Master of Science in Informatics at Grenoble
Master Informatique
Specialization Data Science

Deep learning with Siamese networks for instance search or identification

Matthias Kohl

June 20, 2017

Research project performed at LIG - MRIM

Under the supervision of:
Georges Quénot, Jean-Pierre Chevallet

June 2017

Abstract

Our research is part of the GUIMUTEIC project, which aims at building an augmented audio-guide for museums or tourist sites. In particular, our goal is to develop a system capable of identifying a piece of art using solely the visual clues of an image provided by a camera, in order to provide further information about the piece of art to the user of the audio-guide. We focus on instance retrieval approaches based on deep learning. We make three main contributions: we analyze advantages and limitations of the current state of the art in instance retrieval, we propose a novel approach based on a fully convolutional network (FCN), and we evaluate the performance of these different approaches in depth. We show that the proposed architecture produces state-of-the-art results, is fast to run and reasonably fast to setup for a new museum or tourist site. The evaluation is carried out on the CLICIDE and GaRoFou datasets, both created in the context of the project, containing images of pieces of art from museums in Grenoble and Lyon.

Acknowledgement

This work has been carried out in the context of the Guimuteic project funded by Fonds Européen de Développement Régional (FEDER) of région Auvergne Rhône-Alpes.

I would like to express my sincere gratitude to Georges Quénot, Jean-Pierre Chevallet and Maxime Portaz for their invaluable assistance and comments in reviewing this report.

Publication

The work developed as part of this research project will be submitted as a paper to the second international workshop on egocentric perception, interaction and computing at ICCV 2017.

Résumé

Notre travail de recherche fait partie du projet GUIMUTEIC qui doit produire un audio-guide augmenté pour des musées ou des sites touristiques. En particulier, notre objectif est de développer un système d'identification automatique de l'œuvre d'art contemplé par un utilisateur de l'audio-guide, en utilisant uniquement une image issue d'une caméra. Notre recherche se focalise sur les approches d'apprentissage profond pour l'identification d'instances. Nous apportons trois contributions majeures : une analyse des avantages et inconvénients des solutions d'identification d'instances de l'état de l'art, la proposition d'une nouvelle approche basée sur un réseau entièrement convolutionnel (FCN), et l'évaluation de la performance de ces différentes approches. Nous montrons ainsi que l'architecture proposée donne des résultats performants tout en étant rapide à l'interrogation et raisonnablement rapide à mettre en place pour une nouvelle collection de musée ou site touristique. L'évaluation se fait sur les deux corpus d'images CLICIDE et GaRoFou, tous deux produits dans le cadre du projet, et contenant des images d'œuvres d'art d'un musée de Grenoble et de Lyon.

Contents

— 1 —
Introduction

1.1 Motivation

The research presented here is motivated by the Guimuteic project, a collaborative project between industry and LIG. The aim is to develop a smart audio-guide for touristic or cultural sites.

In practice, the final product should offer an augmented reality interface to the user, with information about the object or objects the user is looking at.

One part of the development consists in finding ways of identifying objects the user is looking at. There are multiple possibilities, for example based on the geo-localization of the user and other sensors. In our research, we focus on the recognition of objects based only on visual clues.

1.2 Research problem

More specifically, we are interested in the following problem: we are given a collection with reference images for each object, or instance, to be recognized. The task is to develop a system that, given an image of one of the instances, can decide which instance the image represents. We assume that there is little variation between the images of a given instance, as they all represent the same object. We will refer to this problem as instance retrieval in the following.

There are a two problems similar to instance retrieval: image classification and image retrieval. in image classification, we are given a collection of images where each image is assigned a semantic class, such as *dog* or *fridge*. The task is to develop a system that, given an image, decides which class the image represents. This problem is similar to instance retrieval if we consider each instance to be a separate class. However, two main differences exist: in classification, the images usually represent semantic classes and contain more variability between them than in the instance retrieval problem, where all images of a class represent the exact same object. A consequence of this is that in classification, one of the goals is to find a pattern shared by each semantic class. This is not as important when we consider a single object to represent a class.

The second problem similar to instance retrieval is image retrieval. In image retrieval, we are given a collection of reference images and a query image. We aim to rank the reference images by similarity to the query image. Usually, in image retrieval, reference images are not labeled or loosely labeled and many images are irrelevant to the query image. The challenge

is to rank the most similar images on top. Instance retrieval is related to image retrieval in that we aim to develop a notion of similarity between images. However, in instance retrieval, we do not care about the rank of the returned images, since we only consider the highest ranked image, which should represent the instance to retrieve.

Finally, to allow fast comparison between images, our goal is to develop a system that provides a single descriptor for each image, which should be small enough to be easily stored for all reference images and be fast to compute. Two images should be compared according to a distance metric, which should also be fast to compute. This requirement is important, since a possible device used as an augmented audio-guide will record many images per second. Thus, to be responsive, it should be possible to run the system in real-time or close to real-time.

1.3 Challenges

For all of the problems described above, deep learning approaches based on convolutional neural networks (CNNs) have recently obtained the state-of-the-art results. One of the drawbacks of deep CNNs is that they require large amounts of data with high variability to be trained.

In our research problem in particular, we do not have large enough amounts of data available to fully train a deep CNN. However, we aim to develop a system that can learn a descriptor specific to the reference images, as the system only needs to recognize and differentiate between instances of that particular dataset. Thus the system should be tuned to the dataset. Since we usually have less than ten images per instance, this is an important challenge to overcome.

Another challenge comes from the nature of the datasets: we were not able to find similar datasets in the literature, with only a few, clean images, for each instance and many instances. Common datasets in image retrieval like Oxford5k, Paris6k or Holidays [29, 30, 17] contain many images per instance, as well as a lot of noise, as they are used to evaluate whether a system can robustly retrieve the correct images out of a set of weakly related images. This means that the approaches used for image retrieval datasets may not work as well for instance retrieval.

1.4 Contributions

We made the following three major contributions:

1. We analyze existing approaches to image retrieval, classification and determine their shortcomings in our problem setting.

2. Based on these shortcomings, we propose a novel approach to improve results in our problem setting. This approach is detailed in Section 3.4. It is based on extracting region proposals from images through a fully convolutional network (FCN), combined with the previous state-of-the-art architecture in image retrieval.

3. We evaluate this novel approach and compare with previously proposed approaches.

— 2 —
State of the art

2.1 SIFT and bag-of-words

Until recently, the state of the art in image retrieval and matching was based on the idea of bag-of-words [29, 26]. The approach was introduced to image retrieval by Sivic and Zisserman [44] as well as Csurka et al. [7].

The idea behind the approach is to represent an image as a histogram, or collection of frequencies, of visual words. The visual words should be small, representative patches of the images in the dataset. Usually, these visual words are obtained in multiple steps. First, local features are extracted and encoded. The most common feature used is SIFT [24], a 128-dimensional vector representing scale-invariant features. Then, the features are extracted for all images and clustered into clusters of similar features. The representative for each cluster is the center of the cluster, i.e. the mean of all features falling into that cluster. Each image can then be represented as a histogram of the occurrences of these representative features. This histogram forms the image descriptor, which has as many dimensions as there are representative features and clusters.

Finally, for image classification, a classifier can be learned from the descriptors of all images in the dataset. On the other hand, for image retrieval or matching, the descriptors can be directly matched against each other, based on some similarity measure. In both cases, it can be useful to use a different similarity measure than the euclidean distance in the original descriptor space. For classification, this is done by employing an SVM classifier with a non-linear kernel [42]. Among the most popular kernels are the radial basis function [40] and the chi-squared function [48].

Until recently, this approach has obtained state-of-the-art results for image retrieval tasks [26].

2.2 Fisher vectors (FV)

The Fisher vector representation is based on the Fisher kernel, as described by Jaakkola and Haussler [16]. The main idea in their work is to derive a kernel function for discriminative classification, where the kernel function is derived from a generative probability model. For this, the log-likelihood with respect to a parameter $\log P(X|\theta)$ is considered, where X is the data and θ the parameter.

The Fisher vector is based on the gradient of this log-likelihood which essentially captures how much the parameter contributes to the generation of a sample X. This principle has been

applied to images as well by Sanchez et al. [39], in the context of image classification. Here, the data (or samples) are the local descriptors of patches of images and the parameters to be estimated are the parameters of a Gaussian Mixture Model (GMM) of these descriptors. This means that we consider the descriptor of an image as a combination of local descriptors, each being independent of all others, with each local descriptor contributing to the global descriptor through a GMM. This GMM can have any number of parameters, which can be learned through the expectation maximization algorithm. The Fisher vector is the gradient of the log-likelihood of this GMM with respect to the parameters. Thus, it represents the combination of deviations from the generative model of all local descriptors of the image, for each parameter of the model.

Another important property is that the Fisher vector embeds each local descriptor into a space of different dimension, which consists of the parameters of the GMM: weights, means and co-variances of the Gaussians. The dimensionality of the Fisher vector is thus ultimately dependent on the number of Gaussians chosen for the GMM.

It can be shown that this representation represents a kernel for a kernel machine and can thus be used very efficiently by linear classifiers learning their parameters in the Fisher kernel space.

While the Fisher vector representation has a high dimensionality, it has been shown that it can be compressed efficiently [39, 28] and can be applied to large-scale image retrieval systems [28].

2.3 VLAD

Another important descriptor used in image retrieval systems is the vector of locally aggregated descriptors or VLAD. It was introduced by Jegou et al. [18] and is related to both the bag-of-words and the Fisher vector representations described in Sections 2.1 and 2.2.

As with the bag-of-words approach, a visual vocabulary is formed using the k-means clustering algorithm. Then, for each visual word w (one of the k centers of the output of k-means) and each local descriptor x associated with w (where w is the closest center), the difference $x - w$ is added to the VLAD representing w.

Hence, VLAD is a sum over the differences between all local descriptors and their associated visual word. This is similar to the Fisher vector representation, which represents a sum of the deviations of local descriptors from the generative model, which can be seen as consisting of the visual vocabulary.

Jegou et al. [18] show that the VLAD representation has many advantages over previous representations. For one, it achieves state-of-the-art results on the Holidays dataset [17]. Furthermore, it can be compressed efficiently using principal component analysis and still achieve promising results, making it faster to compute than the Fisher vector representation, even when compressed as described in Section 2.2.

2.4 Deep learning with CNNs

Starting with the results of AlexNet for image classification in the 2012 ImageNet challenge [21, 37], image classification tasks have been dominated by CNNs, learned using large amounts of data.

