Robust image representation for classification, retrieval and object discovery

Oriane Siméoni

THÈSE DE DOCTORAT DE

L'UNIVERSITÉ DE RENNES 1

COMUE UNIVERSITE BRETAGNE LOIRE

ÉCOLE DOCTORALE Nº 601
Mathèmatique et Sciences et Technologies de l'Information et de la Communication
Spécialité : Informatique

Par
Oriane SIMÉONI

ROBUST IMAGE REPRESENTATION FOR CLASSIFICATION, RETRIEVAL AND OBJECT DISCOVERY

Thèse présentée et soutenue à Rennes le 10 Septembre 2020
Unité de recherche : Inria, Centre Inria Rennes-Bretagne Atlantique

Rapporteurs avant soutenance :

Pr. Patrick PÉREZ	Directeur scientifique, Valeo
Pr. Josef SIVIC	Directeur de recherche, ENS

Composition du jury :

Présidente :	Pr. Elisa FROMONT	Professeur, Université Rennes 1
Rapporteurs:	Pr. Patrick PÉREZ	Directeur scientifique, Valeo
	Pr. Josef SIVIC	Directeur de recherche, ENS
Examinateurs :	Pr. Cordelia SCHMID	Directeur de recherche, INRIA
	Dr. Hervé JÉGOU	Directeur scientifique, Facebook
	Dr. Diane LARLUS	Senior research scientist, Naver Labs
Dir. de thèse :	Pr. Guillaume GRAVIER	Directeur de recherche, CNRS
Co-dir. de thèse :	Dr. Yannis AVRITHIS	Advanced Research Position, INRIA

Invité :

Dr. Ondřej CHUM Maître de conférence, Czech Technical University

Acknowledgements

I would first like to thank Yannis Avrithis for being such an amazing mentor. He has continuously shared with me his passion for research, taught me what he could and gave me endless support during those three years.

Next, I would like to thank Guillaume Gravier for being always available to discuss ideas as well as emotions. He has been able to hear and understand me when I needed it.

I would like to deeply thank all my jury members who made me the honor of evaluating my PhD. In particular, I would like to thank Elisa Fromont for accepting to be the president of my jury, as well as Patrick Pérez and Josef Sivic who took the time to review my thesis. I also want to thank Hervé Jégou–who made the effort of traveling during a pandemic–, Cordelia Schmid and Diane Larlus for their interesting questions and comments.

I also want to say a special thank to Ondřej Chum for supervising my work throughout a significant time of my PhD, welcoming me in Prague and being part of my jury as invited member. I have been excited by the projects that we have done together.

I would also like to express my gratitude to Patrick Pérez for giving me important feedback every year, as part of my CSID committee. And to Cordelia Schmid for giving me the opportunity to do an interesting internship with her.

I also want to thank Ahmet Iscen, Giorgos Tolias and Mateusz Budnik, who took an important role during my PhD, as collaborators and friends.

I also want to thank all my colleagues, with whom it has been a pleasure to meet every day. I have been very happy to go through this PhD alongside Hanwei Zhang, who has shared with me her solar energy.

I would like to thank all the members of my team for building a kind atmosphere in which I have been able to do my exciting PhD. In particular, I would like to thank Pascale Sebillot for giving me the opportunity to teach–which I have greatly enjoyed–and Ewa Kijak for our very nice discussions.

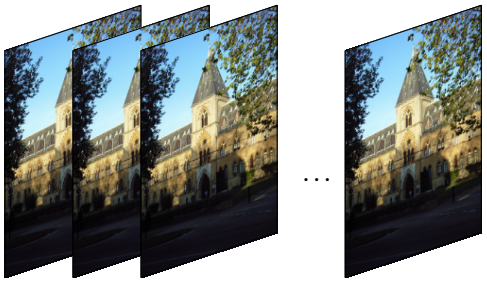
Last but not least, I want to thank my friends and family–you will recognize yourselves–who have been so present during this PhD. They have helped me during the hard times and shared my happy times. I can't thank you enough for your support, and I strongly believe that this PhD would have been much more difficult without you.

