MASTER'S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Fu Yongpan** Personal ID number: **506031**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Cybernetics**

Study program: **Open Informatics**

Specialisation: **Computer Vision and Image Processing**

II. Master's thesis details

Master's thesis title in English:

High-Resolution Images and Knowledge Distillation in Deep Metric Learning with Vision Transformers

Master's thesis title in Czech:

Obrazy s vysokým rozlišením a destilace znalostí v hlubokém metrickém učení s transformátory vidění

Guidelines:

Deep metric learning for vision is performed by optimizing a representation network to map (non-)matching image pairs to (non-)similar representations. During testing, the learned image representation is used to perform retrieval and find images from a large database that are similar to a query image. In fine-grained recognition tasks, such as recognition/retrieval of bird species, products, car models, mushroom species, etc., the details of the objects matter. Examples in the literature indicate that the resolution of the input image matters, and a large resolution is beneficial to capture the necessary details. In this project, we plan to explore the impact of resolution using visual transformer models, which are typically pre-trained for a fixed resolution.
Using large resolutions adds up to the total computational cost. Therefore, working at small resolution while achieving high performance is the end goal of this project. To handle this, the project will use knowledge distillation to use a network operating on a large resolution as a teacher and a network operating on a small resolution as a student, with the goal of improving the latter. The work should handle and take advantage of the specificities of visual transformer models when operating at varying resolutions. Two test-time tasks are going to be explored: i) conventional symmetric retrieval where both the query and the database images are processed with the same network at a small resolution ii) asymmetric retrieval where the query and the database images are processed with different networks at small resolution and large resolution, respectively.
The student is expected to:
1. Extend the implementation/methodology of [1] so that the training of the teacher and the student network work with FlexiViT, while taking advantage of the flexibility of this single model to work at multiple resolutions and patch sizes.
2. The final report should indicate good ways of optimizing the performance versus complexity trade-off with ViT models.

Bibliography / sources:

[1] Suma, Tolias, WACV 2023, Large-to-small Image Resolution Asymmetry in Deep Metric Learning
[2] Beyer et al, CVPR2023, FlexiViT: One Model for All Patch Sizes
[3] Dosovitskiy, Alexey, et al. "An image is worth 16x16 words: Transformers for image recognition at scale." ICLR 2020

Name and workplace of master's thesis supervisor:

doc. Georgios Tolias, Ph.D. Visual Recognition Group FEE

Name and workplace of second master's thesis supervisor or consultant:

Date of master's thesis assignment: **05.10.2023** Deadline for master's thesis submission: **09.01.2024**

Assignment valid until: **16.02.2025**

doc. Georgios Tolias, Ph.D. Supervisor's signature	prof. Dr. Ing. Jan Kybic Head of department's signature	prof. Mgr. Petr Páta, Ph.D. Dean's signature

III. Assignment receipt

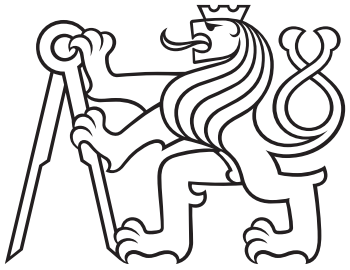
The student acknowledges that the master's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master's thesis, the author must state the names of consultants and include a list of references.

.	
Date of assignment receipt	Student's signature

Czech Technical University in Prague

Faculty of Electrical Engineering

Department of Cybernetics



Master's Thesis

High-Resolution Images and Knowledge Distillation in Deep Metric Learning with Vision Transformers

Bc. Yonpan Fu

Supervisor: doc. Georgios Tolias, Ph.D.

Study Program: Open Informatics

Field of Study: Computer Vision and Image Processing

January 9, 2024

Acknowledgements

I would like to thank my supervisor, Georgios Tolias, Ph.D., for his guidance and valuable advice and Ing. Pavel Suma for his assistance.

Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses.

In Prague on January 9, 2024 .....

Abstract

This thesis investigates the trade-off between performance and complexity with respect to various image resolutions using Vision Transformers in deep metric learning. The objective of metric learning is to use deep neural networks to embed images into representative vectors such that images of the same class cluster together in the feature space while maintaining separation between different classes. Vision Transformers, among various deep architectures, have proven efficient at extracting high-level semantics from diverse image content and are thus employed as the primary models. Nevertheless, high performance is often accompanied by substantial complexity. Knowledge distillation is utilized as an optimization technique to enhance the performance of cost-effective models under the guidance of more complex models. Moreover, image resolution significantly affects the model's performance. Therefore, this thesis examines the performance/complexity trade-off in asymmetric metric learning, where images are processed at different resolutions. The term resolution refers to either the input image resolution or the resolution of the patches that are separately processed at the processing stage of Vision Transformers.

Keywords: Deep Metric Learning, Image Retrieval, Vision Transformers, Knowledge Distillation

Contents

1 Introduction

The essence of metric learning lies in representing images within an embedding space where semantically similar images are mapped closely together, while dissimilar ones are positioned farther apart. This representation is crucial for various visual tasks. In image retrieval, the objective is to find images in a database that match the query's content, such as identifying images of the same bird species. For visual localization, images associated with a specific landmark are represented closely to reflect their proximity within the environment.

In recent years, particularly since the landmark achievements of convolutional neural networks (CNNs) in the image recognition task of 2012, deep learning has rapidly dominated the domain of computer vision. The features generated from deep learning models have progressively overtaken traditional hand-crafted ones, thanks to their powerful representational capabilities and advancements in modern computer hardware. Fully convolutional networks have demonstrated superiority over basic, linearly-connected feedforward networks. Numerous CNN variants, such as VGG and ResNet, have been proposed and recognized as robust architectures for image feature extraction. However, following the significant advancements made by transformer architecture in the field of natural language processing, researchers have begun adapting similar principles to computer vision. The Visual Transformer (ViT) has emerged as the most successful architecture incorporating the transformer paradigm for visual tasks, surpassing CNNs by applying an attention mechanism to patchified image features. Consequently, ViT has become the leading architecture for general visual tasks. Therefore, this thesis employs ViT as the primary model for exploration.

Examples in related literature indicate that image resolution plays a significant role in deep metric learning, particularly in fine-grained recognition tasks. Models trained on higher resolutions often exhibit enhanced representational capabilities, but this comes with increased training difficulty. A model that balances low test-time cost with high accuracy is highly desired. To address this, knowledge distillation emerges as an effective strategy. Here, a teacher model—usually larger and more complex—is trained first and used to guide the training of a student model, which is smaller. The objective is to enhance the student's performance beyond what it could achieve if trained independently. The student model mimics the teacher's output logits to absorb knowledge. However, some recent studies suggest that this network-wise asymmetric distillation in deep metric learning can sometimes be effectively replaced by resolution-wise asymmetric distillation. In this approach, the teacher and student models are architecturally identical but are trained

on different resolutions. The student processes images at a lower resolution, significantly reducing test-time retrieval costs.

1.1 Problem description and objectives

The Visual Transformer is typically trained with a fixed image resolution and patch size. Research indicates that ViT's performance diminishes when applied with varying patch sizes and image resolutions. Additionally, the test-time cost is affected by these two factors. In the context of knowledge distillation, several aspects merit investigation: if a teacher model is fixed to a large resolution and a specific patch size, various combinations of image and patch sizes can be considered for the student model. This raises the question of which combination is most effective in terms of the performance-cost trade-off. Furthermore, it is crucial to determine which combination benefits most from distillation compared to training the student model alone. To address these questions, a ViT model that underperforms with different patch sizes and resolutions is not an ideal benchmark.

One solution is to enhance the flexibility of ViT concerning patch size, enabling stable performance with input images segmented into various-sized patches. Recent literature introduced a novel method to make ViT adaptable to different patch sizes. Two key components are integral to this method: a resizing method that minimizes information loss after interpolating patch embeddings and a training mechanism that allows ViT to process varying sizes of patch embeddings. FlexiViT makes it feasible to distill knowledge from a teacher model to a student model with different image resolutions and patch sizes.

The objectives of this thesis are summarized into several key points:

- Review related work on asymmetric metric learning, its connection to distillation, and investigate various ViT variants.
- Develop algorithms for both symmetric and asymmetric learning, with implementation using the PyTorch library.
- Explore the performance/complexity trade-off by examining a range of image resolutions and patch sizes.

2 Related work

2.1 Asymmetric metric learning

Recently, asymmetric metric learning (AML) [1] has been studied to address the test-time complexity bottleneck posed by symmetric level learning [2, 3, 4]. The asymmetric setup

imposes a constraint that the representation spaces for the network operating on database images (a teacher network) and the one for query images (a student network) should be aligned and compatible. BCT [5] explores this feature compatibility learning issue and enforces that the features of the query model closely match those in the database model space. Other efforts [1, 6, 7, 8] have further attempted to improve compatibility across different models. HVS [6] utilizes neural architecture search to optimize the most effective model architecture that is aware of and adapts to compatibility requirements. FCT [9] employs a method where additional information, stored during training, is later used to adapt existing embeddings to different retrieval tasks without the need for retraining. Another approach [10], in an unsupervised manner, guides a student model to emulate the teacher's contextual image neighbor similarities in its embedding space. Beyond the feature compatibility problem, it should be noted that traditional asymmetric learning focuses on the network-wise difference between the teacher and student models. However, recent research [11] explores the impact of resolution differences using CNNs and finds that resolution asymmetry optimizes the performance/efficiency trade-off better than architectural asymmetry. Inspired by this work, this thesis focuses on resolution asymmetry, though employing different models for the study.

2.2 Knowledge distillation

Literature has shown that larger image sizes help capture more details, generally leading to higher performance [12]. However, models trained on larger images often come with increased computational complexity, posing challenges for deployment in resource-constrained environments. To address this, a technique for model compression, known as knowledge distillation (KD) [13], was developed to enable a smaller model to effectively learn from a large ensemble of models. This process involves the student model mimicking the teacher's predictions, allowing for the transfer of knowledge. In KD, a high temperature (T) is used for computing the softmax output of the cumbersome model, enabling the distilled model to learn soft targets, which effectively regularizes the learning process. Alternatively, some approaches [14, 15] transfer knowledge by matching the student and teacher's attention maps. Another method [16] introduced a noise-based regularizer for KD to mitigate overfitting. Inspired by GANs [17], Xu et al. [18] developed a new loss function learning algorithm using a conditional adversarial network. To harness the richer data information acquired by the teacher model, RKD [19] transfers mutual relations of data examples, extending KD's capability to allow the student to potentially outperform the teacher. MutualNet [20] adopts an innovative approach, dynamically adjusting image

resolution based on mutual learning to optimize the distilled network.

2.3 Vision transformers

The Transformer architecture [21] has achieved significant success in natural language processing, thanks to the self-attention mechanism that enables modeling dependencies in sequences, irrespective of the distance between elements [22, 23]. Recent efforts have focused on extending its application to image processing. A straightforward idea is to allow each pixel to attend to every other pixel in the image, but this is impractical due to high complexity. Parmar et al. [24] attempted to reduce computational costs by enabling each query pixel to attend only to its local neighborhood, though this approach loses global information. Another method maintains global self-attention cost-effectively by using scalable approximations [25]. As a generative model, Image GPT (iGPT) [26] applies self-attention across all pixels but with low image resolution and color space. A simpler method [27] divides the input image into 2×2 non-overlapping patches and feeds them into a Transformer encoder, but this is limited to small-sized images. ViT [28] uses a similar concept with various patch sizes for medium-resolution images and benefits from large-scale pre-training. ViT demonstrates effectiveness compared to other methods, and thus this thesis focuses on this model. One issue with ViT is that changing the patch size usually requires retraining. Recently, FlexiViT [29] was proposed to enhance flexibility in terms of patch size, using randomized patch sizes during training and a novel technique for resizing patch embedding. This aligns well with the thesis's focus, as varying the patch size influences complexity and maintains satisfactory accuracy.

