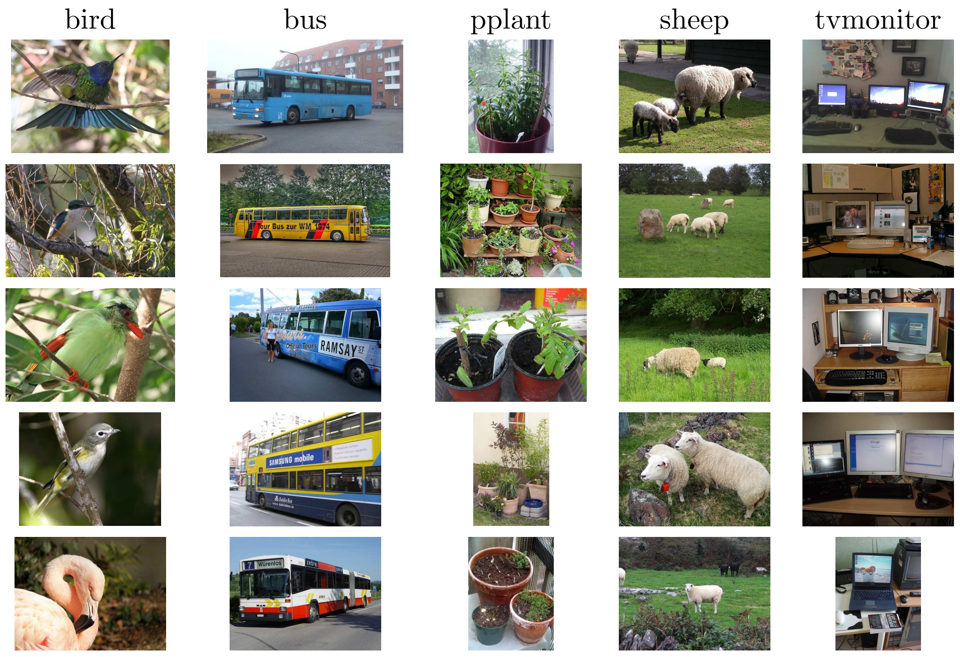


Figure 5.6. Images of highest ranked **true positive** images for some classes from the Pascal VOC 2007 classification challenge test set. The images were scored using the CNN-S-Υ_X^{FGM} classifier with the features compressed by the factor of 10 using the MKL based feature selection method.

Class	CNN-S-Υ_X^{FGM}&ϕ_X				
	Dimensionality reduction				
	1 (no compression)	10^1	10^2	10^3	10^4
aero	**92.5**	**92.5**	92.1	92.0	92.2
bicycle	**86.5**	86.2	85.5	85.7	85.9
bird	**88.8**	88.7	88.2	88.1	88.4
boat	**88.9**	88.8	87.6	87.7	88.5
bottle	**44.1**	**44.1**	41.8	42.4	43.7
bus	**79.6**	**79.6**	78.3	78.8	79.4
car	89.9	**90.1**	89.7	89.9	89.9
cat	**88.8**	88.6	88.1	88.6	88.5
chair	**63.5**	63.4	63.1	63.1	63.0
cow	68.1	**68.6**	65.4	67.1	67.7
dtable	**74.9**	74.6	74.2	74.8	74.6
dog	**85.6**	**85.6**	85.1	85.0	85.3
horse	**90.1**	90.0	89.2	89.7	89.8
mbike	83.1	**83.3**	82.5	82.9	83.0
person	94.1	**94.2**	93.9	94.1	94.1
pplant	**57.2**	56.4	54.5	55.1	55.4
sheep	**80.0**	79.4	78.0	79.0	79.1
sofa	**69.2**	**69.2**	69.0	69.1	68.8
train	**93.9**	93.7	93.8	93.6	93.5
tv	75.3	75.0	**75.5**	75.1	75.3
mAP	**79.7**	79.6	79.3	79.1	78.8

Table 5.4. The results of the early fusion experiments on the Pascal VOC 2007 image classification challenge. See text for the description of individual classifier settings.

needed to score one class (i.e. performing the dot product between Υ_X^{FGM} and vector of weights w_k) is ~ 0.004 sec, thus it should be possible to apply 250 different linear SVM models per second.