Contents

List of Symbols

General

$\mathbf{v}$	a vector
$\mathbf{v}^\top$	transposed vector
$\mathbf{T}$	a tensor
$\mathbf{T}^{-1}$	inverse of the tensor
$\mathbf{1} \in \mathbb{R}^n$	all-ones vector of dimension n
$[i]$	set $\{1, \ldots, i\}$ for $i \in \mathbb{N}$

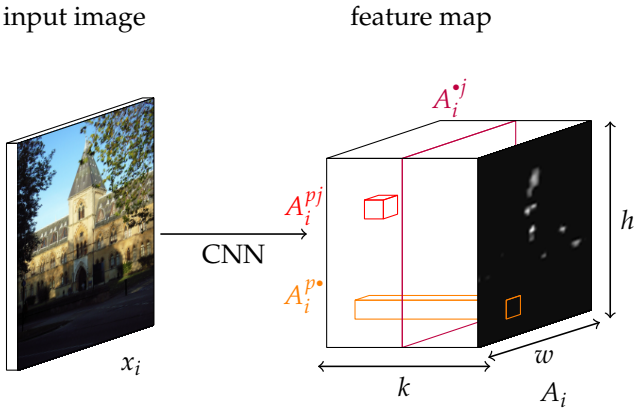


Dataset

x	an image
n	number of images
$\mathcal{I} := [n]$	set of image indices
$X := \{x_i\}_{i \in \mathcal{I}}$	dataset of n images
$\mathcal{X}$	ensemble of all available data
$L \subset \mathcal{X}$	set of labeled examples
$U \subset \mathcal{X}$	set of unlabeled examples

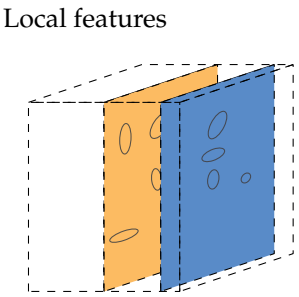
Classification

$\mathbf{y}$	set of labels
y_i	label corresponding to the image x_i with $i \in \mathcal{I}$
c	number of classes
$C := [c]$	set of c classes



Features

$A \in \mathbb{R}^{h \times w \times k}$	feature maps of a CNN
h, w	spatial resolution (height, width) of A
k	number of feature channels of A
A_i	activation map corresponding to the image $\mathbf{x}_i$
A_i^{pj}	element of A_i at position $p \in P$ and channel $j \in [k]$
$P := [h] \times [w]$	set of spatial positions
$A^{p\bullet} \in \mathbb{R}^k$	vector containing all feature channels at position $p \in P$
$A^{\bullet j} \in \mathbb{R}^{h \times w}$	2d feature map of A corresponding to channel $j \in [k]$
S	2d saliency map



Local features

q	a local feature
$\mathcal{Q}$	set of local features
$\mathcal{M}$	a list of set of inliers
T	a transformation

k-NN graph

Considering the graph	of a dataset X containing n images
$W \in \mathbb{R}^{n \times n}$	sparse symmetric nonnegative adjacency matrix
$D := \operatorname{diag}(W\mathbf{1})$	$n \times n$ degree matrix
$\mathcal{W}$	symmetrically normalized adjacency matrix
L_α	graph Laplacian
$\mathcal{L}_\alpha$	normalized graph Laplacian

Acronyms

Chapter 1

Introduction

Computer vision is a research field that focuses on solutions allowing a computer to understand images. Although over fifty years old, the field has gained recent popularity by achieving impressive results that led to the creation of new technologies *e.g.* self-driving cars. This chapter shapes the boundaries of the field in which this thesis belongs.

A historical context is presented in Section 1.1. Section 1.2 discusses difficulties that a computer vision method must deal with, including variations inherent to images captured by different people. We focus on two vision tasks presented in Section 1.3. We discuss our contributions and make our claims in Section 1.4. Finally, we present the structure of this document in Section 1.5 and in particular detail the dependencies between chapters.

1.1 Context

In May 2017, the Economist printed an article entitled "The world's most valuable resource is no longer oil, but data" [17]. It described the shift performed by industries since the early 2000's. Companies are currently making very large profits by exploiting user data often handed over for free. And us, the users, are also taking advantage of all data piled up in servers. We are surrounded by technology containing cameras - phones, cars, drones *etc*. Those machines take pictures and videos of our world, and can help with improving our everyday life if they understand what they capture. What do images depict, what are the objects of interest, what is happening in the scene? All of those questions must be answered exploiting just raw pictures and no other information.

The *computer vision* field studies a large variety of problems. It investigates tasks focusing on 2D, 3D images, or videos, performing image recognition, learning, compression, generation or exploitation. Images can also be associated with other modalities, such as text, sound or meta-information. Computer vision has been studied for decades and today achieves results that reach a broad public. As a result, the field has gained interest and attracts both researchers and industry parties. The 2019 Computer Vision and Pattern Recognition (CVPR) conference gathered about 9000 attendees, 4000 more than in 2017.