3 Background

3.1 Metric learning

Representing input data in an embedding vector space is a desirable choice when dealing with high-dimensional data characterized by redundancy and noise. The embedded representation should reflect meaningful features of the original data. Based on this, some downstream tasks can be performed: clustering based on data labels, classification by assigning labels to all data, or image retrieval by searching for similar candidates for a given query.

In these scenarios, a variety of algorithms have been developed, each characterized by its approach to learning either the separation boundary or the distributions of the original or pre-processed data directly. Notable examples include Support Vector Machines, K-Means

clustering, and EM algorithms. These algorithms function within the data's embedding space, where the quality of the representation is crucial. Consequently, the ability to map matching (or non-matching) data pairs to similar (or dissimilar) representations is critical and significantly simplifies the clustering process. The mapping function can be custom-designed, tailored to specific evaluation criteria. The development of such a mapping function, which maintains the inter-class and intra-class similarities of data, constitutes the core of a branch of machine learning known as metric learning. Within the context of vision tasks in metric learning, the term "data points" refers to embedded image representations (vectors). Here, "metric" broadly denotes the similarity or distance between embedded input data. Therefore, in the subsequent text, "distance" is used interchangeably with "metric".

More formally, we group all data pairs into two sets, denoted as S^+ and S^-:

$$\begin{aligned} S^+ &= \{(x_i, x_j) \in X \times X, \text{ where } x_i \text{ and } x_j \text{ are a similar pair}\} \\ S^- &= \{(x_i, x_j) \in X \times X, \text{ where } x_i \text{ and } x_j \text{ are a non-similar pair}\} \end{aligned} \tag{1}$$

In the context of supervised learning, each data point is assigned a class label y. Here, similarity implies that two data points belong to the same class label:

$$\begin{aligned} S^+ &= \{(x_i, x_j) \in X \times X, \text{ where } y_i = y_j\} \\ S^- &= \{(x_i, x_j) \in X \times X, \text{ where } y_i \neq y_j\} \end{aligned} \tag{2}$$

The objective of metric learning is to learn a mapping function that minimizes the distance within classes and maximizes the distance between classes, subject to certain constraints:

$$\begin{aligned} \underset{\theta}{\text{minimize}} \quad & J(\theta) \\ \text{subject to} \quad & d_\theta(x_i, x_j) \leq \epsilon, \ \forall (x_i, x_j) \in S^+, \\ & d_\theta(x_i, x_j) \geq \delta, \ \forall (x_i, x_j) \in S^-, \\ & \epsilon \leq \delta \end{aligned} \tag{3}$$

The key constraint is that the distance between data representations from different classes should be greater than that of representations from the same class. This ensures that the closest neighbor to a given data point also shares its class (refer to Figure 1). The objective function, $J(\theta)$, which requires minimization, can take various forms. The distance function, denoted by d_θ is our primary focus. The subsequent sections will detail the different choices of objective functions and distance metrics.

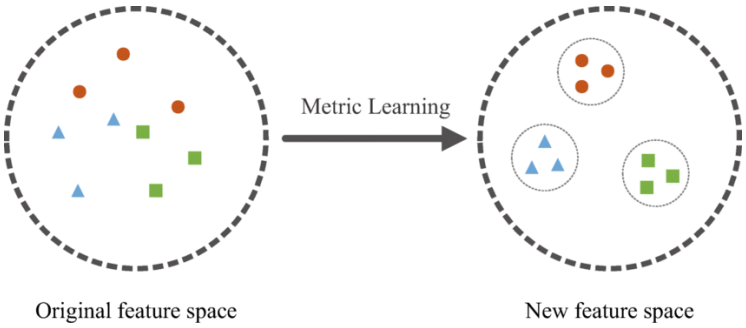


Figure 1: Feature space of metric learning. Data points are embedded image representations, with each class represented by a distinct color. Metric learning aims to cluster representations of the same class closely together while distancing those of different classes. (taken from [30])

3.1.1 Deep metric learning

In the evolution of machine learning research, there has been a shift from traditional handcrafted features to features automatically extracted by neural networks. In the era preceding the dominance of deep learning, data features were computed using well-established mathematical formulas, with substantial evidence supporting the effectiveness of these methods. These methods demonstrated significant discriminative power in practical applications and were relatively inexpensive in terms of computational resources. However, deep learning has elevated performance by fitting data to more complex functions, utilizing vast amounts of training data. In recent years, particularly following the groundbreaking work of AlexNet [31], deep learning has exhibited exceptional representational capabilities in computer vision tasks. Consequently, numerous studies have proposed replacing traditional machine learning methods in various scenarios with deep learning approaches. As a result, metric learning methodologies have been substantially enriched by deep neural networks.

Although various methods exist for implementing different deep neural network architectures, the final output from the network must be a single vector serving as a global descriptor for an image. This vector is utilized to compute the distance function in Equation (3). In this context, the distance function d_θ comprises two components: a network that maps input examples x to a vector, followed by the computation of a standard distance function, such as Euclidean distance or cosine similarity. The objective function $J(\theta)$ presents a variety of options and acts as a loss function for updating network parameters. Below, we introduce two widely used loss functions.

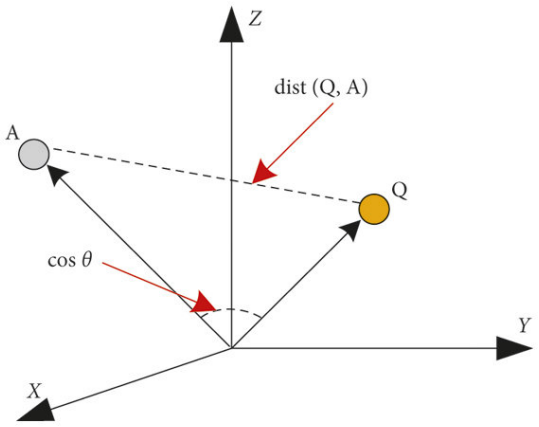


Figure 2: Two variants of distance function: Euclidean distance and cosine similarity. (taken from [32])

Contrastive loss

During the forward pass, the network processes a pair of input images, and the loss function is defined based on the consistency of their classes:

$$\ell(x_i, x_j) = \frac{1}{2} y_{ij} \|x_i - x_j\|_2^2 + \frac{1}{2}(1 - y_{ij}) \left[\tau - \|x_i - x_j\|_2\right]_+^2 \tag{4}$$

where $[\cdot]+ = \max(0, \cdot)$ and $y_{ij} = 1$ if the two images belong to the same class, otherwise it is 0. For positive pairs (same class), the second term becomes 0, and thus the first term is minimized, bringing their descriptors closer. Conversely, for negative pairs (different classes), the first term is eliminated, and the network minimizes the second term, ensuring the descriptors are sufficiently distant. The margin τ serves as a lower bound, ensuring that the distance between negative pairs is at least τ.

Triplet loss

Compared to contrastive loss, triplet loss instead takes a tuple of three images as input, in which one acts as the anchor and the other two as positive and negative candidate respectively. A candidate is positive if it has the same class as the anchor, and negative otherwise.

$$\ell(x_a, x_p, x_n) = \left[\|x_a - x_p\|_2^2 - \|x_a - x_n\|_2^2 + \alpha\right]_+ \tag{5}$$

Different from contrastive loss, triplet loss pushs positive and negative candidates together in the same time. Positive points are getting closer to the anchor whereas negative ones gets far away from it. Here again α acts as a margin to keep negatives far enough.

Negative mining

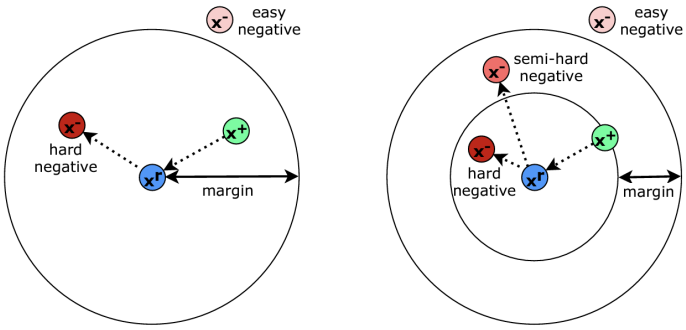
In practice, most pairs are negative, and their descriptors are sufficiently distant in the feature space. This can lead to zero loss values, resulting in no gradient being computed and consequently slow convergence during training. Random selection methods are not effective in addressing this issue. Furthermore, if some pairs already yield zero loss and require no further optimization, they may still be selected in subsequent iterations, which is inefficient.

In the context of triplet loss, negatives can be categorized into three types based on their distance from the anchor:

- *Hard negatives* are those mapped close to the anchor, with a distance even smaller than that between the anchor and the positive.
- *Semi-hard negatives* are farther from the anchor than positives but still within the margin boundary.
- *Easy negatives* are mapped beyond the margin, resulting in zero loss and eliminating the need for optimization.

Ideally, easy negatives should be excluded from the selection process during training. Both hard and semi-hard negatives are targets for optimization, but hard negatives should be prioritized due to their proximity to the anchor and larger loss values, which aid in convergence. Therefore, negative mining based on the distance to the anchor is an effective strategy for selecting training pairs.

For contrastive loss, the absence of an anchor means that negative mining is based on the mutual distance between positive and negative candidates. Positives are pre-selected to guide the selection of negatives, and they collectively serve as inputs for the network.



(a) Contrastive loss (b) Triplet loss

Figure 3: Two different loss functions and the type of negatives. x^r is the reference image, which performs as the anchor in triplet loss. (taken from [11])

3.2 Neural network models

As discussed in the previous section, deep metric learning leverages the feature extraction capabilities of neural networks, trained on large datasets. Recent research in network architecture in the field of computer vision has primarily focused on image classification tasks. In these tasks, networks are designed to map images to representations that align with specific class embeddings, with the expectation of achieving high accuracy on benchmark datasets. However, this thesis work focuses on metric learning, where the objective differs: to map images in such a way that the distance between classes is greater than the distance within a class. To elaborate, the features trained for classification tasks do not inherently learn clustering abilities. For instance, the distance between two descriptors from the same class need not be smaller than the distance between one of them and a descriptor from a different class, as long as they correctly match their respective class embeddings.

Although a network pretrained on a specific task may not be directly applicable to a different task, it can still serve as an effective initial point and be fine-tuned for our purposes. This methodology is known as transfer learning, wherein the robust feature extraction capabilities of a pretrained neural network are retained, making it a strong candidate for initial training. The basis for this transferability lies in the fact that trained networks tend to extract high-level features independent of the objects in the image, which are generally consistent across various images.