A general trend in image related tasks is to move to an end-to-end approach, where the final objective is directly optimized using gradient descent and the gradient is back-propagated to all

previous parts of the system. In contrast, the bag-of-words model requires a choice of features (SIFT, ORB [34], ...), a choice of the method for clustering features (k-means), a choice of the classifier (AdaBoost [8], SVM, ...), as well as a choice of the kernel if an SVM classifier is used.

Using a CNN, features are extracted at a low abstraction level by the first convolutional layers, then higher level features are formed by combining low level features from the previous layers. Finally, high-level features are combined into a classifier by linear layers. All layers are then optimized together, usually through three main steps. First, a loss function evaluates the loss made by the last layer of the network. Then, the back-propagation algorithm [36] exploits the chain-rule of derivatives to propagate the gradients of the errors made by each layer from the last layer to the first layer of the network. Finally, a stochastic gradient descent algorithm [4, 3, 35] is used to optimize the parameters of all layers with respect to the loss. The advantage of this approach is that the features are learned at all abstraction levels at once. Another advantage of neural networks is their modularity. It allows us to easily transfer the lower level features learned from a large dataset to a smaller dataset, where there may not be sufficient amounts of data to efficiently learn lower level features.

The following sections describe the high-level architecture of the most widely used CNNs in classification and image retrieval.

2.4.1 AlexNet

AlexNet [21] contains five convolutional layers and three linear layers. The first convolutional layer has a large kernel and stride to quickly increase the receptive field of the consequent filters. The first two convolutional layers are followed by a pooling layer to further increase the receptive field.

Finally, an important improvement for AlexNet to avoid over-fit is the addition of dropout layers [13] before the two first linear layers.

Apart from that, AlexNet is a simple adaptation of LeNet by LeCun et al. [22] for the ImageNet challenge.

2.4.2 VGG

The VGG architecture, introduced by Simonyan and Zisserman [43] is the first to use exclusively 3×3 convolutional kernels. This means that the receptive field of the first filters is much smaller, and thus many more layers are needed, as well as more pooling layers. The most popular VGG architectures are VGG-16 and VGG-19, having 16 and 19 layers in total respectively.

Using smaller kernels in convolutions and consequently more layers allows to reduce the number of trainable parameters used by the network, while increasing the number of non-linear layers. This was shown to allow better generalization properties, and thus less over-fit [43].

2.4.3 ResNet, Inception, DenseNet

The idea of increasing the number of layers is further developed by the ResNet, DenseNet and Inception architectures. However, naively increasing the number of layers leads to the problem of vanishing gradient [12].

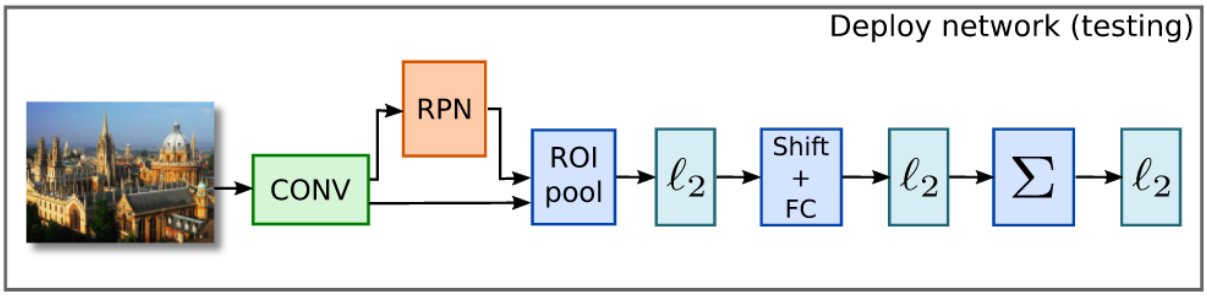


Figure 2.1 – Architecture of a CNN network for image retrieval, developed by Gordo et al. [9]

All of these networks overcome the vanishing gradient problem by introducing skip connections, where the output of a previous layer is added to the output of a layer, skipping some number of layers in between.

The ResNet architecture by He et al. [11] always uses blocks of 3 layers along with a skip of those 3 layers. The smaller types of ResNet use blocks of 2 layers.

The Inception architecture by Szegedy et al. [45] uses a combination of skip connections of different length and different types.

Finally, DenseNet by Huang et al. [14] takes the skip connection idea one step further: the input of each layer is dependent on the output of all n previous layers to some extent.

All of the very deep architectures use batch norm layers after the convolutional layers to further reduce over-fit [15]: a batch norm layer simply normalizes the features over each batch, and then applies a learn-able scaling and shifting. During training, it also produces estimates of the over-all mean and variance of the input of the layer over the whole dataset. These estimates of the global mean and variance are used during evaluation in order to normalize the data after each block of convolutional and pooling layers.

2.5 Image retrieval using CNNs

For image retrieval, the current state of the art is set by Gordo et al. [9]. It is based on an end-to-end approach. The goal is to learn a global descriptor for images that is well suited for comparing images.

Figure 2.1 shows the detailed architecture used. It first extracts the convolutional features of a pre-trained CNN. Then, a Region Proposal Network (RPN) [33] is used to extract the regions of interest. For each region of interest, a shifting layer, followed by a linear layer are used to reduce the dimensionality of the descriptor. The final descriptor consists of a normalized sum of the region-wise descriptors. This network can be learned end-to-end and the obtained descriptor achieves state-of-the-art results, which can be even further improved by using query expansion and database side feature augmentation (described in detail in Section 3.5).

Gordo et al. [9] found that using very deep networks outperforms the shallower networks. The state-of-the-art results are thus set by a very deep ResNet-50 architecture, using 50 layers.

2.5.1 Siamese networks and triplet loss

Figure 2.2 shows the architecture used to train the CNN that produces a descriptor for images, as can be seen in Figure 2.1.

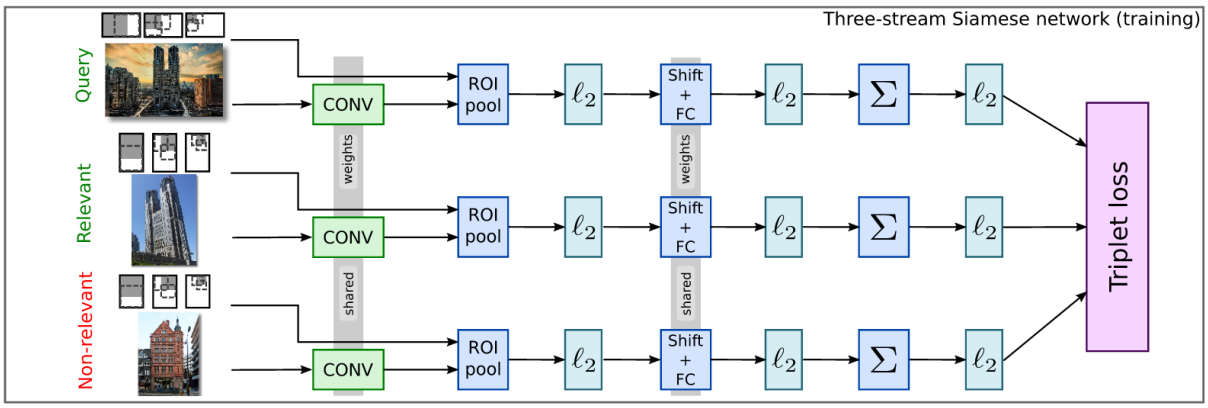


Figure 2.2 – Siamese architecture of a CNN for image retrieval used in training [9]

The major difference is that during training, a Siamese architecture is used: the CNN is evaluated with multiple images, using the same weights for each. Then, a combined loss is obtained from the descriptors. Finally, the loss is back-propagated once through all streams of the network, since shared weights are used for all images.

The Siamese architecture was first proposed by Chopra et al. [5] in the context of facial recognition. In this paper, pairs of images are used and a loss was designed to move the descriptors of a pair of images together in the descriptor space, if the pair represents the same face, while separating the descriptors if they represent different faces.

In the case of the architecture presented in Figure 2.2, three images are evaluated and a triplet loss is used. This triplet loss was shown to be more robust during training than a loss using pairs and has been used to set the state of the art in facial recognition tasks by Schroff et al. [41]. The triplet loss was first introduced by Weinberger et al. [50] as a way of learning the best suited distance metric in a k-nearest neighbor classification problem through gradient descent. This is the reason why the triplet loss was chosen for image retrieval: the goal is to learn a metric which allows to discriminate couples of images containing the same instance from couples of images containing different instances, using the descriptors obtained from the deep CNN. Section 3.2.1 describes this loss in more detail.

Finally, Gordo et al. made further contributions, detailed in their follow-up paper [10]. For one, an important step is to use a suitable dataset. Datasets like Oxford5k or Paris6k are too small for a network to learn a distance metric applicable to any type of image. Instead, the state of the art uses the much larger Landmarks dataset, created by Babenko et al. [2] for training. This dataset contains almost 200000 images in total, of around 600 different landmarks, mostly buildings.

Another important step is the cleaning of the dataset. The Landmarks dataset contains a lot of noise and weak labels: many images do not represent the building they are labeled with. Gordo et al. [10] use a meticulous cleaning process: the main idea is to keep only the largest clusters of matching objects for each of the landmarks. The implementation is based on extracting SIFT and Hessian descriptors, matching them and verifying matches with an affine transformation model. This cleaning process is used in order to annotate the cleaned images with regions of interest as well.

The cleaning requires a lot of processing, but is only done once for the dataset. The cleaned dataset is still fairly large, containing 49000 images and 586 instances of landmarks.

Finally, an important step in training a network on image matching using the triplet loss is

the selection of triplets. Multiple mechanisms have been proposed for this purpose [41, 10], which are further detailed in Section 3.2.3.

2.5.2 Limitations and goals

The main limitation of the state of the art with respect to instance retrieval is that it aims at producing a general descriptor for comparing any type of image. This means that it is most effective when trained on large datasets, with many samples per instance. For datasets containing images from museums or tourist sites, this is usually not applicable, as there are usually only very few images available for each instance. While there are large datasets, these datasets usually contain a large number of instances as well. This makes it difficult to directly apply the state of the art on our datasets.

On the other hand, it also means that the state of the art in image retrieval aims at avoiding over-fit at all costs: the learned descriptor and metric should be able to provide a good metric for any kind of image. While this is usually desired in machine learning, in our case, one goal is to provide a metric specifically designed to discriminate between the images of the given dataset. This is because an audio-guide in a museum can be fine-tuned for that museum, as long as the fine-tuning is computationally cheap enough. So we do not care if a given audio-guide cannot identify images of multiple museums, as long as it can identify images of one museum with high precision.