We focus this thesis on image understanding. As with most of machine learning, early works were inspired by biology and in particular neurosciences. Rosenblatt [Ros58] was wondering how the biological system would sense, encode, and store information from the

physical world. He proposed a "hypothetical nervous system", called the *perceptron*, which was presented as the beginning of Artificial Intelligence (AI). Relayed by the New York Times 1958 article "New navy device learns by doing" [58], the machine received much attention.

> *"The Navy revealed the embryo of an electronic computer today that it expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence. "*
>
> — New York Times, 1958

However, the promises of a "Perceptron thinking machine that will be able to read and write [...] expected to be finished in about a year at a cost of $100,000" were not met. A few years later the work would be critized by fellow researchers [MP88]. As a result, neural networks suffered a drop in interest leading to an AI "winter".

It would take more than fifty years of research efforts for recognition neural models to achieve compelling results. Those included the development of back-propagation [RHW86], convolutional networks [KSH12; SZ14; He+16], the progress of hardware (*e.g.* Graphics Processing Units (GPUs)) and the collection of very large datasets [Kri09; Don+09]. One achievement has been made by Krizhevsky *et al.* [KSH12] in 2012, who showed that Convolutional Neural Networks (CNNs) outperform previously used hand-crafted features combined with linear classifiers. Enthusiastic, the vision community has focused a large part of its research on deep learning. This led to the definition of highly efficient deep networks [He+16; Hua+17], containing millions of parameters to be learned.

Real-world problems can be translated into learning tasks defined by training and test sets, ground-truth labels and loss functions. There are many computer vision problems, and in particular, image understanding includes tasks that focus on class-level or instance-level information. *Image classification* (Figure 1.1 (a)) consists of discovering what class of objects is contained in an image; *object detection* (Figure 1.1 (b)) requires classified objects to be localized on the image by a bounding box; *instance segmentation* (Figure 1.1 (c)) consists of classifying every pixel of an image; *instance-level image retrieval* (Figure 1.1 (d)) expects to find all the images within a database containing one particular object depicted in a *query* image.

CNNs achieve state-of-the-art results in many computer vision tasks, including all tasks listed above, as well as in text, Q&A and audio tasks. Inputs and expected outputs vary depending on the task. However, CNN-based solutions often share a backbone structure. Indeed, deep networks include first a stack of convolutional layers (the *backbone* network) followed by specialized layers. Networks are often trained in a supervised manner, and labels and losses can be designed together in order to make a network learn the right information. We will further show that deep networks are very flexible and can be used to solve different tasks.

1.2 Learning for vision

The desired qualities of an image representation depend directly on the computer vision task; important properties of an image are different for each task. The image representation needs to be exploitable by computers to make decisions. The design of this representation, either hand-crafted or learned, is a core problem of the computer vision field.

(a) Classification (b) Object detection (c) Instance segmentation (d) Image retrieval

FIGURE 1.1 – Computer vision tasks. Three class-level tasks are presented: the image classification (a), the object detection (b) and the instance segmentation (c). The instance-level image retrieval task is represented in (d): similar images to a query image are selected from a database D and sorted by relevance.

A representation must be robust to variability in an image. On one hand, there is a wide variety of images captured under very different conditions, with different camera quality. Depicted objects may vary due to the change of camera orientation, zoom effects, light changes, day/night shift, and objects may be partially represented due to occlusion or visibility. On the other hand, class-level objects also have a high intra-class variance showed in Figure 1.2. Objects of the same class may look dissimilar, in colors, shapes, textures, or even have an entirely different appearance. Only certain variations and deformations must be addressed by a model depending on the problem.

Early recognition methods relied on hand-crafted features [Low99; MS04], which are computed using mathematical operations directly on the image, independently of a task. Such features have proved to be very relevant when performing instance-level operations, allowing the acquisition of local and fine-grained information. They can be aggregated into a global representation using visual vocabularies *e.g.* Bag of Words (BoW) [SZ03]. Local features can be used by classification methods, trainable or not, such as k-Nearest Neighbors (k-NN) [CH67] or Support Vector Machines (SVMs) [BGV92]. However, hand-crafted features are limited due to their fixed definitions.

CNNs offered the possibility to specialize descriptions to a certain problem by learning image representations directly for the desired task. Indeed, different CNN architectures and losses, such as regression or classification losses, allow extracting different properties from an image. For each task the types of supervision must be defined, as well as labels, learning functions and optimizations that will maximize the performance of a model on a task.