Transfer learning involves adapting the main structure of a model, rather than copying it entirely, by removing or replacing the last few classification or pooling layers that are more task-specific. The main architecture serves as the backbone, to which new layers can

be added if necessary. To elaborate, suppose the backbone outputs a feature map of shape (B, D, H, W), where B represents the batch size, and D, H, and W correspond to the number of channels, height, and width, respectively. A pooling layer takes this as input and outputs a tensor of shape (B, D). Generally, D does not match the number of classes, necessitating a linear layer to map it to a specific dimension for classification tasks.

The backbone constitutes the primary component of the network, where most operations occur. Selecting the right architecture is crucial in deep learning. Over the past years, numerous image models have been developed, each surpassing its predecessors in terms of complexity, design innovation, and architectural advancements. For instance, architectures like ResNet [33] introduced skip connections to mitigate the vanishing gradient problem, facilitating the training of deeper networks. EfficientNet [34] scaled up CNNs in a more balanced way across dimensions such as depth, width, and resolution. More recent architectures, like Vision Transformers, have moved away from traditional convolution-based methods, employing self-attention mechanisms to process images, demonstrating impressive performance across various tasks. Subsequent sections will delve into more details about some of these prominent architectures.

3.2.1 Convolutional neural networks

VGG

VGGNet [35], developed by Simonyan et al. in 2014, emerged as a revolutionary architecture at that time. Unlike previous CNNs, the authors focused on an often-overlooked aspect of CNN design: depth. In VGGNet, small 3×3 convolution kernels are utilized in all convolutional layers. A sub-brick comprises 2 or 3 convolutional layers followed by non-linearity activations, and is concluded with pooling layers. By stacking approximately 4 to 5 such sub-bricks, the main structure of VGGNet is formed, serving as the feature extraction branch. Subsequently, a few dense layers are appended to its end to ensure the output has the correct number of channels as required. This practice of dividing the model into two distinct modules has since become a standard in computer vision. This work demonstrated that increasing the depth of CNNs enhances performance in image recognition tasks.

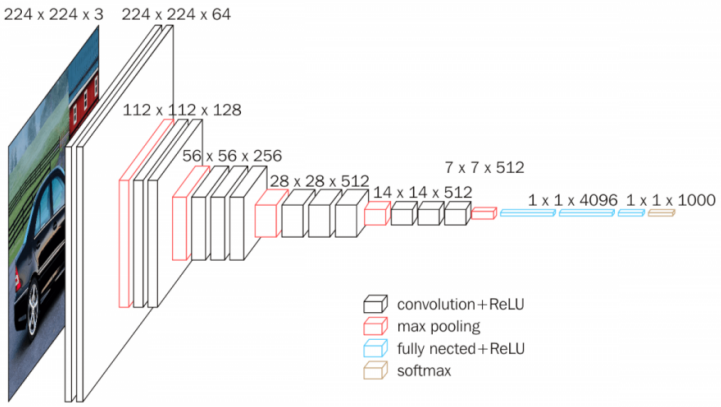


Figure 4: Visualization of the VGG architecture: As the network depth increases, feature maps expand in the depth dimension while their width and height decrease. The output from the feature extraction branch is then processed by the classification branch. (taken from [36])

ResNet

As previously discussed, deep models have significantly advanced many visual recognition tasks. However, simply adding more layers does not always enhance the model. One issue encountered was the vanishing/exploding gradients problem, which hindered convergence. Effective solutions, such as layer normalization and weight initialization, have been demonstrated to mitigate this issue to a certain degree. Another problem, identified by [33], is the degradation problem: accuracy saturates and then rapidly declines as network depth increases, leading to higher training errors in deeper models.

ResNet [33] addresses this issue by redefining layers to learn residual functions in relation to the layer inputs. If the original mapping to be learned is $\mathcal{H}(x)$, ResNet instead learns $\mathcal{F}(x) := \mathcal{H}(x) - x$. Consequently, the original mapping is transformed into $\mathcal{F}(x) + x$. It has been shown experimentally that optimizing the residual mapping is easier than optimizing the original mapping. For implementation, a "shortcut connection" is employed: retaining x, bypassing a few layers, and then adding it to the output, as depicted in the Figure 5

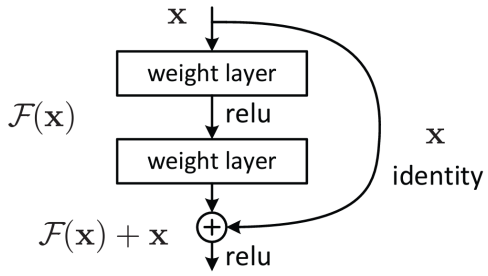


Figure 5: Skip connection in ResNet. (taken from [33])

This skip connection is a straightforward yet effective technique for training very deep networks. In their original work, ResNet models with hundreds or even thousands of layers could be trained effectively without increasing training error or compromising accuracy.

3.2.2 Transformer architecture

The Transformer architecture, based on the attention mechanism, was first proposed in 2017 by Vaswani et al. [21]. Originally designed for sequence-to-sequence models in natural language processing, it efficiently learns language semantics along the temporal dimension. The key feature is the self-attention mechanism, which functions like a query in a soft-version dictionary, assessing a representation's importance based on the inter-relationships within the sequence, thus maintaining long-term memory. Self-attention operations are performed in each Transformer block, and by stacking several blocks, a potent model for sequence input in language processing tasks is constructed.

Subsequent literature has demonstrated that the Transformer can be effectively adapted for computer vision tasks. Among these studies, the most notable model is the Vision Transformer [28], which segments a single image into spatially non-overlapping patches treated as sequential inputs for the Transformer block. Experiments have shown its superiority over CNN, efficiently leveraging attention benefits by treating images as sequential input, thus enhancing the learning of high-level spatial information.

In the following sections, we will first introduce the self-attention mechanism and then explain how the Transformer works. Finally, we will describe the concept of the Vision Transformer.

Self-attention mechanism

Given an input sequence, where each unit is an embedding vector, we produce a new sequence of the same length. Each output vector is a weighted sum of all input vectors,

with the weights computed using the dot product of each vector with all other vectors in the input. The advantage of using the dot product is that it effectively captures the mutual relationships between two vectors.

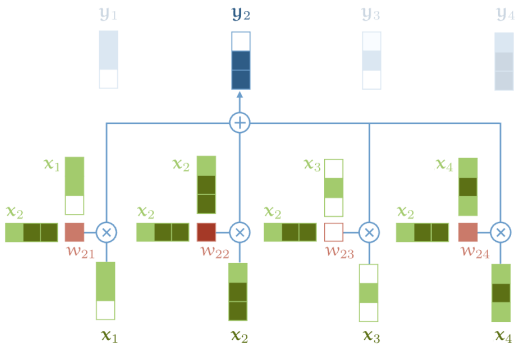


Figure 6: The output vector for a specific position is obtained by calculating a weighted sum of all input vectors. The weights are determined by the dot product between the input vector at this position and all other input vectors. (taken from [37])

Mathematically, we denote the input sequence by $x_1, x_2, x_3, x_4, x_5, x_6$ and output sequence by $y_1, y_2, y_3, y_4, y_5, y_6$, then the self-attention operation is:

$$\begin{aligned} y_i &= \sum_j w_{ij} x_j \\ w'_{ij} &= x_i^T x_j \\ w_{ij} &= \frac{\exp w'_{ij}}{\sum_j \exp w'_{ij}} \end{aligned} \tag{6}$$

If x_i and x_j are semantically close, the dot product result will be significant, thus substantially influencing the output value. A softmax activation is employed to keep the values within a small, positive range. These computations are efficiently executed in a vectorized routine, which also benefits GPU utilization. A key feature of this mechanism is that it uses only the input sequence for weights, with all operations confined to this set. This is the rationale behind the term "self-attention": each vector attends to all other vectors from the same set in a distinct manner.

Another feature of this mechanism is the treatment of all vectors from three distinct perspectives: query, key, and value. First, when a vector is used in the weighted sum that ultimately yields the output, it is termed the value. Second, when it corresponds to the current output and is matched against every other input vector, it is known as the query. Third, the vector against which the query is matched is referred to as the key. These terms are derived from conceptualizing the mechanism as a type of soft version of a dictionary.

In a standard dictionary, retrieving the content provides the exact value corresponding to a key. However, in this soft version, the retrieved result is a weighted sum of all values in the dictionary, with weights computed based on mutual dependency. Hence we treat attention as a soft dictionary:

- Key, query, and value are all represented as vectors.
- Every key matches the query to a certain degree, as indicated by their dot product.
- A mixture of all values is returned, with the mixture weights being the softmax-normalized dot products.

Attention with query, key and value from the same set is hence called self-attention.

Linear transformations

To enhance the capability of self-attention, transformations are applied to the three roles of key, query, and value. Rather than employing three separate vectors for these roles, a single vector is transformed using three different matrices to generate q, k and v:

$$\begin{aligned} k_i &= Kx_i + b_k \\ q_i &= Qx_i + b_q \\ v_i &= Vx_i + b_v \end{aligned} \tag{7}$$

Here, Q, K and V are linear matrices containing learnable weights that enable the single vector to behave differently. This approach introduces additional parameters in the self-attention layer, thereby increasing its flexibility.

Multi-head attention

The concept of multi-head attention arises from the recognition that different embeddings relate to each other through various relationships. For instance, in natural language processing, within the sentence “this restaurant was not too terrible” the word “terrible” is a descriptor of “restaurant” while “not” conveys opposition to “terrible.” Traditional self-attention, however, treats the contribution of these words to “terrible” uniformly. To enable the network to model these diverse types of relationships, self-attention can be divided into multiple “heads.” Each head employs its own set of transformations Q^r, K^r, V^r, where r denotes the r_{th} head, allowing for varying levels of attention. Multi-head attention is computed in parallel, resulting in a concatenated output vector.

Transformer as a complete architecture

A layer incorporating all the aforementioned operations constitutes a self-attention layer. To develop a comprehensive model, multiple self-attention layers must be stacked systematically. Additionally, it is common to include pooling layers at the end of the stack to generate the final output for various tasks. For example, in classification tasks, global sum pooling is effective for extracting global information. This entire network structure is referred to as the Transformer model. Essentially, any sequence-based model that predominantly utilizes self-attention to transmit information along the temporal dimension can be classified as a Transformer model.

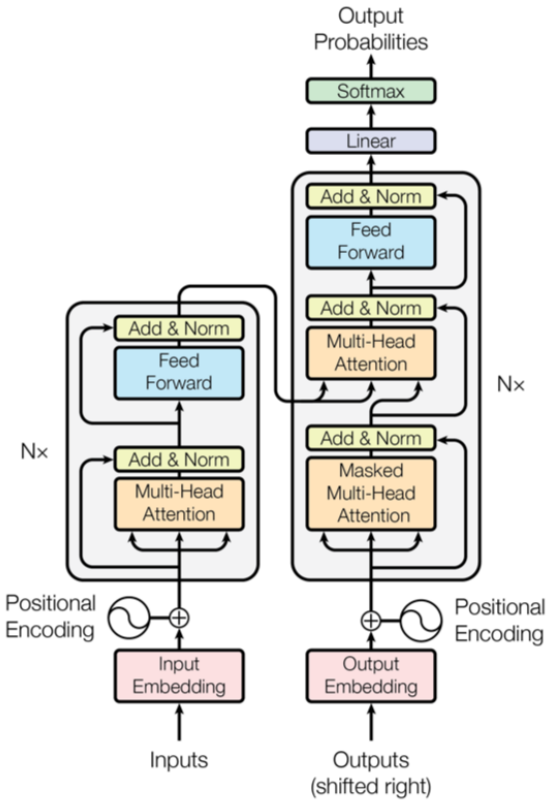


Figure 7: The Transformer architecture. (taken from [21])

Positional information

So far, the only issue we need to address with the Transformer model is its lack of inherent "sequential" information, despite processing sequence data. Consider a scenario where we permute the input words into a different order, creating completely different meanings. In such a case, the self-attention layers would produce equivariant vectors. However, the global pooling layer would render these vectors invariant, ultimately leading to an identical

final output label. Therefore, it is crucial for the Transformer block to encode sequential information. To break this equivariance, several methods have been developed:

- Positional Embedding: This approach assigns an embedding vector to every position in the sequence, similar to assigning embedding vectors to each word in the vocabulary. Each position has a unique embedding vector, distinct from others. These positional embedding vectors are added to the word embeddings. While simple to implement, this method cannot assign vectors to positions in sequences longer than those on which the model was trained.