Another limitation of the state of the art is that it requires a tedious process to clean the dataset and annotate it with regions of interest. This is not suited to museum datasets which are usually very clean to begin with, by the nature of the dataset. Plus, the notion of region of interest is usually not relevant for this type of image: which part of a painting or piece of art contains the semantically relevant information is not obvious.

Finally, an evaluation of the state-of-the-art network [9] was carried out. We used the published weights obtained in training this CNN on the Landmarks dataset. Figure 3.2 shows examples of images in our dataset that were correctly matched with high confidence as well as images that were incorrectly matched. From these images, we make the following observations in Section 3.3.2:

1. Images that are very similar visually are well matched

2. Images at different scales are not well matched

Thus, a limitation of the current state of the art is matching images, where the object of interest is at different scales.

The goal for our research is thus to try to overcome the limitations of the current state of the art by proposing a descriptor which can be fine-tuned to each dataset, is robust to differences in scale and does not require annotations of regions of interest.

— 3 —
Contributions

In the last chapter, we have presented the state of the art and related work in image classification as well as image retrieval, and their limitations when applied to the instance search problem. In order to overcome these limitations, we proceed as follows.

We first look at the instance retrieval problem as a classification problem, with each instance representing a class. We use transfer learning in order to fine-tune a deep CNN pre-trained on ImageNet to this new task. This approach is detailed in Section 3.1.

We then consider the instance retrieval problem as an image retrieval problem. We simplify the state-of-the-art architecture by removing region of interest pooling and apply this simplified Siamese architecture to our research problem. This approach can be seen as fine-tuning a network on image matching. This is detailed in Section 3.2.

In Section 3.3, we identify the limitations of the state of the art as well as the network fine-tuned on classification in more detail. We also show that good region descriptors can be obtained directly from a network fine-tuned on classification.

Based on the observations and analysis from Section 3.3, we propose a novel approach, which combines fine-tuning a network on classification and fine-tuning it on image matching using a Siamese architecture. This approach is presented in Section 3.4.

Finally, in Section 3.5 we discuss additional improvements for descriptors used in image retrieval and propose an adaption of database feature augmentation to the instance retrieval problem.

3.1 Fine-tuning a CNN

In image classification using deep CNNs, we rely on large amounts of data, with high inter-class variability in order to infer the optimal values for millions of parameters. These parameters allow the CNN to describe the patterns that an image consists of, from the lowest abstraction layer to the highest. In the instance search problem, as we want to identify a particular instance, the variability of examples of that instance is less important. This means the data is usually not sufficient to fully optimize a network like ResNet.

In order to overcome this problem, a first possible approach is to fine-tune a classification network to instance classification. This means that we consider each instance as a class, and optimize using a cross-entropy loss, typical for classification.

$$\mathscr{L} = \frac{1}{N}\sum_{i=1}^{N} -y_i \log \hat{y}_i - (1-y_i)\log(1-\hat{y}_i) = \frac{1}{N}\sum_{i=1}^{N} CE_i \tag{3.1}$$

Equation 3.1 describes the cross-entropy loss for a given instance. N is the number of samples, y_i an indicator variable taking the value 1 if sample i belongs to the given instance and 0 otherwise, and $\hat{y}_i$ is the predicted probability that sample i belongs to the given instance. In the following, we denote the cross-entropy loss for the i-th sample as CE_i.

Fine-tuning a CNN for classification on different data has been studied intensively by Yosinki et al. [51]. In particular, their study shows that it is only important to fine-tune the neurons of higher layers of a CNN. Furthermore, they show that fine-tuning can increase generalization even in the fine-tuned model. Both of these properties are desirable in our task, because they reduce the memory and time needed to re-train a network, as well as the need for a large dataset used for fine-tuning.

The specific considerations we take with regards to fine-tuning are described in the following sections.

3.1.1 Data augmentation

Our datasets, presented in Section 4.1, contain an average of less than 10 samples per instance. This makes training a typical CNN model difficult, even when fine-tuning for classification. Data augmentation consists of randomly applying affine transformations, color perturbations and other random transformations. It can help in obtaining a more robust network, with higher geometry invariance than if the network is trained without data augmentation.

Since we specifically identified an issue related to different scales in images in Section 2.5.2 and Section 3.3.2, it is natural to augment the data by randomly scaling the images in both dimensions.

The lack of geometry invariance of the model can also be reduced by randomly rotating and flipping the images, thus we perform this type of data augmentation throughout our experiments.

For data augmentation in order to fine-tune a CNN, we use the following values in our experiments:

1. Rotation: any angle is chosen with the same probability.

2. Scaling: the scaling factor is chosen independently for each dimension in the range $[0.75, 1.25]$.

3. Flipping: with probability 0.5, images are horizontally flipped.

3.1.2 Transfer learning

The modularity of a CNN means that we can easily transfer the weights from a pre-trained model, and only re-train the highest abstraction layers. Specifically, we re-train all linear layers in the model, representing the highest-level layers.

We also re-train the highest level convolutional layers, since our datasets contain many visually different images as compared to the ImageNet dataset used for pre-training the models. For the AlexNet architecture, we choose to re-train all layers above and including the last convolutional layer. For a ResNet architecture, we re-train all layers above and including the third to last block of convolutional layers. This contains the nine highest convolutional layers in total.

The results obtained by transfer learning are listed in Table 5.1 in chapter 5, abbreviated by *FT*.

3.2 Simplified Siamese architecture

We propose a simplified Siamese architecture inspired by the methodology of Gordo et al. [9]. As shown in Figure 2.2, the previous state of the art is set by a Siamese architecture using the triplet loss.

3.2.1 Triplet loss

$$\mathscr{L} = \sum_{i=1}^{N} \frac{1}{2} \max(0, \|x_i^a - x_i^p\|_2^2 - \|x_i^a - x_i^n\|_2^2 + 2m) \tag{3.2}$$

Equation 3.2 describes the triplet loss for N triplets. Each triplet is composed of an anchor image with descriptor x^a, a positive/relevant image with descriptor x^p and a negative/non-relevant image with descriptor x^n. A margin m is added to the loss, which describes the distance at which two images are considered similar enough to be relevant with respect to each other. Throughout our experiments, this margin is set to a value of $m = 0.1$.

In image retrieval, many approaches use $l2$-normalized descriptors [17, 18, 46, 9] since this improves performance and facilitates comparison between two descriptors. We choose to use $l2$-normalization as well, which enables us to re-write the triplet loss.

For normalized descriptors x and y, the squared distance $\|x-y\|_2^2$ is closely related to the cosine similarity and the dot product through the relationship derived in Equation 3.3. One consequence of this equation is that comparing descriptors based on dot product, cosine similarity and squared distance is closely related: a high value of the dot product, high value of cosine similarity and a low value of the squared distance are all equivalent up to a constant factor and constant shift.

$$\begin{aligned} &\forall x, y \in \mathbb{R}^n, \|x\|_2 = 1, \|y\|_2 = 1 \\ &\|x-y\|_2^2 \;\; = (x_1 - y_1)^2 + \cdots + (x_n - y_n)^2 \\ &\qquad\quad = (x_1^2 + \cdots + x_n^2) - 2(x_1 y_1 + \cdots + x_n y_n) + (y_1^2 + \cdots + y_n^2) \\ &\qquad\quad = 2 - 2xy = 2 - 2\cos(x, y) \end{aligned} \tag{3.3}$$

Another consequence of Equation 3.3 is that we can re-write the triplet loss using the dot product, as shown in Equation 3.4. This equation represents the version of the triplet loss used in our experiments. This means all descriptors must be $l2$-normalized before the loss is evaluated.

$$\mathscr{L} = \sum_{i=1}^{N} \max(0, x_i^a x_i^n - x_i^a x_i^p + m) \tag{3.4}$$

3.2.2 Simplified architecture

We simplify the architecture shown in Figure 2.2 by removing the region proposal network and region of interest pooling. We justify in Section 3.3.1 why these components are not relevant for our datasets and research problem in general. Except for this change, the architecture is kept exactly as described by Figures 2.1 and 2.2.

Just as suggested by previous authors [9, 41], when training the Siamese model, it is initialized with weights of a model fine-tuned on classification on the same dataset. Here, this fine-tuning is performed as described in Section 3.1 above.

3.2.3 Triplet selection

As noted by previous authors, when using the triplet loss, it is crucial to choose the best triplets during training in order to obtain convergence. In particular, many triplets are irrelevant and do not produce any loss since they are *too easy* for the network.

Hence, the first idea is to choose the hardest triplets, as proposed by Schroff et al. [41]. However, as they show in their paper, this can lead to a collapsing model early on in training so they choose *semi-hard* triplets instead. Semi-hard triplets are obtained as follows: use all possible positive couples of images: couples of images from the same instance. For each positive couple, choose the hardest negative that is easier than the positive couple. Hard and easy are defined by the dot product between the descriptors of the images: a high value of the dot product for images of the same instance represents an easy positive couple, a high value of the dot product for images of different instances represents a hard negative couple. The value of all dot products are determined before each pass over the whole training data during training, for all couples of images.

A different triplet selection mechanism was proposed by Gordo et al. [10]. First, calculate the values of dot products for all couples of images before each pass over the training data, as in the method described above. Second, for each image, choose the n easiest positive images and the m hardest negatives. Then, calculate the loss for all possible combinations and use the o triplets with the highest loss. This method probably eliminates some noise when choosing the easiest positive couples, for images that are labeled as being the same instance but are not visually similar. However, in experiments, we found that this method does not perform well for datasets with few images per instance, since we either have to choose n as very low or we end up choosing all positive couples for most instances, just like in the *semi-hard* selection.

Hence, in our experiments, we choose the semi-hard triplet selection for the first two passes over the dataset, after which we only choose the hardest negatives for all positive couples.

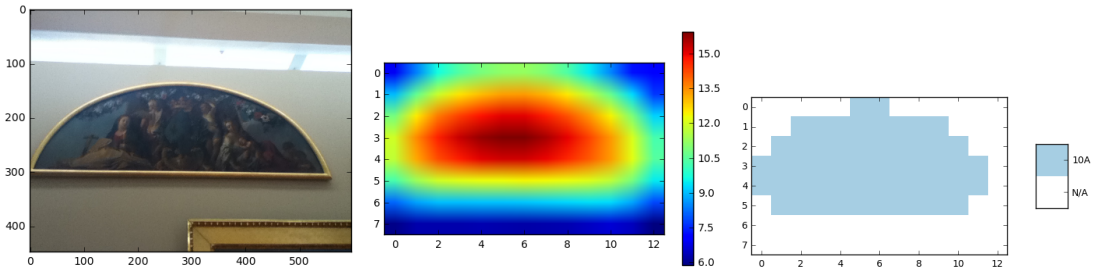
The results obtained by this simplified Siamese architecture, using semi-hard triplet selection, are listed in Table 5.1 in chapter 5, abbreviated by *SS*.