Representation. An image can be described by a set of *local features* or a *global representation*, and, both must describe information invariant to previously listed variations. Local features, *e.g.* Scale Invariant Feature Transform (SIFT) [Low99], characterize details of an object and can help to deal with missing parts, when performing *partial* classification or matching due to occlusion, clutter or the presence of multiple objects. However, storing sets of features per image comes at a high cost, so it can be intractable when dealing with billions of images. Global representations are coarser and contain less information than a set of local features, but they are also more difficult to construct. Good global description often requires learning. Nevertheless, they have the advantage of being cheap to store.

FIGURE 1.2 – Intra-class variability. Images presented in [Par+12] and taken from the MSR ASIRRA dataset [Els+07].

Current state-of-the-art CNNs are translation invariant, and can deal with some rotation and light changes depending on the diversity of the training data. They can be explicitly designed to deal with more image transformations, such as rotation [Jad+15; Est+18], reflection [CW16] or scaling [SM13; KSJ14] using data augmentation. Deep global representations are compact and discriminative; they allow the storage of large datasets in a limited memory space as well as the computation of operations on the whole dataset.

Labels and loss functions. Most machine learning applied to computer vision is trained in a supervised manner. Neural networks require large amounts of labeled data. The definition of a *label* depends on the task: for example a scalar class label per image for classification, a set of bounding box/class label couples for object detection, a label per pixel for instance segmentation, and sets of positive/negative pairs of images for retrieval or manifold learning. The computer vision community has often used the Amazon Mechanical Turk (AMT) service to label images and produce new datasets [Das+17; Agr+19]. This service is expensive in terms of human work, and it is hard to ensure that all workers perform labeling the same way; Oquab *et al*. [Oqu+15] posed the following questions:

> *"For example, should we annotate the dog's head or the entire dog? What if a part of the dog's body is occluded by another object? "*

Improving the construction of datasets is an important problem as the ability of CNNs to generalize depends largely on the variety and quantity of images seen at training.

Supervision. The large classification dataset Imagenet [Don+09] has allowed the broad training of networks, and as an outcome classification results have improved. In order to avoid requiring millions of images carefully labeled by humans for every vision task, the community has worked on minimizing needs in supervision. In particular, unlabeled image data are massively available on the internet and diverse kinds of information can be extracted from them. Efforts have been made to integrate those images to the training of models.

The classification task requires the least and easiest labeling. Minimizing supervision therefore means dealing with unlabeled data, either by learning with only unlabeled data (*unsupervised* learning [JMF99; HS06; Goo+14]), few labeled data (*few-shot* learning [Lak+11; Koc15]) or a mix between the two (*semi-supervised* learning [Zhu+06; Lee13; Shi+18]). *Weakly-supervised* learning [Li+17] can also be performed by adding noisy labels. Alternatively, it is possible to select the most relevant images for model training to be labeled by humans,

which is the core idea of *active learning* [Yan+15b; SS18]. Different levels of supervision can also be combined in order to take advantage of all efforts and data available [DP18].

Tasks requiring more detailed labels, such as object detection or instance segmentation, can also benefit from efforts performed to construct less detailed labels. Weakly-supervised methods allow the object detection task to profit from classification level training [Oqu+15] and instance segmentation can profit from both classification or object detection supervision [He+17]. Other forms of supervision weaker than bounding boxes can also be defined e.g. points [Bea+15] or size estimates [SF16b].

Alternatively to using more or less labeled data, learning can be directly transferred from one model to another using *transfer* learning. In particular, models trained on Imagenet [Don+09] have proved to be a good initialization for new classification task models, or even other tasks. The Imagenet dataset contains 1000 classes, which is more than most of other datasets. *Pre-training* has become standard practice. Experiments proved that initializing models with pre-trained weights improves training quality, both in terms of speed and results, in all tasks including classification [Lec+98], object detection [Lin+14], segmentation [He+17] or image retrieval [Rad+18].

1.3 Understanding and retrieving

Computer vision covers many problems and challenges. In particular, we focus this thesis on two applications, the image classification and image retrieval tasks. Both require different concepts to be extracted from an image. A model designed for image classification must cope with intra-class variability while the image retrieval task asks to deal with instance-level changes. They also differ by the types of supervision and model architectures that can be used to solve them.