- Positional Encoding: This technique addresses the limitations of positional embedding by converting fixed-length discrete representations into continuous representations. Each dimension of a positional vector takes a value from a periodic function. The number of functions used corresponds to the number of dimensions in the vector. Therefore, positional encoding vectors can theoretically represent an infinitely long input sequence.

Vision Transformer

The transformer architecture, initially designed for and becoming the standard in natural language processing due to its proficiency in learning sequential information, was adapted for computer vision tasks by Dosovitskiy et al. [28]. This adaptation led to the creation of the Vision Transformer (ViT), which demonstrates excellent performance compared to convolutional networks while requiring fewer computational resources for training.

The core concept involves dividing an image into several non-overlapping patches and transforming them into vector representations through linear mapping. These vectors then serve as inputs for the transformer blocks. In this approach, image patches are treated similarly to tokens (words) in NLP applications: each embedded image patch maintains semantic relations with all other patches, and the order of these patches is significant. This process closely mirrors the behavior of words in a sentence, making the embedding of image patches in computer vision as straightforward as embedding words in NLP.

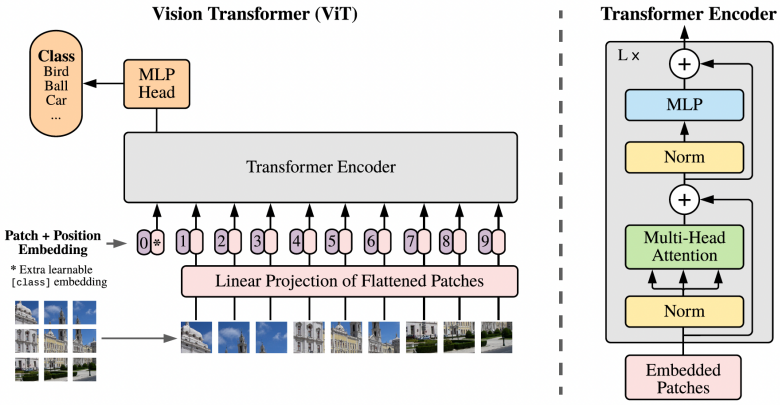


Figure 8: The Vision Transformer Architecture: The input image is split into a sequence of fixed-size patches. These patches are then linearly projected into embedding vectors, which are fed into the standard Transformer encoder. (taken from [28])

As depicted in the figure, ViT closely retains the original transformer architecture, with the addition of extra learnable classification tokens to the sequence. A notable characteristic of ViT is its performance when trained on mid-sized datasets without strong regularization, where it tends to achieve modest accuracies compared to state-of-the-art CNNs like ResNet. Transformers, lacking certain inductive biases such as translation equivariance and locality inherent in CNNs, often struggle with effective generalization in scenarios with limited training data. However, with substantially larger datasets, the scale of data compensates for the disadvantages posed by the lack of inductive biases. Consequently, ViT outperforms CNNs in tests when trained on large datasets.

3.3 Image retrieval

Image retrieval, considered a test-time task, utilizes models trained during metric learning. Techniques like k-means or other nearest neighbor search algorithms assist in finding the most similar images from a database given a query image.

In practice, two critical aspects are evaluated to determine the efficiency of the retrieval result: test-time (query-time) complexity (measured in GLOPS) and retrieval accuracy. GLOPS, or floating point operations per second, calculates the number of operations executed each second, directly reflecting the time taken during testing. Retrieval accuracy is gauged using mean Average Precision (mAP) across all database images, ranked by their similarity to the query. To compute mAP, two further criteria are used: Recall@k and Precision@k for the top-K selected candidates based on ranked similarity.

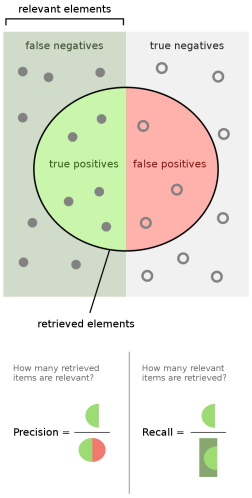


Figure 9: Recall and precision in metric learning. (taken from [38])

- Recall@k measures the proportion of relevant items retrieved from all available relevant items in the database, serving as a measure of completeness.
- Precision@k evaluates the fraction of retrieved items that are relevant, acting as a measure of accuracy.

These two metrics, Recall@k and Precision@k, depend on the number of ranked candidates in the database. By iterating k from 1 to the total number of database images, a series of values for these metrics is obtained. The average precision (AP) is then defined as the area under the precision-recall curve, which can be computed as follows:

$$AP = \sum_{k=1}^{N} \text{prec}(k)\Delta\text{recall}(k) = \sum_{k=1}^{N} \text{prec}(k)\frac{\text{rel}(k)}{T} \tag{8}$$

Here, Mean Average Precision (mAP) is the mean of AP over various queries. In the work presented in this thesis, mAP is computed across all database images, each selected as a query once.

In traditional image retrieval systems, query and database images are treated equally, compared using identical features and representations. Both query and database images undergo processing using the same network architecture and at the same resolution, a method known as symmetric retrieval. However, when computational resources are constrained or fast retrieval is essential, this approach may present limitations due to network or image resolution constraints. An alternative, more efficient strategy involves treating query and database images differently, such as utilizing distinct networks or processing them at different resolutions.

- Network-wise asymmetry: Here, query images are processed using a smaller network architecture compared to that used for database images. For query processing, a more compact or computationally lighter network is chosen to facilitate faster search times. Conversely, the database images are processed using a more complex and powerful network, providing richer and more accurate feature representation.

- Resolution-wise asymmetry: This involves processing images at varying resolutions during retrieval. Database images are stored and processed at a higher resolution to encapsulate greater detail, while query images are processed at a lower resolution for speedier processing. This method is particularly advantageous in mobile applications where query images are captured by lower-resolution cameras, or in scenarios with limited bandwidth.

Given the significance of resolution in fine-grained image recognition tasks, this thesis primarily concentrates on exploring and implementing resolution-wise asymmetry.

3.4 Knowledge distillation

Enhancing the performance of machine learning algorithms typically involves training multiple models for the same target and then aggregating their predictions. However, this ensemble approach can be cumbersome and computationally expensive. An efficient alternative proposed in the literature is knowledge distillation (KD), wherein a teacher model transfers its learned "soft-knowledge" to a student model, which typically incurs a lower inference cost. Unlike traditional label-guided supervised learning, where the model directly learns from data distributions, KD enables the student model to mimic the output logits of the teacher model. This teacher model is pretrained on the same dataset but possesses greater complexity.

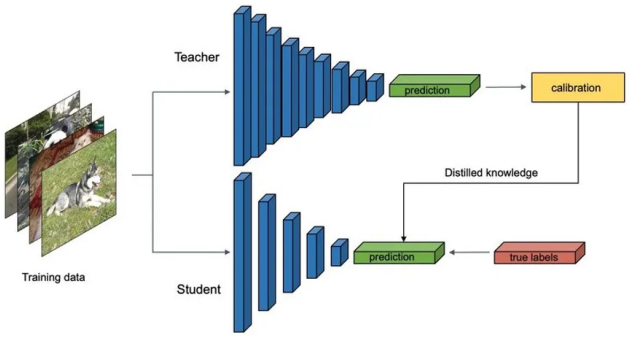


Figure 10: Knowledge Distillation: In addition to utilizing data labels, the student model employs the teacher's output logits as guidance for knowledge distillation. (taken from [39])

Connection to asymmetric retrieval

In the context of asymmetric retrieval, as previously discussed, a challenge arises when the model trained on lower resolution images exhibits less efficacy compared to one trained on higher resolution, thereby constraining retrieval accuracy. Knowledge distillation offers a solution to this problem. By applying KD, the student network can acquire the robust feature extraction capabilities of the teacher network, thus enhancing prediction accuracy even when operating on lower resolution images.

4 Benchmark dataset

In this section, we introduce the benchmark dataset used for evaluating the complexity/performance trade-off in metric learning. The Caltech-UCSD Birds-200-2011 (CUB-200-2011) dataset [40] is the most widely used for fine-grained visual categorization tasks. It comprises 11,788 images across 200 different bird categories. The dataset's original training and testing split is approximately 50%, with 5,994 images for training and 5,794 for testing. Each image in the dataset is accompanied by a categorical label, a bounding box, and even natural language descriptions, making it suitable for various vision tasks, including classification, image generation, image retrieval, and multi-modal tasks, among others.

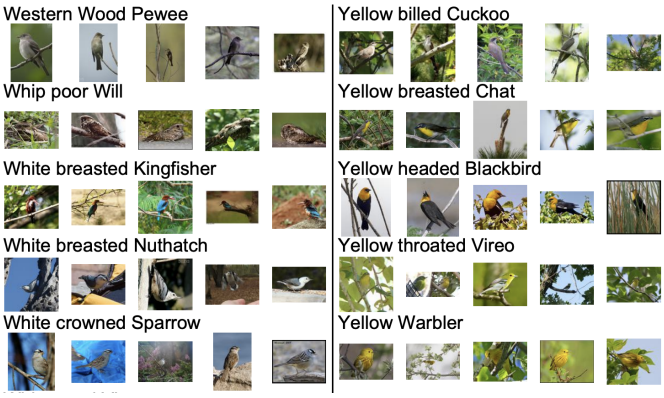


Figure 11: Examples of images from the CUB-200-2011 dataset. Each image depicts birds from the same category, captured against diverse background environments. (taken from [40])

In standard metric learning setups, the classes seen during training are usually different from those used for testing. This approach assesses how well the trained representations perform on unseen classes, adding an extra level of difficulty distinct from image classification tasks, where the testing phase uses the same class pool as training. To accommodate this requirement, we use the first half of the classes for training and the latter half for testing to maintain a balanced train-test division.

5 Methods and implementation details

This project aims to explore two test-time tasks:

- Conventional symmetric retrieval, where both the query and the database images are processed using the same network at identical resolutions. This task involves pre-training several teacher models operating at different resolutions using Vision Transformer architectures. We will then report on the symmetric retrieval performance and test-time complexity, considering two dimensions: image resolution and patch size. This analysis will aid in selecting an appropriate teacher model for asymmetric distillation.