3.3 Region of interests and object localization

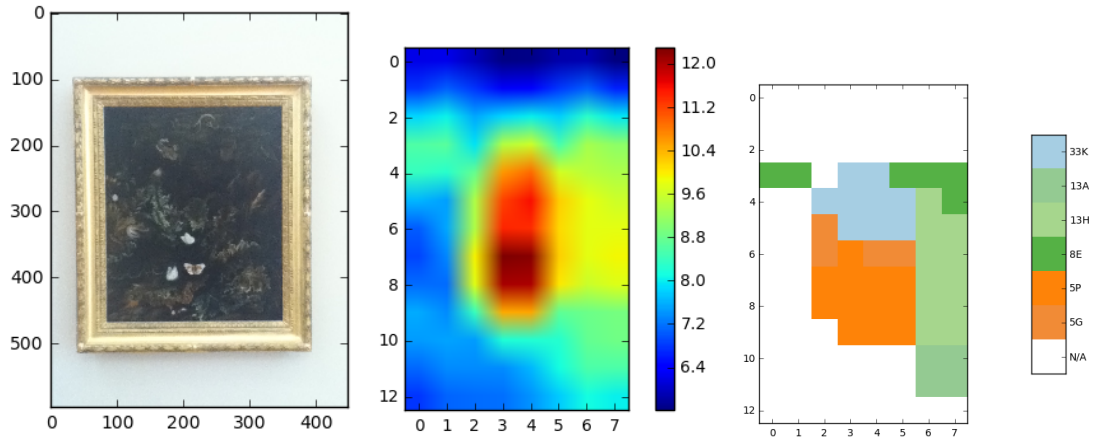
3.3.1 Identifying regions of interest

Previous approaches in image retrieval [10, 38, 46] usually deal with regions of interest in one way or another. The idea is that in most cases, only certain parts of each image can be useful for comparison with other images. In addition to this, cropping images at their regions of interest can help with differences in scale of the images to compare: if a building is visible only in a small part of an image, cropping the image at that part and then re-scaling the part should set the building at a normalized scale.

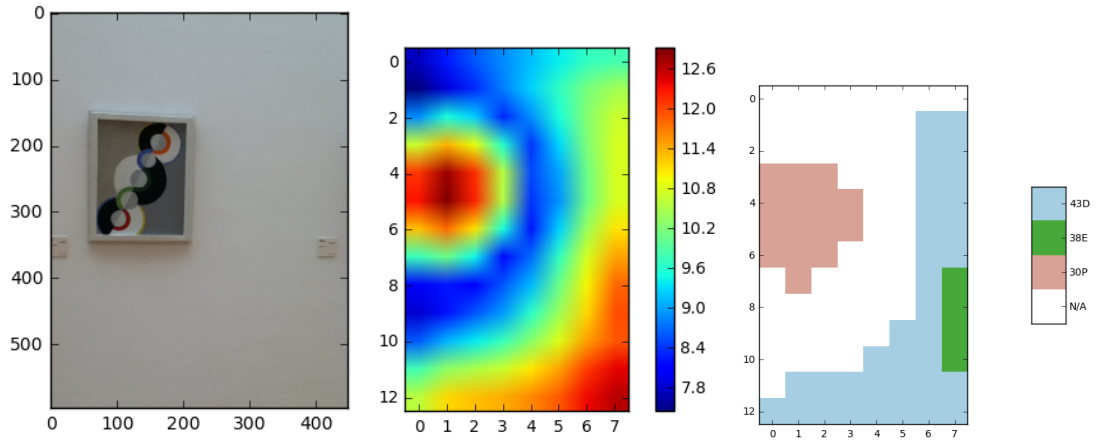
However, in instance search with museum datasets, it is not obvious where the regions of interest should be: most images represent an entire painting or parts of it and only some may contain the painting as part of the image with a wall in the background. This means for most



(a) Image with label 10A (b) Heat-map for 10A (c) Label-map for 10A



(d) Image with label 5P (e) Heat-map for 5P (f) Label-map for 5P



(g) Image with label 30P (h) Heat-map for 30P (i) Label-map for 30P

Figure 3.1 – Sample images (scaled to a smaller side of 448 pixels) along with the heat-map of maximal activation values at each location when a fine-tuned ResNet-152 is applied to the image in a strided manner, as well as the labels of all maximal activations that are greater than the mean maximal activation

images, the ground-truth region of interest is simply the entire image, and some may have a ground-truth region of interest which is almost the entire image, excluding only a small part of the background.

On the other hand, a network fine-tuned on classification on such a dataset should be able to easily identify the region containing the painting, since the background wall is contained in almost all classes, which means it is a particularly bad indicator of the class. Thus, if the network is applied in a strided manner across an image, it should produce low maximal activations in parts containing big sections of background wall. This idea of using a network fine-tuned on classification as a predictor of regions of interest has already been investigated by Oquab et al. [27]. They show that a CNN can predict approximate locations of objects using a multi-scale sliding window training.

Figure 3.1 shows images, along with the heat map representing the maximal activation of a fine-tuned ResNet-152 at each coordinate, when the network is applied in a strided manner across the input image. The fine-tuning was carried out as described in Section 3.1. From this image, we can see that the highest maximal activations of the network usually occur at the location of the object. This is true even if the object is not correctly classified by some of the highest activations as can be seen in images 3.1d - 3.1f.

In the images 3.1g - 3.1i, it seems like many high maximal activations occur specifically in the background area. However, the corresponding label-map shows that these areas correspond to the labels 38E and 43D. Both of these labels are pieces of art which consist mostly of the background wall. In this sense, it is not entirely wrong to consider *wall-only* patches of the image as instances of these pieces of art. This simply means that the image consists of two separate regions of interest: one region with the painting (label 30P) and one region containing only the wall (labels 38E/43D).

From these observations, we can confirm the assumption that the maximal activations of a fine-tuned network are a good indicator of the location of an object, or a combination of different objects. Using this assumption, there is no need for a procedure to annotate regions of interest, as employed by most state-of-the-art image retrieval approaches [9, 46, 32].

On the other hand, using datasets developed for image retrieval, such as Paris6k or Oxford5k [30, 29], this assumption cannot be applied, since the dataset is not clean enough for a network fine-tuned on classification to be a good indicator of location of the query objects.

3.3.2 Incorrectly identified images

Figure 3.2 shows typical images that were correctly identified versus images that were incorrectly identified, by a fine-tuned network as well as the network published by Gordo et al. [9], along with the average precision for the respective queries.

From these results, we see that the networks struggle with images at very different scales, achieving very low average precision, while images with similar scales are usually perfectly matched.

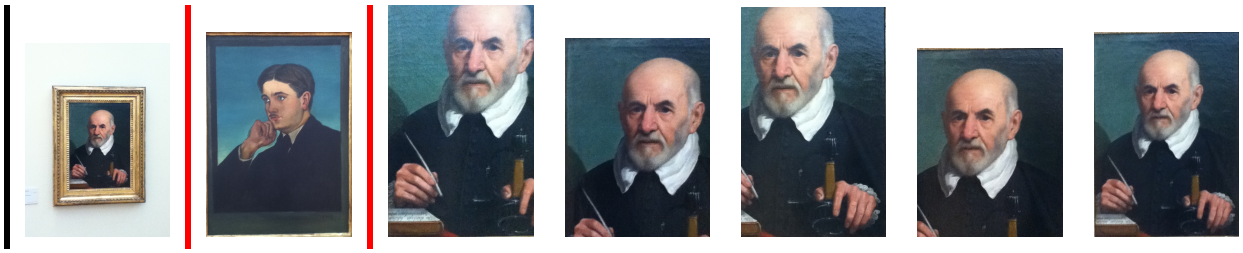
Combining both of the properties identified in this section, we propose a novel approach to learn a strong descriptor for instance search in the next section.

3.4 Proposed instance search architecture

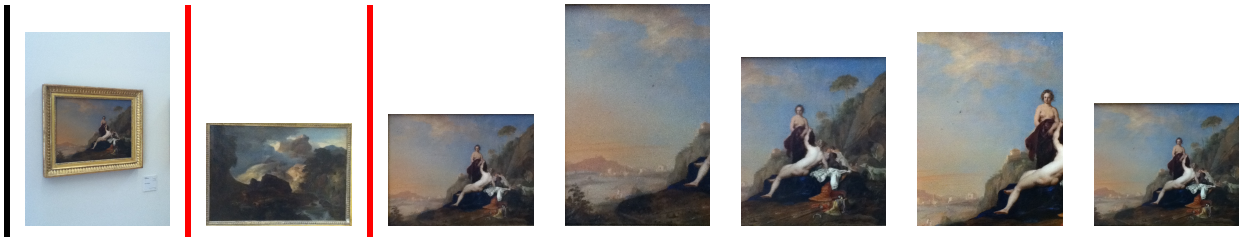
The proposed approach is based on multiple steps as described in the following sections.

(a) Correctly identified query along with the best match and all other possible reference images.
Average precision Gordo et al. [9] (ResNet-50, multi-res): 1.0, fine-tuned ResNet-152: 1.0

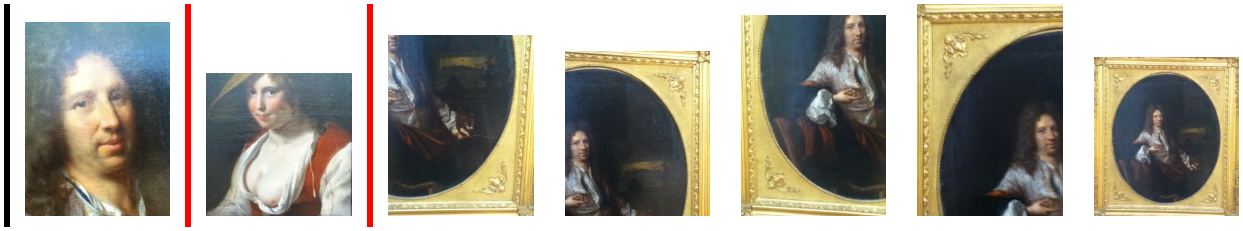
(b) Correctly identified query, along with the best match and all other possible reference images.
Average precision Gordo et al. [9] (ResNet-50, multi-res): 1.0, fine-tuned ResNet-152: 1.0



(c) Incorrectly identified query along with the best match and all possible reference images.
Average precision Gordo et al. [9] (ResNet-50, multi-res): 0.113, fine-tuned ResNet-152: 0.014



(d) Incorrectly identified query along with the best match and all possible reference images.
Average precision Gordo et al. [9] (ResNet-50, multi-res): 0.013, fine-tuned ResNet-152: 0.002



(e) Incorrectly identified query along with the best match and all possible reference images.
Average precision Gordo et al. [9] (ResNet-50, multi-res): 0.004, fine-tuned ResNet-152: 0.004

Figure 3.2 – Sample images that were correctly and incorrectly identified by a fine-tuned ResNet-152, as well as the modified ResNet-50 published by Gordo et al. [9] using images at multiple resolutions. For each image, the query image is shown at the very left, the best match of the ResNet-50 is shown next to it with green bars indicating a correct match, red bars indicating a wrong match, and all possible correct reference images are shown to the right of the query. The caption contains the associated average precision values. Note that this figure is best viewed in color.