1.3.1 Image classification

Among the recognition problems, image classification is one of the easiest in terms of supervision. Considering a limited set of classes, the task consists in predicting a scalar class label per image. Objects of a same class may differ radically from one another. As any class-level recognition problem, image classification models must cope with high intra-class variability. Hand-crafted methods fail to extract high-level semantic information that would allow a classifier to make correct predictions. However, CNNs can be designed to extract conceptual information with specific losses (*e.g.* cross-entropy) and a training set of images annotated by a class label. The ability of CNNs to generalize to very different objects of the same class depends on the amount of images seen at training.

1.3.2 Image retrieval

In the second part of the thesis, we focus on image retrieval. The task requires images depicting the same object(s) as a given query image to be selected from a database and sorted by relevance. It is possible to perform a fast search using *global* image representations. The accuracy of the sorting depends directly on the quality of global representations - either aggregated hand-crafted descriptors [PD07; Jég+10] or CNN-based descriptors [Gor+17; RTC18]. However, a global descriptor often lacks pertinent knowledge. It contains information about the entire image and misses details or fails to focus on the objects of interest.

Regional matching [TSJ16] improves search performances by comparing exhaustively different parts of the images, allowing partial matching. Regional search is more expensive both in terms of computation and storage cost, as several descriptors are stored per image. Such a high expense is not tractable to large scale datasets.

Given sorted images selected by a cheap search with global representations, it is possible to reorder the first images using more expensive techniques. *Spatial matching* [FB81; CMK03] is performed pairwise and consists in fitting a transformation model to a set of local descriptors. This robust matching discards obvious non-matching images but requires the storage of local descriptors. Alternatively or additionally, *query expansion* [Chu+07] allows the discovery of similar images to an expanded version of the query. Although not requiring more image information than a global descriptor, Query Expansion (QE) adds the cost of additional global search.

1.4 Contributions

In this thesis, we discuss several techniques to *extract the most knowledge with the least supervision*. In particular, we focus on the opportunities given by a CNN. As discussed before, a deep network backbone can be used to perform various tasks. We show that a convolutional backbone output - a set of *feature maps* - can be exploited in different ways as presented in Figure 1.3.

The figure presents an outline of the contribution presented in this thesis. Once an image is represented by a feature tensor, it is possible to predict class labels using a trained fully-connected layer as presented in Figure 1.3 (a). A global descriptor can also be pooled (Figure 1.3 (b)) from the feature maps and used in the context of image retrieval. The quality of an image search depends largely on image representations. In particular, one image often depicts more objects than just the object of interest, including clutter and background, which may alter the globally pooled description. We propose a method to identify objects of interest and focus the representation on them (Figure 1.3 (c)). Finally, we show that compact local features can be extracted directly from feature maps, allowing to perform accurate spatial matching (Figure 1.3 (d)).

1.4.1 Active learning

We first focus on the classification task and try minimizing needs in supervision. In the active learning [Set09] scenario, a certain budget of images can be labeled and they must be selected by a system in order to maximize the quality of a training performed with them. This task reduces the need for labeled images. Active learning proved to be relevant when performed in the context of hand-crafted features. Images were selected one at the time, labeled, and added to the training set. When deep learning achieved state-of-the-art on the classification tasks, works adapted the selection process to CNN requirements. In particular, labeling one image at a time would not be relevant as deep models are trained on batches of images; it is now standard practice to label a certain *budget* of images [GE17] at every cycle, usually between hundreds and thousands.

Several acquisition functions were proposed to select images, either based on geometry or a model uncertainty. However, recent works reported that the difference of performance