- Asymmetric retrieval, where the query and the database images are processed using different networks, specifically at small and large resolutions, respectively. Our focus for the network difference is primarily on the resolution-wise aspect. Two identical model architectures trained at different resolutions will have distinct weights, thus differing from each other. As indicated by previous work [11], this resolution asymmetry proves to be a more effective approach for optimizing the performance/efficiency trade-off compared to architecture asymmetry.

A clarification regarding the term "resolution-wise" asymmetry and its implications for the performance/efficiency trade-off is necessary. This term encompasses two dimensions of adjustment: altering the image resolution (size) and varying the patch size of the image. Both modifications impact the complexity of the process. Therefore, in this thesis, the term "resolution" may refer to either image resolution adjustments or changes in patch size, depending on the context.

In the following sections, we detail the methodologies for both symmetric and asymmetric learning. The implementation is conducted using the PyTorch library. Before delving into these details, it is necessary to define some mathematical representations for clarity. Given an input image x at resolution r, which can be either a query or a database image, we feed it into a network f to obtain the embedded vector. The network is parameterized by its weights θ and patch size p. The output vector for the input image is then formulated as follows:

$$v = f(x_r; \theta, p) \tag{9}$$

5.1 Symmetric metric learning with identical resolutions for query and database images

Image retrieval is a test-time task that leverages a network trained for metric learning. In this process, image representations from the same class are brought closer, thereby facilitating more effective nearest neighbor search. This section outlines the symmetric training approach, focusing on calculating the similarity between a query image q and a database image x. The feature vectors for both images are extracted using the same network f, as detailed below:

$$\begin{aligned} v_{db} &= f(x_r; \theta, p) \\ v_q &= f(q_r; \theta, p) \end{aligned} \tag{10}$$

5.1.1 Model used

As previously mentioned, the Vision Transformer (ViT) divides the input image into several non-overlapping small patches, each with a shape of $(ps, ps, 3)$. These are then linearly projected to obtain a 1-D vector of shape (1, $num\_dimensions$) for each patch. Here, $num\_dimensions$ is set to 386 and 768 for small and base sizes, respectively. These patch embeddings function similarly to embedded words in transformers for language tasks. This patchification results in a sequence of vectors, to which a positional embedding vector is added for each, incorporating information about their positions in the sequence. Similar to BERT's class token [41], a learnable embedding is also added to the sequence of embedded patches. The sequential patch embeddings then serve as input to the transformer encoder. The output shape is $(B, num\_patches+1, D)$; however, a single vector is required as the global feature for each image. Various aggregation methods have been explored, including the CLS token, the mean of the patch features alone, and the mean including the CLS token. In practice, it is found that using the output CLS token alone provides the best image representation. During training, both the patch size and image size are fixed, eliminating the need for interpolation in resizing patch embeddings and positional embeddings.

In the case of using FlexiViT, every forward pass detail remains identical to the standard ViT, with two exceptions: the use of a variety of different patch sizes from a predefined pool (40, 30, 24, 20, 15, 12, 10) with certain probabilities during training, and a distinct patch embedding resizing technique — PI resize — as outlined in the original work [29]. Unlike the standard ViT, FlexiViT defines an underlying patch size of 32×32 and a position embedding size of 7×7. Upon receiving an input image, FlexiViT selects a patch size at random from the pool to PI resize the patch embedding. This alters the sequence length

and, consequently, the positional embeddings. FlexiViT employs bilinear interpolation, consistent with the standard ViT, to resize position embeddings. The final output vectors of both models are L2-normalized to mitigate scale issues during neighbor searching in retrieval tasks.

For weight initialization, there are two options: pre-training on either the 1k subset or the 21k subset of ImageNet [42]. The latter, being more extensive, provides a richer information source. Models pre-trained on the 21k subset generally exhibit higher performance and better generalize to other datasets.

5.1.2 Algorithms for metric learning

In metric learning, the objective is to minimize intra-class distances while maximizing inter-class distances. To this end, three distinct roles are assigned to all image instances in the training set. Each image serves as an anchor once, with those belonging to the same class as the anchor being designated as positives, and those from different classes as negatives. Triplet loss, defined based on these three roles, is utilized to optimize model parameters. Prior to the forward pass, a negative mining procedure is necessary for the given anchor to select candidate images for these roles. Negative mining is based on distance, favoring hard or semi-hard negatives that are close to the anchor due to their higher probability of selection. Positives are chosen within the batch, and the entire training set is pre-permuted to ensure each batch contains a fixed number of positives and negatives.

As feature representations shift in the space with each iteration due to weight updates, some instances previously categorized as negatives may no longer be negatives, necessitating the re-computation of negative mining. This re-computation is performed at the end of each epoch rather than after each optimization step.

5.1.3 Data pre-processing

Data preprocessing is a crucial step before training, involving image transformation that encompasses various data augmentation techniques and resizing the input image to a specified fixed resolution. These transformations enhance the model's generalization capability and aid in mitigating overfitting. In the following sections, we detail the distinct transformations employed for the training and testing datasets:

- In line with standard protocol [12, 43, 44], the transformation during training includes a random crop [45] followed by a random horizontal flip with a probability of 0.5.

- For testing, the transformation is a center crop. It is important to note that this process should not include any randomness to ensure that the representations produced by different experiments are comparable.

Additionally, all images are standardized to achieve a mean value of 0 and a standard deviation value of 1, thereby addressing the scale issue inherent in the original data.

5.2 Resolution-wise asymmetric metric learning

For the asymmetric learning experiments, we initially utilize the models trained during symmetric training and evaluate their performance across various resolutions and patch sizes. Specifically, database images are processed at a higher resolution, while query images are processed at a lower resolution. In addition to the difference in image size, the disparity in patch size is also incorporated into the experiments. Mathematically, the networks extract image features as follows:

$$\begin{aligned} v_{db} &= f(x_{r_{db}}; \theta, p_{db}) \\ v_q &= f(q_{r_q}; \theta, p_q) \end{aligned} \tag{11}$$

where $r_{db} \neq r_q$ and $p_{db} \neq p_q$ in the asymmetric setup. Both the reduction in image size and the increase in patch size contribute to more representative descriptors, leading to different combinations that result in varying accuracy/complexity trade-offs.

As for knowledge distillation, initializing the student network with the teacher network's weights is crucial to ensure compatibility in the feature space. The teacher's parameters are frozen, providing consistent guidance without any randomness:

$$v_{db} = f(x_{r_{db}}; \theta_t, p_{db}) \; v_q \quad = f(q_{r_q}; \theta_s, p_q) \tag{12}$$

The student network's parameters θ_s initially duplicate the teacher's parameters θ_t, but they are updated during training, leading to $\theta_s \neq \theta_t$. Optimizing the student network requires an effective distillation loss. While the straightforward choice is the absolute regression loss, which calculates the L2 distance between the student's and teacher's output logits, incorporating relational information from the data into the loss terms often yields better results. This thesis utilizes a modified relational loss based on images' coupled augmentations, as proposed in previous work [11].

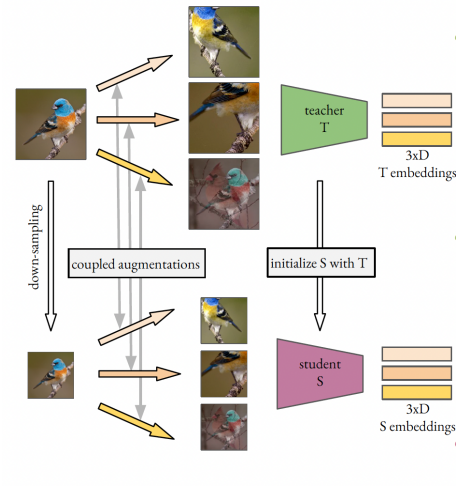


Figure 12: Coupled augmentations: The output embeddings from both networks are used to compute relational and absolute losses. (taken from [11])

Coupled augmentations involve dividing images for the student model in the same manner as for the teacher model, ensuring that each pair of corresponding image segments shares similar semantic content up to scale of resolutions. Both models process their respective image segments to produce output embeddings, with an equal number of image segments, which are then used to compute both the absolute regression term and the relational loss simultaneously:

$$
\begin{aligned}
l^1_{\text{rel}} &= \sum_{i \neq j} (\text{MSE}(t_i, t_j) - \text{MSE}(t_i, s_j))^2 \\
l^2_{\text{rel}} &= \sum_{i \neq j} (\text{MSE}(t_i, t_j) - \text{MSE}(s_i, s_j))^2 \\
l_{\text{abs}} &= \sum_{i=j} \text{MSE}(t_i, s_i) \\
l &= l^1_{\text{rel}} + l^2_{\text{rel}} + l_{\text{abs}}
\end{aligned}
\tag{13}
$$

where t_i and s_i represent the embedded feature vectors from the teacher and student models for the i_{th} image segment, respectively. l^1_{rel} and l^2_{rel} serve as the relational components of the distillation loss, while l_{abs} acts as the absolute term. The final loss is a summation of them. This loss function, by enabling the student to learn the inter-segment relationships captured by the teacher network, outperforms the pure absolute distillation loss.

6 Experiment results with symmetric metric learning

6.1 Comparison between CNN and ViT

Previous work [11] investigated the performance/complexity trade-off using the ResNet architecture, a prominent CNN in vision tasks. However, as previously discussed, the Vision Transformer (ViT) has demonstrated superior performance in certain scenarios. In this section, we conduct comparative experiments between ResNet and ViT to identify scenarios where ViT outperforms CNNs in terms of efficiency.

Both networks follow identical training and testing procedures, with their architecture being the only difference. For ResNet, both the 50-layer and 18-layer versions are employed. It is important to note that previous work [11] incorporated an additional whitening layer to achieve a specific output dimension, a step not required in ViT due to its CLS token output. Therefore, to maintain a fair comparison, this linear layer is omitted in our experiments. FlexiViT, as an alternative to the standard ViT, is utilized for its convenience in varying patch sizes during testing, which circumvents the need for re-training the standard ViT with different patch sizes.

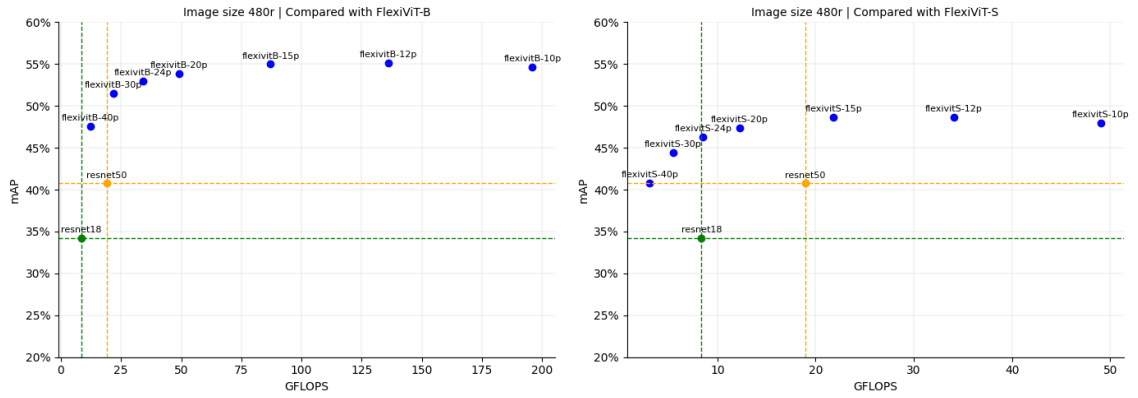


Figure 13: Comparison of symmetric accuracy and extraction cost for FlexiViT-B, FlexiViT-S, ResNet18, and ResNet50 at 480 image resolution. The FlexiViT models are trained and evaluated using various patch sizes: 40, 30, 24, 20, 15, 12, 10. A cross-dashed line indicates the FlexiViT setups that outperform ResNet in terms of the performance/complexity trade-off.