3.4.1 Fine-tuning on classification using an FCN

As shown in Section 3.3, a fine-tuned CNN is already a good indicator of the location of an object in our datasets. Additionally, it seems like scale is a particularly important factor.

Thus, the idea is to start by fine-tuning a network with images at different scales. This can be achieved by using a fully convolutional network (FCN), as introduced by Long et al. [23].

In an FCN, the final fully connected layers of a network are replaced by convolutional layers having a kernel which fits the entire domain of the output of the previous layer. This type of convolution is equivalent to a fully connected layer, but allows inputs (and outputs) of any size. The effect is that the network can be applied in one pass to an arbitrarily sized image. The output then represents the activations of the network as if it was applied in a strided manner across the image. The stride of a full network depends on the architecture and is 32 pixels for the architectures used here: AlexNet and ResNet.

Once an FCN is applied to the image, the loss is calculated by averaging the cross-entropy loss across all locations and outputs. The final loss is then obtained by passing images at different scales through the FCN and averaging across all cross-entropy losses of all outputs and scales. Note that we assume that most of the images contain almost exclusively the object of interest. If this is not the case, many patches in the sliding window would be mis-labeled.

Equation 3.5 shows the loss for a single image. S represents the number of scales at which the image is passed through the network. H_s and W_s represent the height and width of the feature map respectively, given the scale s of the input image. $CE^{h,w}$ represents the cross-entropy loss applied at spatial location (h,w) in the full feature map. It is applied for each prediction $y^{h,w}$ of the network and the constant true label $\hat{y}$.

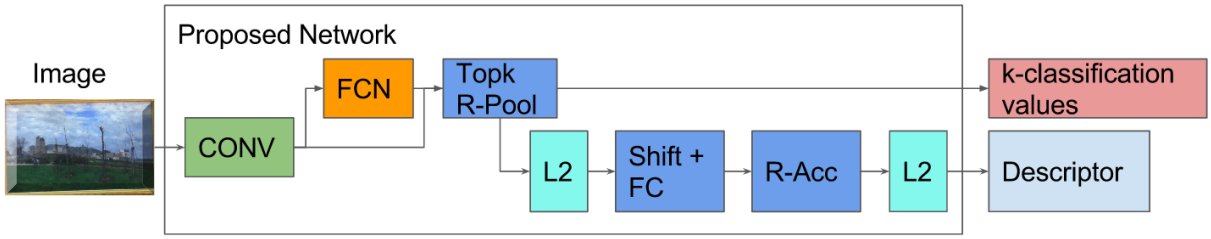
$$\mathscr{L} = \frac{1}{S}\sum_{s=1}^{S}\frac{1}{H_s * W_s}\sum_{h=1}^{H_s}\sum_{w=1}^{W_s} CE^{h,w} \tag{3.5}$$

Additionally, we can see from Equation 3.5 that we choose to give each scale of the image the same weight in the loss. This attenuates the effect of large scales with many predictions (high values for H_s and W_s).

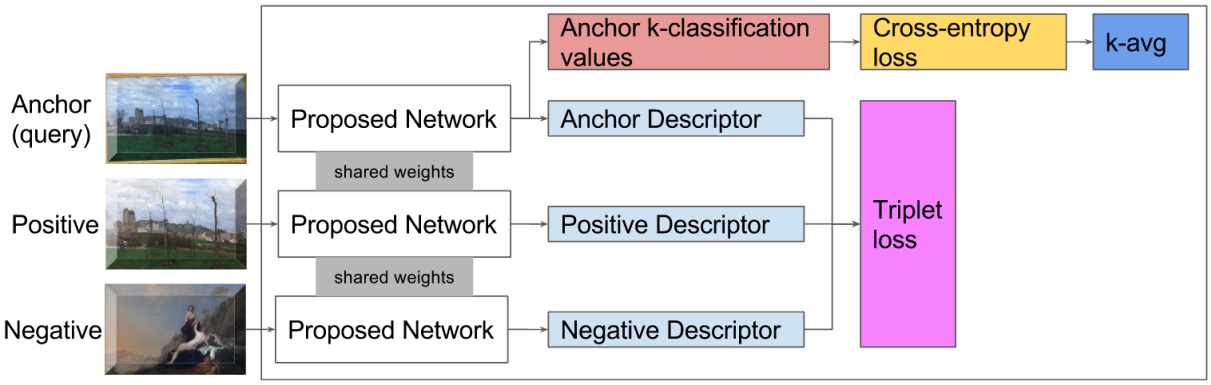
Finally, this loss can only be applied to one image at a time because the images are passed to the network at their true aspect ratio, which means the loss for different images may have different values for the heights and widths of the feature maps H_s and W_s.

In order to normalize the sizes of the features present in the images, all images are scaled to have the same number of pixels in the smaller side. Note that for large aspect ratios and large scales of the smaller side, the memory consumption of training can be high for single images having a very large aspect ratio. To limit this spike in memory consumption, the aspect ratios are limited by introducing uniform random noise on the smaller side of images with high aspect ratios.

In our experiments, we use a maximal aspect ratio of 2.0 and images at two scales of 448 and 224 pixels for the smaller side. We found that the AlexNet architecture did not have good convergence behavior. This is most likely due to the worse generalization property of AlexNet as compared to ResNet and the large number of patches in an image with smaller side 448. Thus, for AlexNet we used scales of 384 and 224 instead.



(a) Proposed architecture for instance search, at deploy time



(b) Proposed architecture for instance search, at training time

Figure 3.3 – Proposed architecture for instance search, based on a FCN [23] for region proposals. The descriptor extraction for each region is similar to the state-of-the-art architecture shown in Figures 2.1 and 2.2

Gradient descent using micro-batches

Since only a single image is passed to the network at a time, the stochastic gradient descent algorithm used to optimize the parameters of the model does not converge, as a single image is not representative of the dataset as a whole and does not form a large enough mini-batch.

Instead, we form a mini-batch by accumulating gradients from many *micro-batches*, each of which is made up of only a single image. This allows to form a mini-batch of any size to obtain stable convergence in stochastic gradient descent.

However, using a micro-batch formed of a single image represents an issue with the ResNet architecture: as described in Section 2.4.3, this model uses batch-norm layers [15]. When training a model, this layer keeps a running mean and variance, which approximates the true mean and variance of the dataset and which is applied during testing. With a micro-batch size of 1, this running mean and variance cannot be inferred correctly, which leads to very poor convergence behavior of the model. Instead, we pre-train the model on the new dataset as described in Section 3.1 in order to obtain a correct estimate of the running mean and variance. Then, we preset the network with the parameters of this fine-tuned network when training the FCN as described above.

3.4.2 Descriptor extraction network

The second step of the proposed approach relies on the FCN, trained as described in the first step in Section 3.4.1.

Figure 3.3 illustrates the proposed architecture. To obtain a descriptor, we first apply the convolutional layers of a previous architecture, such as AlexNet or ResNet. We then obtain all classification outputs at all locations using the pre-trained FCN. We only consider the maximal activation at all locations. The locations with the top k maximal activations will form the descriptor. Here, we can see that we only need an approximation of the location of an object since the descriptor is composed of convolutional activations with a receptive field as large as the receptive field of the whole network: usually a patch of more than 224×224. Hence, we only need to locate objects in this large patch, which can be achieved by an FCN as discussed by Oquab et al. [27], and described in Section 3.4.1.

For each of the top k locations, we closely follow the architecture shown in Figure 2.1. The convolutional features are reduced by a $\|\cdot\|_2$-normalization, then a shifting and fully connected layer. Finally, all descriptors from the k locations are sum-aggregated and $\|\cdot\|_2$-normalized again.

When training, the network is applied to a triplet of images. These triplets are chosen as described in Section 3.2 for the simplified Siamese architecture.

Additionally, we regularize the triplet loss by a cross-entropy loss to make sure that the k locations with highest maximal activations are correctly classified. This loss is averaged over the k locations.

Equation 3.6 shows the full loss as used in our experiments to train the proposed model, for a N images. In this equation, (h_l, w_l) represents the spatial coordinates of the l-th region of highest maximal activation in the feature map produced by the FCN.

$$\mathscr{L} = \sum_{i=1}^{N} \left(\max(0, x_i^a x_i^n - x_i^a x_i^p + m) + \alpha \frac{1}{N} \frac{1}{k} \sum_{l=1}^{k} CE_i^{h_l, w_l} \right) \tag{3.6}$$

In our experiments, we choose the number of regions with highest maximal activation to be $k = 6$ and the regularization hyper-parameter $\alpha = 1.0$. The margin of the triplet loss is $m = 0.1$ as described in Section 3.2.

The results obtained by the proposed approach are listed in Table 5.1 in chapter 5.

3.4.3 Advantages and limitations

As shown in Section 3.3, this approach allows the network to decide which region of interest is best suited for classification and ultimately which regions are best suited for comparison with other images.

Another advantage is that this approach does not require any annotation of the images with regions of interest, which can be a long, manual or automatic process, as evident from the cleaning process used by Gordo et al. [10].

Finally, an important property of the descriptor is that it heavily relies on the classification capabilities of the network. This means the descriptor is mostly meaningless for a different dataset and needs to be learned for each dataset. This can be an advantage, since the descriptor can be better suited to a particular dataset and the learning process does not take long, as shown in Section 5.2. On the other hand, this means that the descriptor cannot be applied in a typical image retrieval task.

Another limitation is the following: for the region descriptor extracted from an FCN to be robust, the dataset used to fine-tune the FCN needs to be very clean, as described in Section 3.3. Since there are very few images per instance and each instance represents a class, a

classification network can easily identify a class if it consists of one or two images representing background noise. This can render the region descriptor meaningless, since in this case, any part of an image which contains mostly background noise may be labeled with high confidence as such a *background noise instance*. For the region descriptor to be meaningful, it is thus important that the training images contain only *meaningful* representations of instances and that they are correctly labeled with respect to those instances.