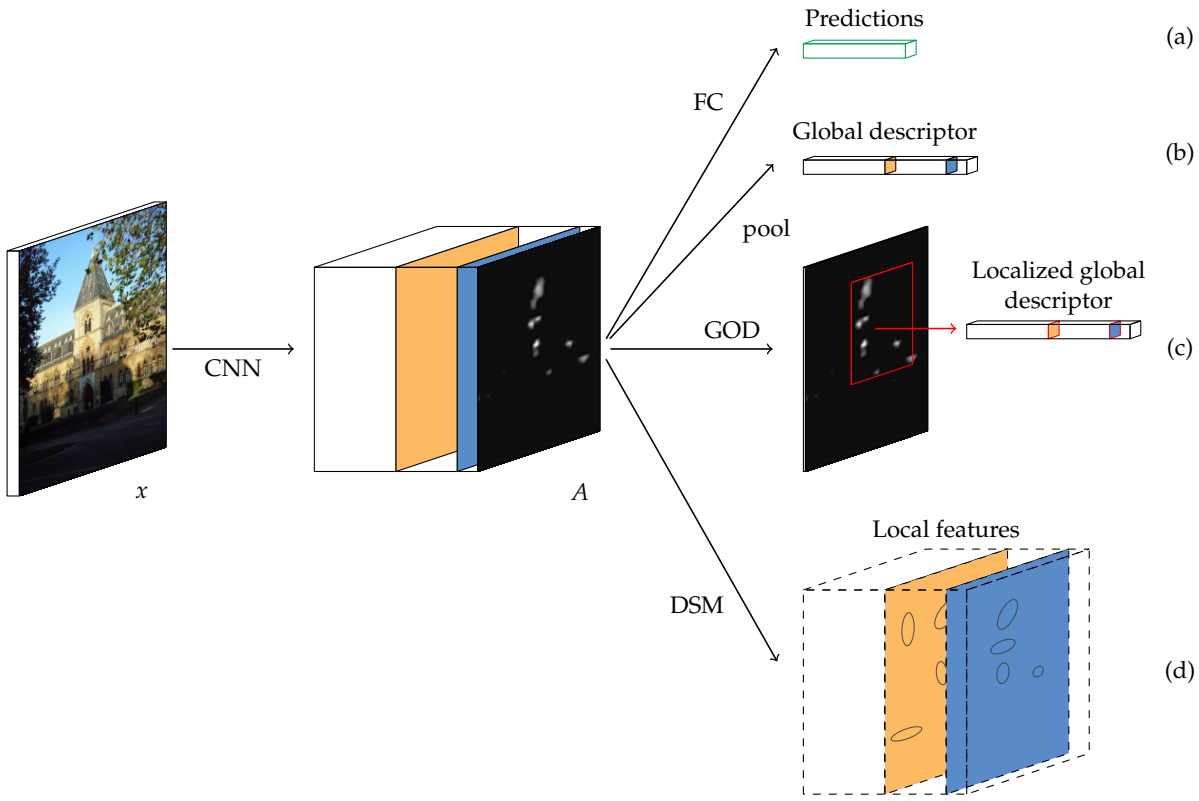


FIGURE 1.3 – Thesis summary. We consider an input image x processed by a fully convolutional backbone network. We show several ways to exploit the resulting feature tensor A that allow the computation of a robust image representation. In order to perform image classification, a vector of predictions can be computed (a) by using a fully-connected (FC) layer (used in Chapter 3). An image global representation can be pooled (b) from the feature tensor (seen in Chapter 4, 5). It is possible to construct a localized global representation (c) using our Graph-based Object Discovery (GOD) method (Chapter 5). At last, (d) shows the output of our local feature extractor presented in Deep Spatial Matching (DSM) (Chapter 4).

between the functions were not significant [GS18; CAL19; Bel+18], and could not help selecting the "best" acquisition function. CNNs appeared only partially dependent on the images they learn on.

Acquisition methods depend on the quality of the used models, so with a small budget we show that selected images are less relevant than a random selection. It therefore appear that performing active learning in the context of deep learning requires further thinking. We propose to include knowledge from unlabeled images, by adding unsupervised and semi-supervised methods to the active learning pipeline. Both types of supervision are known to achieve good results alone and naturally results obtained with such a combination are better. We also propose to always compare to a uniform selection - which is equivalent as not performing active learning at all - in order to have fair comparisons.

We argue that in the context of deep learning, the framing of active learning can be improved.

1.4.2 Pooling

CNN-based global descriptors are constructed by pooling information from convolutional feature maps. Traditionally, spatial details are aggregated into a single value, losing structural information. However, it is possible to go further than performing a simple pooling and to actually keep spatial information. It has been shown that feature maps trained for the classification task are activated on different parts of an image [Zho+16], corresponding to certain characteristical zones. Therefore, using pre-trained networks on manifold learning tasks, it is possible to exploit CNN feature maps and construct saliency maps that can help to localize interesting objects. This discussion, rather new, opens the door to many possibilities to perform weakly-supervised learning using only class labels.

We argue that relevant localization information can be extracted from feature maps at a low cost.

1.4.3 Local features

We investigate how such localization information can be used in the context of *spatial matching*. This task requires images to be represented by a set of local features that can be matched from one image to another. Features are usually characterized by a position, a region and a descriptor that must be invariant to instance-level variations. We propose to extract local features directly within feature maps, almost for "free". We detect *affine* regions in every channel and match them from one images to another by performing efficient Fast Spatial Matching (FSM) [Phi+07]. We force matches to be done between features from the same channel, which enforce that the *semantic* information contained by feature maps is maintained during matching. This semantic matching alleviates the need for descriptors. Our proposed method Deep Spatial Matching (DSM) does not require any additional training and proves to be efficient on networks trained for the classification task.