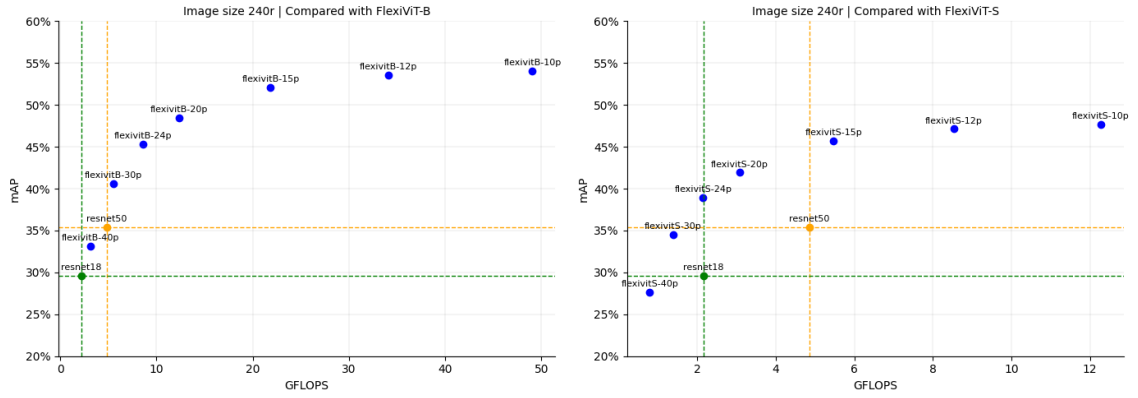


Figure 14: Comparison of symmetric accuracy and extraction cost for FlexiViT-B, FlexiViT-S, ResNet18, and ResNet50 at 240 image resolution. The FlexiViT models are trained and evaluated using various patch sizes: 40, 30, 24, 20, 15, 12, 10. A cross-dashed line indicates the FlexiViT setups that outperform ResNet in terms of the performance/complexity trade-off.

As illustrated in Figures 13 and 14, FlexiViT generally achieves significantly better retrieval accuracy. However, when the patch size is too small relative to the image size, the complexity greatly increases, exceeding the requirements for query-time speed. In every combination of model size and patch size, there is always a FlexiViT model that outperforms in accuracy while maintaining a query extraction cost comparable to, or even lower than, ResNet. For instance, at an image resolution of 240, FlexiViT-S/24p outperforms ResNet18 by 53% with nearly the same extraction cost. Compared to ResNet50, several FlexiViT setups are more efficient at resolution 480, including FlexiViT-S/20p, FlexiViT-S/24p, and FlexiViT-S/30p, all offering higher accuracy at a lower cost.

Given these results, we observe that the ViT model is superior to the most commonly used CNNs in our task when evaluated in a properly optimized setup in terms of the performance/complexity trade-off.

6.2 Transferability of ViT pretrained on ImageNet - no fine-tuning

Transfer learning [46] is a technique in machine learning where a model developed for a particular task is reused as the starting point for a model on a different task. Essentially, it involves taking a pre-trained model (a model trained on a large dataset) and fine-tuning it with a smaller dataset for a different but related problem. Throughout our experiments, we take the advantage of transfer learning to initialize our models as good starting point for training. In computer vision field, the most widely used pre-trained source is ImageNet

[42], a large visual database designed for use in visual object recognition research. It contains a larger variety of high resolution images compared to previous image databases. Two subsets of it named ImageNet-1K (with 1,000 categories) and ImageNet-21K (with around 21,000 categories) are used to see how well the pre-trained model when transferred to our image retrieval task.

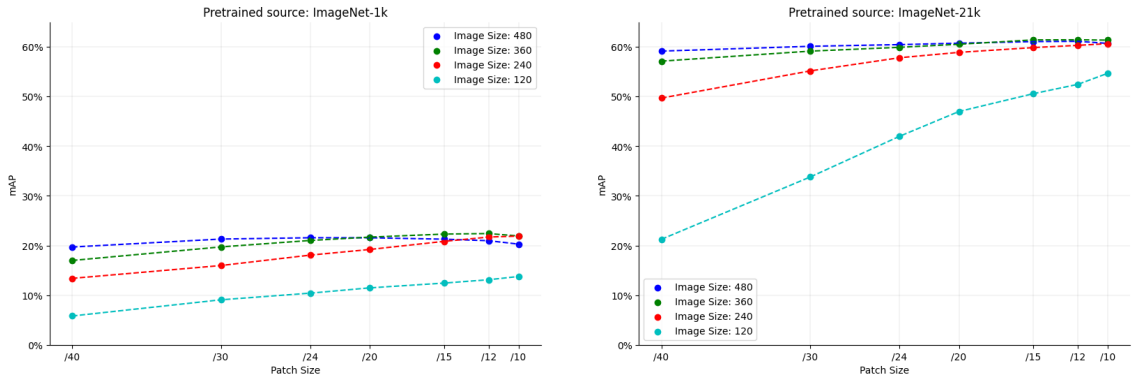


Figure 15: FlexiViT-B, pretrained on ImageNet-1k and ImageNet-21k, is utilized to evaluate retrieval accuracy. Even without fine-tuning on the CUB-200-2011 dataset, the pretrained models demonstrate satisfactory results with ImageNet-1k and excellent results with ImageNet-21k. Furthermore, pretraining on the larger-scale dataset, ImageNet-21k, is observed to significantly enhance performance.

FlexiViT VS standard ViT

In the previous section, we observed that a FlexiViT model pretrained on ImageNet can be directly used to initialize a model for metric learning on our retrieval dataset. This property also holds true for the standard ViT model, with the notable difference being the absence of flexibility in terms of patch size.

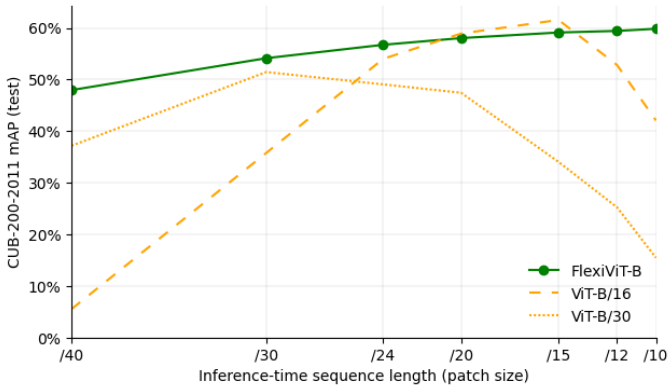


Figure 16: Standard ViTs lack flexibility in patch size. However, FlexiViT can be trained to offer this flexibility with minimal performance loss.

As depicted in Figure 16, within our retrieval dataset, the standard ViT demonstrates optimal performance only at its trained patch size. Its performance degrades significantly when evaluated with different patch sizes. In contrast, FlexiViT maintains consistent stability across a variety of patch sizes, due to the PI resize technique and exposure to various patch sizes during pre-training.

6.3 Gains brought by flexible training with PI resize

Although literature indicates that FlexiViT maintains competitive stability with respect to patch size compared to the standard ViT on the ImageNet image classification benchmark, it remains to be seen whether its flexible training approach and the PI resize trick are equally effective in metric learning tasks. To investigate this, we plan to conduct comparative experiments by fine-tuning both a standard ViT and a FlexiViT under identical conditions.

Experiment setup

To ensure a fair comparison, four models with the same architecture and complexity (ViT) are established prior to training. A consistent image resolution of 240 is used to eliminate scale bias. For initialization, two models are based on pretrained weights from ViT-B/16 and the other two from ViT-B/30. During training, we maintain both flexible and fixed training regimes for the models initialized from the same weights. The details of the setup are summarized in the table:

Setup / Model ID	ViT Size	Init	Training Mode
1	B	ViT-B/16-i21k	Fixed patch size(16)
2	B	ViT-B/16-i21k	Flexible training + PI resize
3	B	ViT-B/30-i21k	Fixed patch size(30)
4	B	ViT-B/30-i21k	Flexible training + PI resize

Table 1: The setups for training 4 models to compare the benefits from FlexiViT.

The decision to initialize separate models with identical weights but employ different training modes aims to determine whether flexible training, as in FlexiViT, contributes to stability with respect to patch size compared to standard ViT training in our metric learning task. The evaluation metric used in this context is mean Average Precision.

Results and analysis

All training sessions reached convergence after approximately 30 to 50 epochs, followed by a tendency to overfit. Consequently, we monitored the mean Average Precision (mAP) on the test set after each epoch and saved the best-performing model. The results are depicted in the figure below.

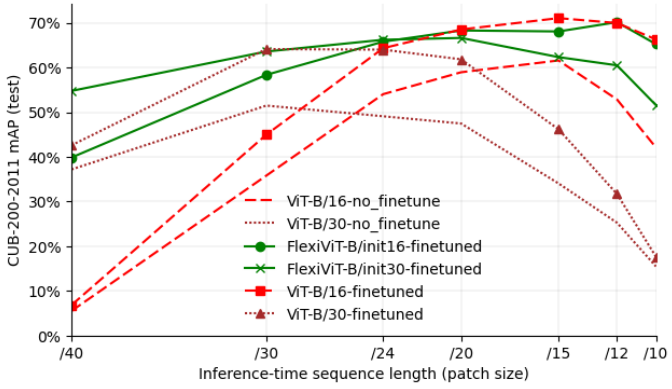


Figure 17: After training, FlexiViT shows satisfactory stability in patch size compared with standard ViT.

For ViTs initialized and trained on specific patch sizes, the evaluation performance improved by 15% to 30%. However, limitations become apparent when evaluated on patch sizes different from those used during training. In contrast, models trained with flexible patch sizes using the PI resize technique exhibited equal or improved performance

across all patch sizes, underscoring the clear advantages of this approach.

Impact of initialization

As discussed in the background, a good initialization always generate a better trained model. Using pre-trained weights trained on a benchmark dataset during initialization is a key point in transfer learning. In this section we compare the impact of initialization using different weights: from ViT-B/16 and ViT-B/30 which are trained in a fixed patch size (16 and 30 respectively) and from FlexiViT which is trained for uniformly-distributed patch sizes.

As established in the background section, effective initialization typically leads to better-trained models. Utilizing pre-trained weights from a benchmark dataset during initialization is a pivotal aspect of transfer learning. In this section, we examine the impact of initialization using different weights: those from ViT-B/16 and ViT-B/30, pre-trained with fixed patch sizes of 16 and 30 respectively, and those from FlexiViT, which is pre-trained with uniformly-distributed patch sizes. ImageNet-i21k is used as the pre-trained source.