Note that the decrease of D (which could be achieved by setting different dimensionality reduction factors) is accompanied with the proportional reduction of SVM scoring times. Thus the time that the SVM part takes could be completely neglected once the dimensionality decrease factor is set to the value of 100 and more, while decreasing the performance by only a small amount.

The comparison of the speed of $CNN-\phi_X$ and $CNN-\phi_X^{FGM}$ is biased in favor of the classic CNN architecture $CNN - \phi_X$. The standard feed-forward pass takes around 0.15 seconds per image which is roughly 10 times faster than the extraction of the pseudo-loglikelyhood gradients. The SVM scoring part is very fast and in comparison with the CNN processing time is again negligible.

5.4. Object detection experiments

Since the experiments in the previous section have shown how to compress the CNN-FK features such that they form a very compact codes with their dimensionality significantly reduced, it is further possible to use them for the object detection task of Pascal VOC 2007.

The detection pipeline is described in detail in Section 4.2. Again in this experiment, the network that was used was CNN-S. The Υ_X features were compressed by the MKL based feature selection method to decrease their dimension by the factor of one thousand[2]. The final dimension of the CNN-S descriptor ϕ_X concatenated with the compressed Υ_X^{FGM} vector is $\sim 10^5$.

5.4.1. Reference detection system

The detection system that uses solely the CNN-S network features without appending the Fisher Kernel based vectors is denoted DET-CNN-S-ϕ_X. Note that this network is basically equivalent to the state of the art detection method of Girschick et.al. [19]. The difference is in the used CNN. Here the architecture has more parameters, thus should produce better results.

Another important remark here is that the CNN-S network is trained on the ImageNet classification task and no fine-tuning is employed. Thus the reader of this thesis should compare all the results with the mAP achieved by the non-finetuned network in [19]. This means that **the performance of the baseline detection method [19] is 46.2 mAP**.

Also because in [19] it has been shown that the features coming from the first (instead of last) fully connected layer actually give much better performance, the approach is adopted in this thesis and as the CNN-S features ϕ_X, the neuron activations from the *first* (instead of last) fully connected layer are used.

5.4.2. DET-CNN-S-ϕ_X vs DET-CNN-S-$\Upsilon_X^{FGM}\&\phi_X$

The detection pipeline proposed in this thesis (and explained in Section 4.2) is termed with abbreviation DET-CNN-S-$\Upsilon_X^{FGM}\&\phi_X$. The reimplementation of [19] with different CNN architecture (CNN-S from [7]) is named DET-CNN-S-ϕ_X. Table 5.5 compares the results of these two pipelines. Also the results of the reference method [19] are included.

The results show that in terms of mAP the DET-CNN-S-ϕ_X pipeline wins over the proposed DET-CNN-S-$\Upsilon_X^{FGM}\&\phi_X$ by a small amount of 0.1 mAP. However the proposed pipeline is able to surpass DET-CNN-S-ϕ_X by more than 2 AP points on some classes (tv, motorbike, sheep and cat). R-CNN is slightly inferior due to the lower-capacity convolutional neural network used in their experiments.

The reason why the appended Υ_X^{FGM} features do not improve the results by the same amount as it happened in the image classification task could be the fact that the set of selected features is optimized for the image classification task.

5.4.3. Detection pipeline run-time analysis

As it happens in the case of the proposed classification pipeline, the detection system's speeds scales well with increasing number of classes K. This is mainly due to the fact

[2] Recall that the set of selected features is optimized on the classification task.

Class	R-CNN [19]	DET-CNN-S-ϕ_X	DET-CNN-S-Υ_X^{FGM}&ϕ_X
aero	59.3	**61.5**	56.9
bicycle	61.8	**67.7**	66.8
bird	**43.1**	40.7	39.7
boat	34.0	**36.4**	35.7
bottle	**25.1**	20.9	21.5
bus	53.1	**61.3**	58.7
car	60.6	64.1	**64.5**
cat	52.8	59.7	**61.4**
chair	21.7	23.8	**24.6**
cow	47.8	**55.2**	52.4
dtable	**42.7**	41.9	39.6
dog	47.8	**54.3**	54.2
horse	52.5	58.7	**59.5**
mbike	58.5	62.3	**65.5**
person	44.6	**47.7**	46.9
pplant	**25.6**	19.9	21.6
sheep	48.3	49.8	**53.4**
sofa	34.0	**38.9**	38.1
train	**53.1**	52.4	49.7
tv	58.0	59.9	**62.8**
mAP	46.2	**48.8**	48.7

Table 5.5. The performance of the proposed object detection pipeline concluded on the Pascal VOC 2007 object detection challenge.

that there is always a fixed amount of descriptors that have to be extracted from each image (2000 ϕ_X and 100 Υ_X features).