We argue that local features exploitable for spatial matching can be extracted from feature maps, without requiring any additional training or descriptors.

1.4.4 Unsupervised instance-object mining

We propose a new pooling scheme, which also takes advantage of the information contained in feature maps. We propose to consider the dataset as a whole–structured in a graph–and to mine objects of interest by using the simple concept of repetition. In particular, our

method Graph-based Object Discovery (GOD) discovers depicted objects by combining information contained in CNN feature maps and the dataset graph. We are able to construct *saliency maps* that accurately highlight desired objects. Once they are discovered, we can construct a localized global representation, thus greatly improving global search.

We argue that a dataset can be considered as a whole in order to discover interesting objects, without supervision, before constructing a robust global representation.

1.5 Structure and dependencies

In Chapter 2, we present concepts that will be used in this thesis. We first investigate in Section 2.1 the evolution of CNNs since their first definition to current very proficient architectures. Section 2.2 gathers solutions to maximize a training while using the minimum amount of labeled data. In Section 2.3, we focus on the feature maps generated by the convolutional layers of CNNs and investigate what information can be extracted from them. Section 2.4 presents how to construct a k-NN graph and use it efficiently in the context of computer vision. Finally, in Section 2.5, we present the background of the image retrieval field.

As stated above, minimizing the need in labels is an important challenge. In Chapter 3, we investigate the active learning task, which consists of constructing an acquisition function that selects images to be labeled. In particular we focus on the context of deep learning and show that all acquisition functions give similar results. We propose to add unlabeled images to the training pipeline and show striking results. This method will be published at ICPR 2020 [Sim+20]. The reader is invited to read Section 2.1, presenting CNNs, and Section 2.2, which discusses supervision, in order to get the background necessary to read the chapter.

The remaining part of the thesis focuses on the image retrieval task, and is still concerned with extracting as much information as possible with the least amount of supervision. In Chapter 4, we show that feature maps trained for classification or manifold learning contain localization information accurate enough to detect local features. We also demonstrate that feature maps are semantic enough to get rid of the local features descriptors, allowing features with low-cost representation. Additionally to the work published at CVPR 2019 [SAC19], we give results on tentative experiments to incorporate our spatial matching to a training (in Section 4.5). Section 2.3, 2.4 and 2.5 present background on localization in feature maps, graph theory and image retrieval, respectively, which are needed to read this chapter.

Finally, Chapter 5 introduces an unsupervised technique to construct relevant image representations. We focus on the instance level image retrieval task, which requires finding images depicting the "exact" same object. We consider the database as a whole and utilize the repetition information; objects that appear frequently in the database are most likely the objects of interest. This work was first published at WACV 2018 [Sim+18], and then presented in a longer format for a journal version [Sim+19]. Similarly to the previous chapter, the reader is invited to read Section 2.3, Section 2.4 and Section 2.5 which introduce background on localization in feature maps, on graph theory and image retrieval, respectively.

Chapter 2

Background

Contents

In this chapter, we present concepts that are exploited later in the thesis. In Section 2.1 we describe the key evolution of CNNs that lead to the impressive results achieved today in classification, detection, segmentation and other tasks. CNNs are commonly trained in a supervised manner and require large amounts of labeled data; in Section 2.2 we investigate different solutions to minimize the need for labels. In Section 2.3 we get a closer look at feature maps, which are the outputs of CNN convolutional layers and can provide semantic and localization information "for free". Section 2.4 we present k-NN graph representations, which will be used throughout the thesis. Finally, Section 2.5 is a brief review of the progress of instance-level image retrieval from hand-crafted features to CNNs.

2.1 Convolutional Neural Networks

CNNs have a long history in which they phased in and out of popularity. In 1958, Rosenblatt was investigating brain-inspired mechanisms to perceive the physical world, store the collected information and make decisions based on it. He introduced an early form of neural network: the multilayer perceptron architecture [Ros58] shown in Figure 2.1. Ten years later, Minsky and Papert redefined "perceptron" as a single-layer network and derived a sequence of negative results [MP88] that led to the loss of interest in neural networks, known as the first "winter" of AI.

At the same time, Hubel and Wiesel were experimenting on cats [HW59; HW62] and monkeys [HW68] to understand how living organisms process visual information. They investigated cortical cells and measured their reaction to light pattern stimuli on the retina. The experiments, despite being dreadful from the animals' perspective, have allowed Hubel and Wiesel to understand the role of cortical cells as visual feature detectors. In particular, depending on their reactions they classified cells as *simple*, *complex* and *hypercomplex*.