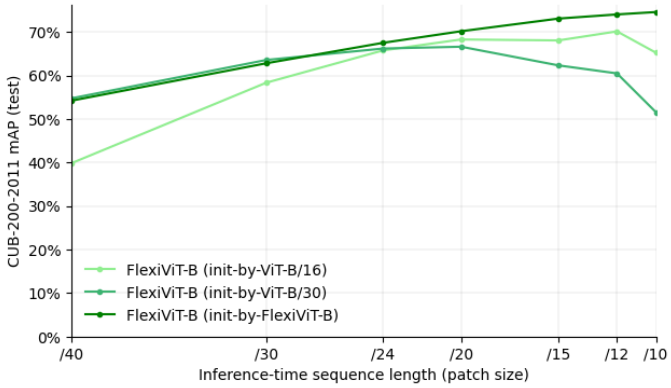


Figure 18: Fine-tuning FlexiViT-B models using three different initializations: standard ViT-B/16, standard ViT-B/30, and FlexiViT-B. The model fine-tuned with FlexiViT-B as the initial weights exhibits the best performance.

Although FlexiViT exhibits considerable flexibility in patch size, our experiments indicate that without proper initialization, the model may not achieve optimal performance. The adaptability of FlexiViT, developed through pre-training on ImageNet-21k, can be effectively transferred to fine-grained datasets in metric learning tasks, playing a significant role during fine-tuning training.

6.4 Resolution sensitiveness

In this section, we investigate the resolution sensitivity of Vision Transformers, specifically how standard ViT and FlexiViT perform when trained on one image resolution and tested on various others. This analysis aims to understand the performance gap across different image resolutions and determine if FlexiViT's flexible training approach provides greater stability with respect to image size.

We train both standard ViT and FlexiViT on the CUB-200-2011 dataset using four distinct resolution setups: 480, 360, 240, and 120. For standard ViT, the training patch size is fixed at 15, while for FlexiViT, a range of patch sizes (40, 30, 24, 20, 15, 12, 10) is employed. Both training and testing are conducted symmetrically, with query and database images sharing the same resolution. The testing transformation is consistent and deterministic: images are first rescaled so that the shorter edge matches the specified size while maintaining the original aspect ratio, followed by a central crop to capture the primary content.

To ensure a fair comparison between standard ViT and FlexiViT, we use base-sized models and initialize both with weights pretrained on ImageNet-21k for 300 epochs. All other training parameters are kept consistent across the models.

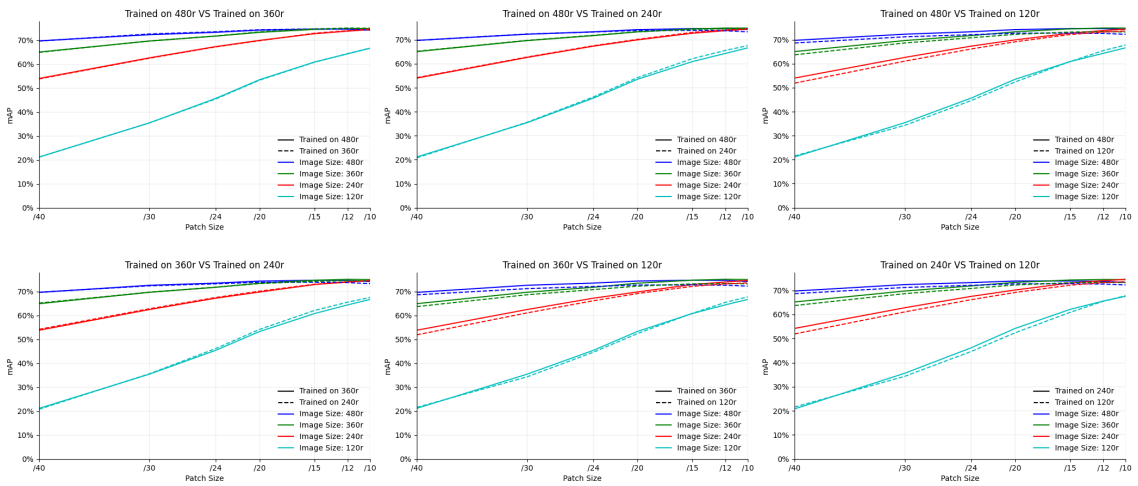


Figure 19: Each subplot compares the symmetric performance of two FlexiViT-B models trained at different resolutions.

In Figure 19, we present the results for FlexiViT. Each subplot illustrates the symmetric testing performance across various combinations of resolution and patch size for two models trained at different resolutions. The first observation is that FlexiViT generally exhibits the expected stability with regard to patch size. However, this stability diminishes to

some extent when trained on smaller resolutions. Secondly, models trained at resolutions of 480, 360, and 240 show no significant differences. This is evidenced by the overlapping performance curves when tested on varying resolutions. A model trained at 480r can be effectively used at 360r without any additional training. The only deviation occurs with the 120r training, which yields slightly inferior performance on larger resolutions, though the gap is minimal and can be considered negligible.

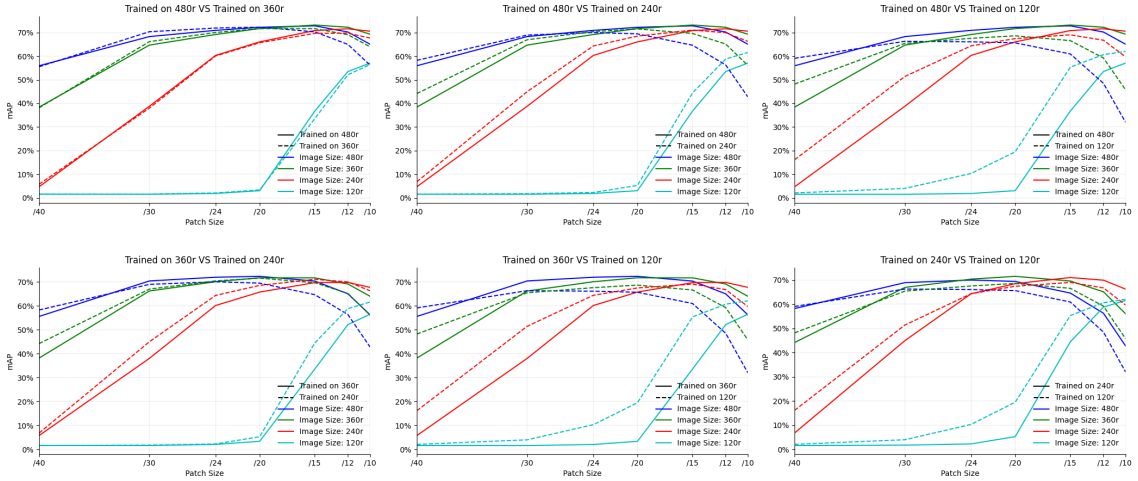


Figure 20: Each subplot compares the symmetric performance of two ViT-B/16 models trained at different resolutions.

In Figure 20, we present the results for a standard ViT model trained with a patch size of 15. It is evident that ViT achieves its best performance around its trained patch size. Contrary to FlexiViT, ViT models trained on resolutions of 480r and 360r exhibit similar performance. However, reducing the image resolution to 240r and 120r leads to a more pronounced drop in accuracy. Notably, the model trained at 120r demonstrates significantly reduced accuracy when evaluated on patch sizes larger than 24. This highlights the compounded negative impact of the inflexibility and increasing patch size in standard ViT.

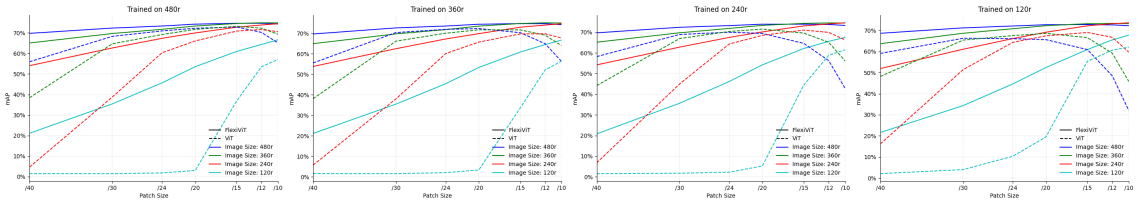


Figure 21: Each subplot compares the symmetric performance of FlexiViT-B and ViT-B/16 models trained at the same resolution.

In Figure 21, we compare the performance of FlexiViT and ViT. Across each tested resolution, FlexiViT consistently outperforms ViT, notably on patch sizes not seen during ViT's training. An important observation is that standard ViT is more sensitive to changes in image resolution compared to FlexiViT. When image size is reduced, FlexiViT experiences less performance decline than ViT.

From the experiments conducted, we draw the following conclusions:

- Using high resolution is beneficial. Processing database images at larger resolutions is advantageous for capturing more image details.
- FlexiViT demonstrates greater stability with image resolution compared to ViT trained on a specific patch size. The PI resize technique in FlexiViT provides flexibility not just in terms of patch size, but also to a certain extent with image resolution.

7 Experiment results with asymmetric metric learning

In the earlier section on symmetric metric learning, we trained several FlexiViT models at multiple image scales to determine the optimal teacher model setup for guiding the student's training. The findings indicated that increasing image resolution and decreasing patch size significantly enhances model performance, albeit at the cost of increased complexity. However, for a teacher model used to extract feature representations from database images, runtime cost is not a primary concern, as it is not involved in test-time retrieval. Therefore, we selected the model trained at an image resolution of 480 as the teacher model. For this model, database images are patchified with a small patch size (15) to yield strong descriptors. We employed knowledge distillation, supplemented with extra data augmentation, to enhance the asymmetric performance of the student model, as detailed in Section 5.

7.1 Asymmetric testing

Prior to delving into the effects of distillation, we evaluated the asymmetric performance by directly applying the teacher model to queries with various combinations of image size and patch size. Furthermore, we conducted a fair comparison with the ResNet50 architecture. For this purpose, a small-sized FlexiViT was chosen due to its runtime complexity, which is comparable to ResNet50, as discussed in Section 6.1. The results are presented in Figure 22.

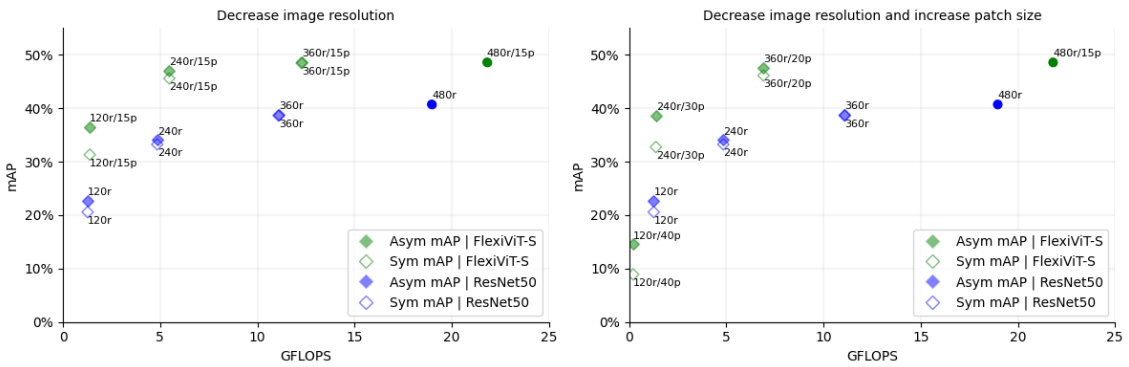


Figure 22: Comparative analysis of asymmetric retrieval performance: The solid circles indicate the configuration for database images. Each diamond shape represents the asymmetric performance and complexity corresponding to queries with specific image resolution and patch size.