The mean processing time per image for the proposed pipeline DET-CNN-S-Υ_X^{FGM}&ϕ_X is 378.8 sec on CPU. The state of the art architecture DET-CNN-S-ϕ_X consumes 218.4 sec per image in average. The SVM scoring and non-maxima suppression times are again negligible. All the time is spent on extracting CNN and CNN-FK features and obtaining a set of tentative bounding boxes ($\sim$ 20 seconds per image).

The bounding box extraction phase is relatively slow, however there have already been proposed methods that surpass the used selective search regions from [38]. For instance by employing [8] the set of 2000 tentative bounding boxes that have the same quality as [38] could be obtained in 0.01 seconds per image.

Note that 5 minutes per image could seem like a very high number, however we note that in the case of detection systems, it is a very common practice to move all the computations on the GPU, which typically speeds up the extraction times by the factor of $\sim$ 10. Another important thing to mention here is that recently a new method [20] that speeds up the extraction of the CNN features from a set of tentative objects in an image by a large factor of 64 was proposed. If the similar improvement was done for extracting the CNN-FK features, the extraction times would be around 5 sec per image on CPU.

6. Conclusion

In this thesis a novel method for extracting Fisher Kernel based statistics from convolutional neural networks was presented. It has been exhaustively tested on the Pascal VOC 2007 image classification and also object detection challenge with positive results.

We have shown that an image classification pipeline that is built on top of the Fisher Kernel based feature vectors is able to produce results comparable with current state of the art methods. It is further shown that the proposed approach improves the performance of the standard CNN image classification architecture. This is achieved by utilizing a classifier that combines the scores outputted by the pipelines that use Fisher Kernel based features and CNN neuron activations of the last fully connected layer.

The clear downside of the proposed FK based features is their high dimensionality. Several compression techniques were tested as a potential solution to this problem. In this thesis we evaluated the performance of three different methods for decreasing the descriptor size - mutual information based measures, binarization and Sparse SVM learning algorithm (SSVM).

The SSVM turned out to be the best performer. The used SSVM solver is an improved version of the MKL based feature generation machine [37]. The proposed improvement consists of enabling the use of the original algorithm in the multi-label classification tasks such as image classification. A surprising result was that the FK based features compressed using this feature selection algorithm surpass even the standard CNN neuron activations coming from the last fully connected layer. Furthermore it has been shown that it is possible to decrease the descriptor dimensionality up to the factor of 10^3 without getting below the performance of the uncompressed features.

The MKL based feature selection algorithm also gave insights on which dimensions of the FK based feature vectors are important for making classification decisions and visualizations of the distributions of the most discriminative features were presented.

Several other classification pipeline architectures were tested, including early and late descriptor fusion for many different settings of the dimensionality decrease factor.

The last contribution was the evaluation of performance of an object detection pipeline that uses the newly proposed features. It has been shown that the proposed features improve the performance of the detector on some of the classes. In general the proposed detector performs on par with the current state of the art architecture.

6.1. Future work

An interesting topic for the further research would be to try the CNN-FK features derived from the CNN that is used in [7] and gives impressive result of 82.4 mAP. This is currently the highest reported result of Pascal VOC 2007 image classification task. We have already demonstrated that the CNN-FK features produce performance superior to the CNN neuron activities for two strong performing network architectures, thus it would be convenient to test our method on the currently best performing system, which unfortunately is not publicly available.

It would also be worth to explore the use of more aggressive descriptor compression techniques that would decrease still relatively high memory footprint of the used CNN-FK features. The supervised dimensionality reduction method from [36] was already tested, however it seems that Pascal VOC 2007 is too small dataset since the method overfits on the training set.