3.5 Augmenting image descriptors

Up to this point, we consider each approach as if it provides a single, strong descriptor for each image. In information retrieval, many approaches combine descriptors, or use augmentation techniques in order to improve the results provided by a single descriptor.

The following sections quickly describe these approaches and justify whether or not they may improve the results obtained here.

3.5.1 Combination of descriptors

Many approaches in information retrieval combine multiple descriptors in order to achieve better results [30, 19, 6]. In particular, it is common to combine SIFT descriptors [24] and multi-scale Hessian interest points [25]. However, one of the main motivations for our research is precisely to avoid this combination by using an end-to-end approach based on optimizing a single descriptor globally. On the other hand, individual descriptors are usually optimized for a specific task, such as scale-invariance or lighting invariance. Since we want to avoid choosing different descriptors to combine, we do not explore this approach.

3.5.2 Query expansion

Query expansion is a method of improving results by performing multiple queries: first, the descriptor of a query image is used as-is. Then, the descriptor is combined with the descriptors of the top k retrieved results, by simply adding the descriptors together. This new descriptor is used to perform a second query, which gives the final result.

This technique was introduced by Chum et al. [6] and is especially useful when the descriptor of the query can be spatially matched against the returned descriptors. This eliminates some of the returned descriptors from the combined descriptor used to perform a second query. However, as CNNs produce global descriptors of images, this spatial matching cannot be performed.

Furthermore, we do not expect query expansion to provide any major improvements in our research problem, since we expect to have very few images returned. This means the only plausible value of k would be $k = 1$. However, if the best matching descriptor of the first query already matched, the second query cannot improve the result and if it does not match, it is unlikely that the second query would match. Overall, query expansion may improve mean average precision, but it is unlikely to improve mean precision@1.

3.5.3 Database-side feature augmentation

Another common approach to improve the performance of descriptors in image retrieval was introduced by Turcot and Lowe [47] and Arandjelovic and Zisserman [1].

The idea is to combine the descriptors of the reference images in order to form more robust database-side descriptors. Every reference descriptor is replaced by a combination of itself and the k nearest neighbors. This combination is computed as a weighted sum, weighted by the rank of the neighbors with respect to k (the closest neighbor has the highest weight and the k-th neighbor the lowest). The neighbors can be pre-computed once all descriptors have been computed for the reference images, by comparing each image to all others. This can be done efficiently in our case where descriptors can be compared to other descriptors using a matrix multiplication.

However, it is difficult to choose the value for k, as it should be low enough to not include many irrelevant images for the descriptors, but high enough for the technique to have an impact at all.

3.5.4 Instance feature augmentation

Since we have the labels of the reference images available, we propose an approach similar to database-side feature augmentation in order to improve the results: each descriptor is replaced by a combination of itself with all the descriptors of the same instance.

Just as in database-side feature augmentation, for each descriptor, we compute a weighted average of the neighboring descriptors, with the neighbors being all descriptors of reference images of the same instance. For each descriptor, the average is weighted by the rank of the other descriptors in comparison to it. Finally, the resulting descriptor is normalized again.

The results obtained by instance feature augmentation are listed in Table 5.1 in chapter 5, abbreviated by *IFA*.

— 4 —
Evaluation

4.1 Datasets

The proposed approaches as well as several base-lines are evaluated on two datasets: the CLICIDE and GaRoFou datasets. These datasets are described in detail by Portaz et al. [31]. Both datasets are typical of instance search datasets in museums or touristic sites: the objects represented by their images are paintings for one and glass cabinets containing sculptures and artifacts for the other dataset. Both datasets contain a small number of images per instance and a small number of images in total. Table 4.1 lists the content and statistics of the datasets in detail. In particular, this table shows that the total number of images is quite small (order of magnitude of 1000), the number of instances is large both in the dataset as well as the test sets (order of magnitude of 100), and the median number of images per instance is small (2 for GaRoFou, 6 for CLICIDE). This sets both datasets clearly apart from the most common datasets used in image retrieval, such as Oxford [29], Paris [30] as well as Holidays [17].

	CLICIDE	*GaRoFou*
Type of objects	Paintings	Glass cabinets containing various objects
Filtered reference images	3245	1068
Instances in filtered reference images	464	311
Median images per instance	6	2
Min images per instance	1	1
Max images per instance	22	45
Filtered query images	165	184
Instances in filtered query images	134	166
All reference images	3452	1068
Instances in all reference images	473	311
All query images	177	184
Instances in all query images	143	166

Table 4.1 – Details of the datasets used for evaluation

4.1.1 CLICIDE

The CLICIDE dataset contains photographs taken in the Grenoble Museum of Art. The objects represented in the photographs are paintings and still images, exclusively. The full dataset contains 207 images with no meaningful content, mostly showing walls. These images are labeled as such and have been filtered out of the dataset in our evaluation. This explains the difference between total number of images and number of images in Table 4.1. Furthermore, there are 12 query images which contain objects not present in the reference images. These have been filtered out as well.

The CLICIDE dataset is characteristic because the different images for each instance usually consist of at least one global view of the painting and multiple other images representing sub-regions of the same painting.

4.1.2 GaRoFou

The GaRoFou dataset contains high-resolution photographs taken in the Museun of Fourvière in Lyon. The objects represented in the photographs are a cultural heritage collection: sculptures, steles and small artifacts grouped together in glass cabinets.

The full GaRoFou dataset is larger than the part used here: along with the dataset of images, it contains a dataset of videos as well as images extracted from these videos. The video collection has not been used as part of our research.

4.2 Metrics

There are several commonly used metrics in classification and information retrieval. The following sections define these metrics and their use in our research.

4.2.1 Precision

$$P = \frac{TP}{PRED} \tag{4.1}$$

$$P_q = \frac{RR_q}{RET} \tag{4.2}$$

$$P_q@k = \frac{RR_q^k}{k} \tag{4.3}$$

$$MP@1 = \frac{1}{Q}\sum_q P_q@1 \tag{4.4}$$

In classification, precision is a measure of how many correct predictions were made out of all predictions. Equation 4.1 shows this relationship: TP are the true positives, the correct predictions, and $PRED$ is the number of all predictions, including incorrect ones. $PRED$ is usually equal to the size of the testing dataset in classification.

In information retrieval, similarly, precision and more generally precision@k is a measure of how many relevant documents were retrieved out of all retrieved documents. However, the metric is evaluated for each query, instead of the whole testing dataset. Equations 4.2 and 4.3

illustrate this: P_q is the precision for query q, RR_q represents the number of retrieved and relevant documents for query q. RET represents the total number of retrieved documents. Finally, RR_q^k represents the number of retrieved, relevant documents out of the k first retrieved documents for query q.

Thus, precision@k simply uses a cut-off at k: it measures how many relevant documents there are in the first k retrieved documents.

In instance search, only the first retrieved document is important: if the first result is correct, we correctly identify the instance. Hence, we are mostly interested in the precision@1 metric here. In order to generalize this metric to all queries, the mean precision@1 is used, as defined in Equation 4.4. In this equation, Q represents the number of queries.

4.2.2 Mean average precision

$$R_q = \frac{RR_q}{REL_q} \tag{4.5}$$

$$AP_q = \frac{1}{REL_q} \sum_{k=1}^{RET} rel(RET_k) P@k \tag{4.6}$$

$$MAP = \frac{1}{Q} \sum_{q} AP_q \tag{4.7}$$

In information retrieval, average precision is commonly used, since it allows to consider the order in which documents are returned by a system. Average precision represents the area under the precision-recall curve for a given query q. Recall for query q is defined as in Equation 4.5 as the fraction of retrieved, relevant documents for query q as compared to all relevant documents to query q.

Equation 4.6 defines average precision for a given query q: it can be expressed as a finite sum over the precision@k values at every possible cut-off k, multiplied by an indicator $rel(RET_k)$, which is 1 if the k-th retrieved document is relevant. This sum is then normalized by the total number of relevant documents for query q REL_q. Note that in the literature, different definitions of average precision can be found. We chose to follow a common definition of this metric, which is shared by the Oxford [29] and the Paris [30] datasets.

Finally, mean average precision MAP represents the mean of all average precision values over Q queries, as defined in Equation 4.7.

In instance search, mean average precision is not as important as in image retrieval, since we only care about the first ranked result. Nevertheless, it can be an interesting metric to allow better comparison between different systems, so we compute the mean average precision for all results along with the mean precision@1.

4.3 Evaluation methodology

Unless explicitly noted, an evaluation of the system is carried out as follows:

All reference images are encoded with a descriptor extracted from the network to be tested. The query image is encoded the same way. All descriptors are normalized and the descriptor of the query image is compared to all reference images by computing the dot product between it and all descriptors of the reference images. This can be performed by a single vector-matrix

	CLICIDE	*GaRoFou*
SIFT	70.08	78.82
Gordo et al. [9] (ResNet-50)	90.30	**95.65**
Gordo et al. [9] (ResNet-50, multi-res)	**92.73**	**95.65**
AlexNet IN	72.73	85.87
ResNet-152 IN	72.12	85.33

Table 4.2 – Evaluation base-lines set for the CLICIDE and GaRoFou datasets. The results are expressed in percentage points of mean precision@1

multiplication. The resulting values describe the cosine similarity between the descriptor of the query image and the descriptors of reference images, as described in more detail in Section 3.2.1.

Hence, after a query image is compared to all reference images through a vector-matrix multiplication, the index of the highest value in the resulting vector indicates the most similar reference image.

4.4 Baselines

Table 4.2 shows the baselines set for the two evaluation datasets, as described in Section 4.1. The metric used to compare systems is mean precision@1, as described in Section 4.2.1, and the evaluation is carried out as described in Section 4.3.

The descriptors used for comparing images are extracted using different methods. The SIFT method has been previously evaluated by Portaz et al. [31], who have detailed this evaluation.

For the *AlexNet IN* and *ResNet-152 IN* methods, an AlexNet and ResNet-152 architecture is pre-trained on the ImageNet dataset. Then, the descriptor is simply extracted by passing an image through the convolutional layers of the network and normalizing the output. For the ResNet architecture, the final average pooling is performed as well to produce a descriptor of similar dimensions to the others. This means the descriptor has dimension 2048 for the ResNet architecture and 9216 for the AlexNet architecture.

For all networks presented here which require images of a specific size, all images are simply pre-processed by resizing them to the expected size. Since the networks are based on networks built for the ImageNet challenge [37, 11, 21], the expected size is 224×224. The results of pre-trained networks on ImageNet (*AlexNet IN* and *ResNet-152 IN*), fine-tuned networks on classification (Section 3.1) and simplified Siamese networks (Section 3.2) all use images resized to 224×224.