The first observation is that FlexiViT offers a flexible means to balance runtime com-

plexity and accuracy. For instance, by adjusting the patch size from 15 to 20 at an image size of 360, FlexiViT surpasses ResNet, achieving 22.55% higher accuracy and 37.64% less complexity.

Secondly, the asymmetric gains in ResNet are not as pronounced: merely increasing the resolution of database images does not significantly enhance ResNet's performance in most cases. In contrast, FlexiViT substantially benefits from asymmetric retrieval in scenarios where there is a large disparity in image size between query and database images. However, when the difference in image size is minimal, the gains are negligible. This aligns with our findings in Section 6.4, where no significant performance discrepancy was observed between FlexiViT models trained at resolutions of 480 and 360.

7.2 Distillation results in asymmetric learning

In this section, we investigate the potential gains of employing knowledge distillation (KD) as a supervised method to aid the training of the student network in resolution-wise asymmetric metric learning. The supervisory information stems from the teacher's representations, not image labels. Section 6.4 revealed a decline in patch size stability when FlexiViT was trained and evaluated at lower resolutions, such as 120, possibly due to significant loss of image details. KD, focused on resolution rather than architecture, presents a viable solution to enhance student performance by leveraging resolution advantages learned from the teacher. Consistent with the approach in the previous section without distillation, a small-sized FlexiViT is again utilized. For comparison, we use FlexiViT trained at three different resolutions: 360, 240, and 120, and conduct asymmetric testing, maintaining a database resolution of 480.

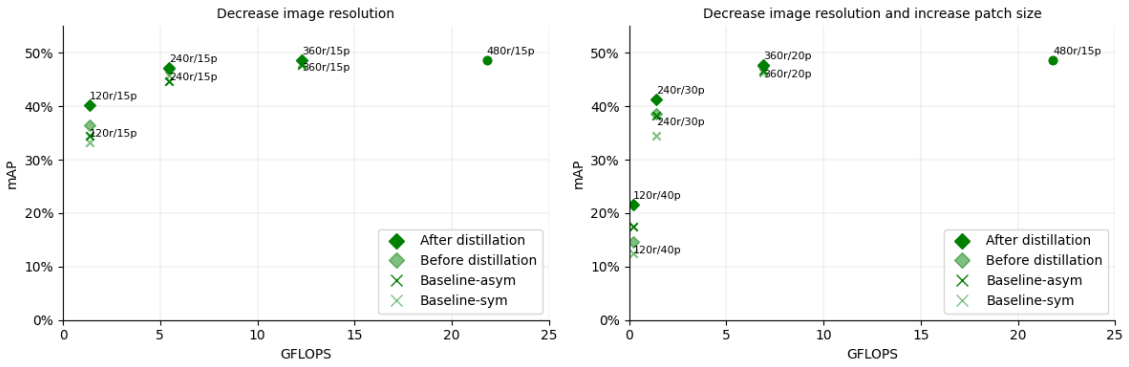


Figure 23: Evaluating the impact of knowledge distillation on student performance: Baseline models, trained at resolutions other than 480, perform asymmetric testing to assess performance simultaneously.

As shown in Figure 23, we apply distillation from a fixed teacher to multiple students with varying complexity levels. This is achieved either by maintaining the same patch size while reducing image size or by simultaneously increasing patch size. The effectiveness of distillation is assessed by comparing it with baseline models. We observe that distillation indeed boosts student performance significantly when there is a substantial resolution gap. However, the situation differs when distilling to a model operating at a resolution of 360. Due to the resolution-stable characteristic of FlexiViT, as concluded in Section 6.4, the teacher and student exhibit similar performance, leaving little room for distillation to be effective.

8 Conclusion

Image retrieval is a rapidly advancing area in computer vision with practical applications in fields such as Google Image search, robotics localization, recommendation systems, and more. The cost of query time and retrieval accuracy are crucial evaluation metrics for different algorithms. Deep neural networks have become a mainstay for extracting potent image descriptors, facilitating effective similarity searching. Recent developments have focused on efficient deep models that strike a balance between performance and accuracy/complexity trade-offs.

In this thesis, I utilize the renowned self-attention-based model, Vision Transformer (ViT), for metric learning. ViT capitalizes on the relationships between image patches, allowing each patch to attend to others and capture higher-level semantics, thus producing embedding vectors that accurately describe image contents. While standard ViT is trained with a fixed patch size, its flexible counterpart, FlexiViT, allows for variable patch sizes without significant performance loss and is thus chosen as the primary model. I investigated the accuracy/complexity trade-off with FlexiViT by varying image resolution and patch size, and compared its performance with one of the most robust CNNs, ResNet, to demonstrate the superiority of transformers in metric learning tasks.

Generally, high image resolution and small patch sizes yield excellent accuracy at the expense of increased model complexity. Knowledge Distillation (KD) is an effective optimization technique that enhances the performance of cost-effective networks by leveraging high-cost model representations for guidance. During symmetric metric learning, I experimented with various FlexiViT setups and selected the best-performing model as the teacher for knowledge distillation. In my work, KD is applied to resolution-based asymmetric metric learning, where query images are resized to a lower resolution than database images. Experiments in the last section have demonstrated the indispensable benefits of KD for Vision Transformers in certain scenarios.

References

[1] Mateusz Budnik and Yannis Avrithis. Asymmetric metric learning for knowledge transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[2] Artem Babenko and Victor Lempitsky. Efficient indexing of billion-scale datasets of deep descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[3] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128, 2011.

[4] Yannis Kalantidis and Yannis Avrithis. Locally optimized product quantization for approximate nearest neighbor search. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[5] Yantao Shen, Yuanjun Xiong, Wei Xia, and Stefano Soatto. Towards backward-compatible representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[6] Rahul Duggal, Hao Zhou, Shuo Yang, Yuanjun Xiong, Wei Xia, Zhuowen Tu, and Stefano Soatto. Compatibility-aware heterogeneous visual search. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10723–10732, 2021.

[7] Qiang Meng, Chixiang Zhang, Xiaoqiang Xu, and Feng Zhou. Learning compatible embeddings. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 9939–9948, 2021.

[8] Binjie Zhang, Yixiao Ge, Yantao Shen, Shupeng Su, Fanzi Wu, Chun Yuan, Xuyuan Xu, Yexin Wang, and Ying Shan. Towards universal backward-compatible representation learning. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1615–1621, 2022.

[9] Vivek Ramanujan, Pavan Kumar Anasosalu Vasu, Ali Farhadi, Oncel Tuzel, and Hadi Pouransari. Forward compatible training for large-scale embedding retrieval systems. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19386–19395, 2022.

[10] Hui Wu, Min Wang, Wengang Zhou, Houqiang Li, and Qi Tian. Contextual similarity distillation for asymmetric image retrieval. In *CVPR*, 2022.

[11] Pavel Suma and Giorgos Tolias. Large-to-small image resolution asymmetry in deep metric learning. *arXiv preprint arXiv:2210.05463*, 2022.

[12] Karsten Roth, Timo Milbich, Samarth Sinha, Prateek Gupta, Bjorn Ommer, and Joseph Paul Cohen. Revisiting training strategies and generalization performance in deep metric learning. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 8242–8252. PMLR, 2020.

[13] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[14] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *International Conference on Learning Representations*, 2017.

[15] Zehao Huang and Naiyan Wang. Like what you like: Knowledge distill via neuron selectivity transfer. *arXiv preprint arXiv:1707.01219*, 2017.

[16] Bharat Bhusan Sau and Vineeth N Balasubramanian. Deep model compression: Distilling knowledge from noisy teachers. *arXiv preprint arXiv:1610.09650*, 2016.

[17] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.

[18] Zheng Xu, Yen-Chang Hsu, and Jiawei Huang. Training shallow and thin networks for acceleration via knowledge distillation with conditional adversarial networks. 2018.

[19] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *CVPR*, 2019.

[20] Taojiannan Yang, Sijie Zhu, Chen Chen, Shen Yan, Mi Zhang, and Andrew Willis. Mutualnet: Adaptive convnet via mutual learning from network width and resolution. In *European Conference on Computer Vision (ECCV)*, 2020.

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

[22] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR, abs/1409.0473*, 2014.

[23] Yoon Kim, Carl Denton, Luong Hoang, and Alexander M Rush. Structured attention networks. In *International Conference on Learning Representations*, 2017.

[24] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *ICML*, 2018.

[25] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

[26] Mark Chen, Alec Radford, Rewon Child, Jeff Wu, and Heewoo Jun. Generative pretraining from pixels. In *ICML*, 2020.

[27] Jean-Baptiste Cordonnier, Andreas Loukas, and Martin Jaggi. On the relationship between self-attention and convolutional layers. In *ICLR*, 2020.

[28] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[29] L. Beyer, P. Izmailov, A. Kolesnikov, M. Caron, S. Kornblith, X. Zhai, M. Minderer, M. Tschannen, I. Alabdulmohsin, and F. Pavetic. Flexivit: One model for all patch sizes. *arXiv preprint arXiv:2212.08013*, 2022.

[30] Lmnnb: Two-in-one imbalanced classification approach by combining metric learning and ensemble learning - scientific figure on researchgate. `https://www.researchgate.net/figure/Working-mechanism-of-metric-learning-The-metric-learning-method-aims-to-find-a_fig3_355201453`, 2022.

[31] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 1097–1105, Lake Tahoe, NV, USA, December 2012.

[32] Linqin Cai, Sitong Zhou, Xun Yan, and Rongdi Yuan. A stacked bilstm neural network based on coattention mechanism for question answering. *Computational Intelligence and Neuroscience*, 2019:1–12, 08 2019. doi: 10.1155/2019/9543490.

[33] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[34] Mingxing Tan and Quoc V Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv preprint arXiv:1905.11946*, 2019.

[35] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[36] Vggnet-16 architecture: A complete guide — kaggle. `https://www.kaggle.com/code/blurredmachine/vggnet-16-architecture-a-complete-guide`, 2021.

[37] Transformers from scratch. `https://peterbloem.nl/blog/transformers`, 2019.

[38] Wikipedia contributors. Precision and recall — Wikipedia, the free encyclopedia, 2024. URL `https://en.wikipedia.org/w/index.php?title=Precision_and_recall&oldid=1194043485`. [Online; accessed 9-January-2024].

[39] Knowledge distillation. `https://medium.com/vafion/knowledge-distillation-9a0bf514fbeb`, 2020. [Online; accessed 10-January-2024].

[40] Peter Welinder, Steve Branson, Takeshi Mita, Catherine Wah, Florian Schroff, Serge Belongie, and Pietro Perona. Caltech-ucsd birds 200. Technical report, California Institute of Technology, 2010.

[41] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019.

[42] Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[43] Andrew Brown, Weidi Xie, Vicky Kalogeiton, and Andrew Zisserman. Smooth-ap: Smoothing the path towards large-scale image retrieval. In *ECCV*, 2020.

[44] Fatih Cakir, Kun He, Xide Xia, Brian Kulis, and Stan Sclaroff. Deep metric learning to rank. In *CVPR*, 2019.

[45] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.

[46] S. Jialin Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, Oct 2010.