Bibliography

[1] Bogdan Alexe, Thomas Deselaers, and Vittorio Ferrari. What is an object? In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 73–80. IEEE, 2010. 9

[2] Artem Babenko, Anton Slesarev, Alexander Chigorin, and Victor S. Lempitsky. Neural codes for image retrieval. *CoRR*, abs/1404.1777, 2014. 21

[3] Yoshua Bengio. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009. 9

[4] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *Neural Networks, IEEE Transactions on*, 5(2):157–166, 1994. 9

[5] Christopher M Bishop et al. *Pattern recognition and machine learning*, volume 1. springer New York, 2006. 13

[6] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152. ACM, 1992. 8, 18

[7] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the Devil in the Details: Delving Deep into Convolutional Nets. *ArXiv e-prints*, May 2014. 8, 10, 21, 29, 31, 33, 41, 43

[8] Ming-Ming Cheng, Ziming Zhang, Wen-Yan Lin, and Philip Torr. Bing: Binarized normed gradients for objectness estimation at 300fps. In *IEEE CVPR*, 2014. 9, 42

[9] Ramazan Gokberk Cinbis, Jakob Verbeek, and Cordelia Schmid. Segmentation driven object detection with fisher vectors. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 2968–2975. IEEE, 2013. 5, 9

[10] Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, volume 1, pages 1–2, 2004. 5, 8, 9

[11] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005. 8, 9, 27

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009. 5, 7, 8, 9, 10

[13] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. *arXiv preprint arXiv:1310.1531*, 2013. 8, 21, 23, 29, 30

[14] Bradley Efron. Bootstrap methods: another look at the jackknife. *The annals of Statistics*, pages 1–26, 1979. 9

[15] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html. 5, 7, 29, 49

[16] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Imagse Understanding*, 106(1):59–70, 2007. 5, 7

[17] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(9):1627–1645, 2010. 8

[18] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4):193–202, 1980. 5, 9

[19] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *arXiv preprint arXiv:1311.2524*, 2013. 9, 21, 24, 41, 42

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *CoRR*, abs/1406.4729, 2014. 9, 42

[21] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012. 9

[22] Tommi Jaakkola, David Haussler, et al. Exploiting generative models in discriminative classifiers. *Advances in neural information processing systems*, pages 487–493, 1999. 8, 10

[23] Anil K Jain and Farshid Farrokhnia. Unsupervised texture segmentation using gabor filters. In *Systems, Man and Cybernetics, 1990. Conference Proceedings., IEEE International Conference on*, pages 14–19. IEEE, 1990. 35

[24] Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. Aggregating local descriptors into a compact image representation. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3304–3311. IEEE, 2010. 5, 8

[25] James E Kelley, Jr. The cutting-plane method for solving convex programs. *Journal of the Society for Industrial & Applied Mathematics*, 8(4):703–712, 1960. 20

[26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 8, 9, 10, 13

[27] Gert RG Lanckriet, Nello Cristianini, Peter Bartlett, Laurent El Ghaoui, and Michael I Jordan. Learning the kernel matrix with semidefinite programming. *The Journal of Machine Learning Research*, 5:27–72, 2004. 19

[28] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 2169–2178. IEEE, 2006. 8

[29] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989. 8, 9

[30] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004. 8, 11

[31] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814, 2010. 9

[32] Florent Perronnin, Jorge Sánchez, and Thomas Mensink. Improving the fisher kernel for large-scale image classification. In *Computer Vision–ECCV 2010*, pages 143–156. Springer, 2010. 5, 8, 9, 10, 11, 12, 13, 16, 17

[33] Alain Rakotomamonjy, Francis Bach, Stéphane Canu, Yves Grandvalet, et al. Simplemkl. *Journal of Machine Learning Research*, 9:2491–2521, 2008. 20

[34] Jorge Sánchez, Florent Perronnin, Thomas Mensink, and Jakob Verbeek. Image classification with the fisher vector: Theory and practice. *International journal of computer vision*, 105(3):222–245, 2013. 5, 11

[35] Shai Shalev-Shwartz, Yoram Singer, Nathan Srebro, and Andrew Cotter. Pegasos: Primal estimated sub-gradient solver for svm. *Mathematical programming*, 127(1):3–30, 2011. 17