The method by Gordo et al. [9] is based on the pre-trained network published publicly, based on a ResNet-50 architecture. For this network, images are only pre-processed to have a common size of 800 pixels on the smaller side of the image, as the network works with region proposals and thus accepts any scale of image. This network outputs a descriptor with 2048 dimensions. Finally, the multi-resolution method uses input images with different sizes of the smaller side (550, 800 and 1050 pixels).

4.5 Performance

The performance evaluation is carried out on a typical system used for training CNNs. The components approximately represent the highest standard available for a single-GPU server architecture in 2016.

The server contains the following components:

1. CPU: Intel® Xeon® E5-2623 v4, 16 cores, 10MB cache, 2.60GHz
2. Memory: 144 GB, DDR4
3. Graphics card: NVIDIA® Titan X, Pascal architecture, 12 GB memory (two of such graphics cards were available but only one was used in the performance evaluation at a time)

Performance is measured in time in milliseconds or hours required to execute the task, as well as GPU memory required. If multiple sub-tasks are performed for a task, the reported performance is simply the accumulation of time and the maximum of required GPU memory of the sub-tasks.

4.5.1 Training

The training performance is evaluated using the server as described above and the CLICIDE dataset as described in Section 4.1, containing 3245 images.

The training performance was measured only for tasks specific to the dataset, as other tasks can be performed once upfront for any dataset and parameter weights can be transferred in a negligible amount of time. In particular, this means that pre-training a network on a dataset not specific to our task is counted as using no time or memory.

4.5.2 Testing

The testing performance is evaluated using the server as described above and the validation dataset of CLICIDE, as described in Section 4.1. This dataset contains 165 images. However, the performance is reported as the average performance for a single image, as this is the most common use-case when applying the instance search system.

— 5 —
Results

In this section, we present the experimental results obtained in our research.

5.1 Evaluation results

In this section, we present the results obtained on the presented datasets using our network as described in Section 3.4, as well as various other approaches.

Table 5.1 gives an overview of the results obtained. First, the baselines established in Section 4.4 are shown again. Additionally, we show the relevant results obtained by fine-tuning a classification network, abbreviated by FT in the table. This method is described in Section 3.1. We then show the results obtained by a simplified Siamese architecture, as described in Section 3.2, abbreviated SS. Finally, we show the results obtained by the proposed network as described in Section 3.4. In addition to the mean precision@1, we show the mean average precision

	Mean Precision@1		*Mean Average Precision*	
	CLICIDE	*GaRoFou*	*CLICIDE*	*GaRoFou*
SIFT	70.08	78.82	N/A	N/A
Gordo et al. [9] (ResNet-50)	90.30	95.65	65.49	88.43
Gordo et al. [9] (ResNet-50, multi-res)	92.73	95.65	70.36	89.32
AlexNet IN	72.73	85.87	32.71	66.11
ResNet-152 IN	72.12	85.33	40.99	70.15
AlexNet FT	78.18	90.76	38.51	72.92
AlexNet SS	75.76	90.20	36.20	77.73
Proposed AlexNet	81.21	83.15	45.53	71.71
Proposed AlexNet (IFA)	80.61	82.61	71.02	81.66
ResNet-152 FT	79.39	94.57	75.11	93.44
ResNet-152 SS	85.45	95.11	83.00	91.90
Proposed ResNet-152	**94.55**	**96.20**	82.94	91.83
Proposed ResNet-152 (IFA)	93.94	95.11	**94.23**	**93.86**

Table 5.1 – Evaluation results for the CLICIDE and GaRoFou datasets. The results are expressed in percentage points of mean precision@1 and mean average precision (only indicative)

	CLICIDE training	
	Time (h)	GPU memory (MiB)
Gordo et al. [9] (ResNet-50)	0	0
Gordo et al. [9] (ResNet-50, multi-res)	0	0
AlexNet IN	0	0
ResNet-152 IN	0	0
AlexNet FT	1.429	1706
AlexNet SS	9.985	1720
Proposed AlexNet	49.06	1956
ResNet-152 FT	1.621	6910
ResNet-152 SS	12.59	9332
Proposed ResNet-152	86.62	10494

Table 5.2 – Performance of different approaches in terms of time in hours and maximal GPU memory usage in MB when training on the CLICIDE dataset

obtained by the different approaches.

In Table 5.1, all descriptors have dimension 2048, except for the AlexNet IN and AlexNet FT methods. As described in Section 4.4, these descriptors have dimension 9216, since they consist of the convolutional features of an AlexNet architecture.

In Table 5.1, *IFA* refers to the results obtained using instance feature augmentation, as presented in Section 3.5.4.

5.2 Performance

Table 5.2 shows the performance of various approaches when training on the CLICIDE dataset. We assume that pre-trained networks have 0 cost. This table shows that fine-tuning a network on classification on the new dataset is quite fast and has a low memory footprint. This is interesting since fine-tuning already achieves better results than many standard approaches, as can be seen in Section 5.1. We can also see that training our proposed architecture is much more costly, although the absolute cost is not excessive: around 86h of training time in total for the very deep ResNet-152 and 49h for AlexNet, using our larger dataset with over 3000 images.

Table 5.3 shows the performance of the different networks when evaluating on a single image. For the baseline based on SIFT, no performance measures were available. For the Gordo et al. [9] baseline, we simply took the measurements from their paper, performed on a slightly different system, as we could only test a slower version of the model, which does not evaluate the entire system on GPU, and is thus about 10 times slower than what was reported in the paper. For the results of the multi-resolution system, we scale the time by a factor measured by this slower version between the multi-resolution and the normal system.

We can see from Table 5.3 that all networks of the same architecture have approximately the same performance, with the proposed architecture being 2-3 times slower than a network fine-tuned on classification. However, compared to the best performing network proposed by Gordo et al. [9], the proposed AlexNet is two orders of magnitude faster and the proposed ResNet152 is one order of magnitude faster.

	CLICIDE test per image	
	Time (ms)	GPU memory (MiB)
Gordo et al. [9] (ResNet-50)	200	N/A
Gordo et al. [9] (ResNet-50, multi-res)	922	N/A
AlexNet IN	1.14	850
ResNet-152 IN	23.9	972
AlexNet FT	4.6	1012
AlexNet SS	6.27	800
Proposed AlexNet	12.3	1116
ResNet-152 FT	26.1	944
ResNet-152 SS	27.6	1754
Proposed ResNet-152	45.2	2212

Table 5.3 – Performance of different approaches in terms of average time in milliseconds and maximal GPU memory usage in MB when extracting the descriptor of a single image of the CLICIDE dataset

This means that the proposed AlexNet architecture can be evaluated in real-time and the proposed ResNet152 almost in real-time on a suitable server. Note that this real-time performance may not be possible with the current state of the art in mobile computing, but it may be an indication of what will be possible in future mobile platforms.

The difference in performance can be mostly explained by the differences in image sizes used by the various systems. While a fine-tuned network uses images of size 224×224, the proposed network uses images with smaller side 448 for ResNet-152 or 384 for AlexNet and the system by Gordo et al. uses images with smaller side 800 and 550, 800 and 1050 for the multi-resolution version.

5.3 Interpretation

The comparison of performance between different descriptors should be based on the mean precision@1. However, the mean average precision is indicative of how well the model captures the pattern or characteristics which defines an instance or sets it apart from other instances. In particular, it is possible to obtain a high mean precision@1 while achieving a low mean average precision if the descriptor describes visual similarity well enough. This is because there is usually at least one reference image that is visually similar to the query. However, in order to achieve a high mean average precision, the descriptor needs to capture the specific characteristics of the instance that set it apart from other instances. For these reasons, we show the mean average precision obtained by the different descriptors along with the mean precision@1.

From the baselines presented in Section 4.4, we can make two observations. First, even a simple global descriptor obtained from the convolutional features of a CNN pre-trained on ImageNet performs better than matching local SIFT descriptors on our datasets. Second, the ResNet-50 proposed by Gordo et al. [9] out-performs the descriptors from pre-trained networks by far, even though it has never seen the images from our datasets during training, either.

Table 5.1 confirms these observations when taking into account the mean average precision

of the ResNet-50 and the convolutional features of networks pre-trained on ImageNet. The difference is more than 10 points gained in mean average precision even when comparing against the ResNet architecture. This means that a ResNet fully optimized for image matching captures the visual information much better than just the convolutional features of a ResNet pre-trained on image classification. This is expected, since that was one of the goals of the approach proposed by Gordo et al. [9].

Another observation we can make from Table 5.1 is that fine-tuning a network on the reference dataset consistently out-performs a pre-trained network. Furthermore, the training cost of fine-tuning is very low, as can be seen from Table 5.2, and the testing costs are equivalent, since the exact same architecture can be used. This shows that transfer learning is very powerful for small datasets with many classes. Indeed, networks with many parameters such as AlexNet and ResNet could not have been trained on such small datasets with uninitialized weights.

However, when comparing the classification fine-tuning method with the simplified Siamese architecture (fine-tuning on image matching using a triplet loss), it is not as clear which one performs better. From the results, we can see that the classification fine-tuning has a better performance for AlexNet while the triplet loss fine-tuning has a better performance for ResNet-152. This is most likely due to two factors: the hyper-parameters when training the Siamese AlexNet were not perfectly suited, hence the convergence behavior is not as good as with the Siamese ResNet. Furthermore, the AlexNet fine-tuned for classification has a much larger descriptor of dimension 9216 versus the descriptor of dimension 2048 of the simplified Siamese architecture. This may explain that the simplified Siamese architecture performs worse for AlexNet and better for ResNet-152.

Finally, when comparing the proposed architecture with the previous ones, it is clear that the proposed architecture out-performs all of them. It achieves higher precision@1 as well as higher mean average precision, especially when combined with the instance feature augmentation as described in Section 3.5.4. The proposed ResNet-152 also out-performs the network proposed by Gordo et al. [9], based on a ResNet-50. The comparison between these two networks is difficult though. This is because on the on hand, our proposed network is trained on the reference dataset used when comparing images, giving it an unfair advantage. On the other hand, the ResNet-50 is trained on the much larger Landmarks dataset [2], giving it the advantage of data volume. As shown in Sections 3.2.3 and 3.3.1, the training methodology developed by Gordo et al. [10] is not applicable to a small, clean dataset, such as the ones used in our evaluation.

A surprising result is the result given by instance feature augmentation as described in Section 3.5.4. While the mean average precision is greater than the mean average precision of the proposed network by a large margin, the mean precision@1 is slightly lower. This may show that the proposed network matched some outliers correctly, while the descriptors produced by instance feature augmentation are more robust overall. However, we do not have a more detailed explanation of this result as of now.

Note that the difference in mean average precision between the ResNet-50 by Gordo et al. [9] and the proposed ResNet-152 are statistically significant, confirmed by a paired student's t-test with p-value $p < 0.001$. This is valid for both versions of the proposed ResNet-152, including IFA or not, compared to both versions of the ResNet-50, multi-resolution or not.

— 6 —
Conclusion

6.1 Conclusions from our research

During our research, we analyzed state-of-the-art approaches in image classification and image retrieval and their application to the instance search problem. We found that these approaches struggle with query images and reference images being at different scales. Additionally, the region proposals used by the state of the art do not seem applicable to the problem of instance search.

We have presented a novel approach, which consists of two key steps. First, we leverage the concept of fully convolutional networks in order to perform classification training at different scales, without a heavy computational overhead. Second, we show that the fully convolutional network can be used to obtain region proposals without the need for an additional component in the network and training. This is particularly important, since region proposals are difficult to define manually in our research problem.

Finally, the proposed network keeps all the benefits of state-of-the-art approaches: it can be trained end-to-end and it produces an effective global descriptor, which can be compared using the dot product. Additionally, it is modular in the sense that it can be built upon any type of CNN, pre-trained for classification. Furthermore, the proposed network is fast to evaluate and the training time for each dataset is reasonable.

Through multiple experiments on two datasets, we show that the descriptor obtained using our proposed network outperforms previous state-of-the-art approaches on the instance search task, while being just as memory-efficient and fast for encoding images.

6.2 Future work

6.2.1 Few-shots classification

While we explored different ideas in this report, some have not been considered yet. For one, a paper by Kaiser et al. [20] was presented earlier this year, which focuses on few-shots classification and achieves state-of-the-art results.

As noted before, our research problem shares many similarities with the few-shots classification problem. Furthermore, this paper presents a method of combining the triplet loss and cross-entropy loss in order to organize a memory of so-called rare events. This is similar to our

proposed approach and may be adapted to instance retrieval. At the very least, it could provide significant insight to the instance retrieval problem.

6.2.2 Conditional similarities

Another interesting paper presented earlier this year is by Veit et al. [49]. It presents a method of capturing semantically different notions of similarity. The main idea is to consider different types of similarity (color, category, etc.) and thus to embed images into many different subspaces, which capture these different types of similarity. Finally, they present a method to jointly learn a global embedding, made up of different sub-embeddings for the sub-spaces representing different similarities.

This idea could be interesting in the instance search problem, as it may allow us to learn different types of categories first and then produce an embedding based on these different categories. For instance, a painting could be described by its main tone (dark, bright, etc.), its art movement (impressionism, dadaism, etc.) or simply its size (big, small, squared, wide, high). Each of these notions may be easily described individually and a joint learning of these different types of descriptors could provide significant improvements over learning a single, global descriptor for each image.

— A —
Appendix

A.1 Reproducibility

A.1.1 Datasets

The datasets used in our research are publicly available at the following URL: `http://lig-mrim.imag.fr/?page_id=455`

A.1.2 Implementation

The implementation used to produce all results presented in this work are accessible publicly at the following URL: `https://github.com/MatthiasKohl/nnForRetrieval`

Links to pre-trained weights of the models used in the evaluation of our research will be added to the repository.

For training and testing on a new dataset, the following steps are needed:

1. Required packages: Python 2.7, NumPy, SciPy, OpenCV-Python (Python bindings for OpenCV 3.2) and PyTorch 0.12

2. Edit and run *pre_process_dataset.py* to pre-process images to be at the correct size (or correct size on the smaller side and given maximal aspect ratio)

3. Edit and run *create_mean_std_file.py* to create the file needed to normalize images by mean and standard deviation

4. Edit *utils/params.py* and add all global parameters to the following dictionaries. All dictionaries have the dataset ID as key. This is the first part of the file-name of the dataset (excluding folder path) before an underscore

    a) *mean_std_files*: key is dataset ID, value is the file-name of the file generated by *create_mean_std_file.py*

    b) *match_label_functions*: a function to extract labels from the name of an image

    c) *num_classes*: the number of classes/instances in the dataset

    d) *image_sizes*: this is usually always (3, 224, 224)

5. For testing, run *python test/[training method]_test.py* and follow the instructions for command-line parameters, which should be self explanatory

6. For training, edit *train/[training method]_p.py*. Most parameters should be self explanatory. Then, run *python train/[training method].py* from the main directory

Bibliography

[1] Relja Arandjelović and Andrew Zisserman. Three things everyone should know to improve object retrieval. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2911–2918. IEEE, 2012.

[2] Artem Babenko, Anton Slesarev, Alexandr Chigorin, and Victor Lempitsky. Neural codes for image retrieval. In *European conference on computer vision*, pages 584–599. Springer, 2014.

[3] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[4] Léon Bottou. Stochastic gradient descent tricks. In *Neural networks: Tricks of the trade*, pages 421–436. Springer, 2012.

[5] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546 vol. 1, June 2005.

[6] Ondrej Chum, James Philbin, Josef Sivic, Michael Isard, and Andrew Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

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

[8] Yoav Freund and Robert E. Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *European conference on computational learning theory*, pages 23–37. Springer, 1995.

[9] Albert Gordo, Jon Almazán, Jerome Revaud, and Diane Larlus. Deep Image Retrieval: Learning Global Representations for Image Search. In *Computer Vision – ECCV 2016*, pages 241–257. Springer, Cham, October 2016.

[10] Albert Gordo, Jon Almazán, Jerome Revaud, and Diane Larlus. End-to-End Learning of Deep Visual Representations for Image Retrieval. *International Journal of Computer Vision*, pages 1–18, June 2017.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[12] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

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

[14] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[15] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 448–456, Lille, France, July 2015. PMLR.

[16] Tommi Jaakkola and David Haussler. Exploiting generative models in discriminative classifiers. In *Advances in neural information processing systems*, pages 487–493, 1999.

[17] Herve Jegou, Matthijs Douze, and Cordelia Schmid. Hamming Embedding and Weak Geometric Consistency for Large Scale Image Search. In *Proceedings of the 10th European Conference on Computer Vision: Part I*, ECCV '08, pages 304–317, Berlin, Heidelberg, 2008. Springer-Verlag.

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

[19] Herve Jegou, Cordelia Schmid, Hedi Harzallah, and Jakob Verbeek. Accurate image search using the contextual dissimilarity measure. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(1):2–11, 2010.

[20] Łukasz Kaiser, Ofir Nachum, Aurko Roy, and Samy Bengio. Learning to remember rare events. *arXiv preprint arXiv:1703.03129*, 2017.

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[22] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[23] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

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

[25] Krystian Mikolajczyk and Cordelia Schmid. Scale & affine invariant interest point detectors. *International journal of computer vision*, 60(1):63–86, 2004.

[26] Andrej Mikulik, Michal Perdoch, Ondřej Chum, and Jiří Matas. Learning Vocabularies over a Fine Quantization. *International Journal of Computer Vision*, 103(1):163–175, May 2013.

[27] Maxime Oquab, Léon Bottou, Ivan Laptev, and Josef Sivic. Is object localization for free?-weakly-supervised learning with convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 685–694, 2015.

[28] Florent Perronnin, Yan Liu, Jorge Sánchez, and Hervé Poirier. Large-scale image retrieval with compressed fisher vectors. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3384–3391. IEEE, 2010.

[29] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[30] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[31] Maxime Portaz, Johann Poignant, Mateusz Budnik, Philippe Mulhem, Jean-Pierre Chevallet, and Lorraine Goeuriot. Construction et évaluation d'un corpus pour la recherche d'instances d'images muséales. In *COnférence en Recherche d'Informations et Applications - CORIA 2017, 14th French Information Retrieval Conference. Marseille, France, March 29-31, 2017. Proceedings, Marseille, France, March 29-31, 2017.*, pages 17–34, 2017.

[32] Filip Radenović, Giorgos Tolias, and Ondřej Chum. CNN Image Retrieval Learns from BoW: Unsupervised Fine-Tuning with Hard Examples. In *Computer Vision – ECCV 2016*, pages 3–20. Springer, Cham, October 2016.

[33] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[34] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: An efficient alternative to SIFT or SURF. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2564–2571. IEEE, 2011.

[35] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning internal representations by error propagation. Technical report, DTIC Document, 1985.

[36] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.

[37] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252, December 2015.

[38] Amaia Salvador, Xavier Giro-i Nieto, Ferran Marques, and Shin'ichi Satoh. Faster R-CNN Features for Instance Search. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2016.

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

[40] Bernhard Scholkopf, Kah-Kay Sung, Christopher JC Burges, Federico Girosi, Partha Niyogi, Tomaso Poggio, and Vladimir Vapnik. Comparing support vector machines with Gaussian kernels to radial basis function classifiers. *IEEE transactions on Signal Processing*, 45(11):2758–2765, 1997.

[41] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 815–823, 2015.

[42] John Shawe-Taylor and Nello Cristianini. *Kernel methods for pattern analysis*. Cambridge university press, 2004.

[43] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv:1409.1556 [cs]*, September 2014. arXiv: 1409.1556.

[44] Josef Sivic, Andrew Zisserman, and others. Video google: A text retrieval approach to object matching in videos. In *iccv*, volume 2, pages 1470–1477, 2003.

[45] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. *arXiv:1602.07261 [cs]*, February 2016. arXiv: 1602.07261.

[46] Giorgos Tolias, Ronan Sicre, and Hervé Jégou. Particular object retrieval with integral max-pooling of CNN activations. *arXiv:1511.05879 [cs]*, November 2015. arXiv: 1511.05879.

[47] Panu Turcot and David G. Lowe. Better matching with fewer features: The selection of useful features in large database recognition problems. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pages 2109–2116. IEEE, 2009.

[48] Andrea Vedaldi and Andrew Zisserman. Efficient additive kernels via explicit feature maps. *IEEE transactions on pattern analysis and machine intelligence*, 34(3):480–492, 2012.

[49] Andreas Veit, Serge Belongie, and Theofanis Karaletsos. Conditional Similarity Networks. In *Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, July 2017.

[50] Kilian Q. Weinberger, John Blitzer, and Lawrence Saul. Distance metric learning for large margin nearest neighbor classification. *Advances in neural information processing systems*, 18:1473, 2006.

[51] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In Z. Ghahramani, M. Welling, C. Cortes, N. d Lawrence, and K. q Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3320–3328. Curran Associates, Inc., 2014.