[36] Karen Simonyan, Omkar M Parkhi, Andrea Vedaldi, and Andrew Zisserman. Fisher vector faces in the wild. In *Proc. BMVC*, volume 1, page 7, 2013. 44

[37] Mingkui Tan, Li Wang, and Ivor W Tsang. Learning sparse svm for feature selection on very high dimensional datasets. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 1047–1054, 2010. 6, 19, 20, 43

[38] Koen EA Van de Sande, Jasper RR Uijlings, Theo Gevers, and Arnold WM Smeulders. Segmentation as selective search for object recognition. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1879–1886. IEEE, 2011. 9, 24, 42, 50

[39] Andrea Vedaldi and Brian Fulkerson. Vlfeat: An open and portable library of computer vision algorithms. In *Proceedings of the international conference on Multimedia*, pages 1469–1472. ACM, 2010. 29

[40] Andrea Vedaldi, Varun Gulshan, Manik Varma, and Andrew Zisserman. Multiple kernels for object detection. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 606–613. IEEE, 2009. 9

[41] Jason Weston, André Elisseeff, Bernhard Schölkopf, and Mike Tipping. Use of the zero norm with linear models and kernel methods. *The Journal of Machine Learning Research*, 3:1439–1461, 2003. 19

[42] Jianchao Yang, Kai Yu, Yihong Gong, and Thomas Huang. Linear spatial pyramid matching using sparse coding for image classification. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1794–1801. IEEE, 2009. 8

[43] Yu Zhang, Jianxin Wu, and Jianfei Cai. Compact representation for image classification: To choose or to compress? In *Proceedings of the The IEEE Int'l Conference on Computer Vision and Pattern Recognition (CVPR 2014)*, pages 907–914, 2014. 17, 18

A. Evaluation of average precisions

The performance of the proposed methods was evaluated on the Pascal VOC 2007 dataset [15]. The following two sections describe the evaluation protocol for the image classification and object detection tasks.

A.1. Image classification evaluation

In image classification, each image $\{I_1, ..., I_n\}$ is labeled with a subset of ground truth labels $\{1, ..., K\}$. The fact the task is defined as a multilabel problem means that classical evaluation using the confusion matrices is not possible. Instead, the image classifiers are obliged to output a score measure, that expresses the probability of each class being present in an image (i.e. the higher the more likely it is presence of given object class).

Once each image is labeled by these K probabilities, the evaluation may start. For each class separately the images are sorted in descending order according to the assigned class score measure. The precision-recall curve is then computed using the ground truth labels for each of the K orderings. The K AP metrics are then computed by obtaining a mean of precisions that are observed every time a new positive sample is recalled.

The final mAP metric is a mean over APs of each class.

A.2. Object detection evaluation

The object detection task evaluation protocol is very similar to the image classification one. The classifier is again required to output K sets of bounding boxes together with the score measure, that estimates the probability of presence of an object from given class inside each bounding box.

Each of the K sets of bounding boxes is then sorted according to the score measure. The detections are then labeled as true or false positives depending on their overlap with ground truth objects (for definition of the overlap measure see the next section). To proclaim a bounding box as a true positive detection its maximum overlap with a ground truth annotation inside an image must be higher than 0.5. Note that subsequent true positive detections of the same object are not counted and are regarded as false positive redundant detections.

The PR curve computation followed by the evaluation of the AP for each class then follows. Again, mAP is the mean of the K obtained AP measures.

A.2.1. Overlap measure

The measure of overlap between two bounding boxes is defined as the intersection-over-union metric. More precisely it is the area of the intersection of the two regions divided by the area of their union. The outputted number is in range $\langle 0, 1 \rangle$, where 1 means identical bounding boxes and 0 means that the bounding boxes do not share any common part.

B. Contents of the enclosed DVD

./thesis.pdf The electronic version of the thesis.
./pipe/ Contains examples of the two detection and image classification pipelines proposed in this thesis (see ./pipe/README.txt for details).
./pipe/data/imgs/ Example images from VOC-2007 test set.
./pipe/data/networks/ Pretrained image classification and object detection models.
./pipe/helpers/ A set of MATLAB helper functions.
./pipe/SelectiveSearchCodeIJCV/ The code that implements the object proposal method from [38].