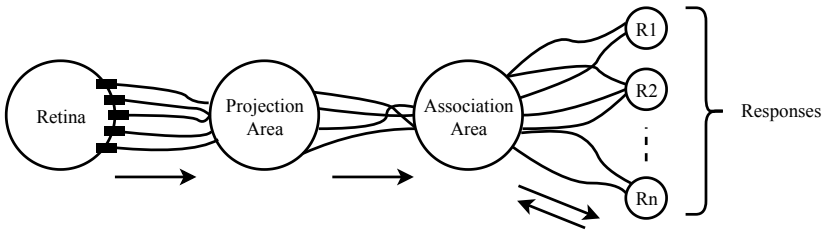


FIGURE 2.1 – The design of a multi-layer perceptron presented by Rosenblatt [Ros58].

This work inspired Fukushima to design the cognitron [Fuk75] and neocognitron [Fuk80] multilayered neural networks respectively in 1975 and 1980. Both shift invariant networks include several layers of simple (S-cells) and complex (C-cells) "cells" organized in a network, performing convolution and pooling operations, respectively. However, neural networks were outperformed by methods using hand-crafted features [HS88; Low99; MS02] (discussed in 2.5) such as k-NN algorithm [CH67] or SVMs [BGV92; CV95; SS02]. The convex SVMs were trained on 30-50 examples; easy to train and explain as opposed to neural networks. Neural networks suffered their second "winter".

A decade later, Rumelhart *et al*. [RHW86] proposed the Back Propagation (BP) algorithm, which LeCun *et al*. used to train the "ConvNet" [LeC+89], a 3-layer CNN in 1989. Increasing the network depth from 3 to 5 layers, LeCun *et al*. presented good results of "LeNet5" [Lec+98] on a realistic Optical Character Recognition (OCR) problem, tested on the now famous MNIST dataset [Lec+98]. They trained the network on batches of images rather than the whole dataset using Stochastic Gradient Descent (SGD) [KW52].

In order to make CNNs training practical at large scale, [CPS06; UB09] fitted networks in GPUs but designed the networks to satisfy hardware constraints, which was not optimal. Then in 2011 and 2012, Ciresan *et al*. [C C+11] and Krizhevsky *et al*. [KSH12] showed that networks could be fully trained on GPUs in an online manner without requiring a particular design. Both works demonstrated the efficiency of CNNs on several benchmarks (NORB [LHB04], CIFAR-10 [Kri09]). In particular, the network "AlexNet" [KSH12] caught attention for improving results by over 10% error-rate on ImageNet Large-Scale Visual Recognition (ILSVRC) 2010 challenge [Rus+15], thanks to the addition of Rectified Linear Unit (ReLU) [NH10] non-linearity and *dropout* [Hin+12] regularization.

Any learning algorithm is prone to overfitting: they tend to learn perfectly the training set while providing poor generalization on unseen data. In order to avoid having to produce new labeled images, several tricks can be used as dropout regularization [KSH12; Hin+12], data augmentation, flipping and cropping [SSP03; C C+11] and multi-scale approaches [Ser+13a].

Following the promising results of deeper CNNs [ZF14], the depth of neural networks has increased to 15 (VGG16 [SZ14]) and 22 learnable layers (GoogleNet [Sze+14]). Those two networks integrated interesting improvements to CNN architecture. VGG [SZ14] proposed to use small kernels (3×3) accross all layers and to iteratively train a network initializing from shallow networks. GoogleNet [Sze+14] is composed of "inception" modules combining information at different resolutions, increasing the number of kernels but at a lower cost. Szegedy *et al*. also integrated "auxiliary classifiers" to the network at different depths to improve the gradient flow. Subsequently, the inception network was updated and improved [Sze+16; Sze+17].

Two years later, network depth was augmented drastically: 152 (ResNet [He+16]) and 264 layers (DenseNet [Hua+17]). Additionally to the overfitting problem, He *et al*. [He+16] showed that deeper networks suffered accuracy "degradation" and vanishing gradients compared to shallower ones. They investigated the impact of depth by defining a deep network that was functionally equivalent to a shallow network (using identity layers) and showed that training was significantly harder. In order to fix this problem, they introduced "residuals" constructed using shortcut connections. Following the intuition that connecting layers helps, Huang *et al*. [Hua+17] proposed "DenseNet", a deep network in which every layer is connected to all layers of matching output resolution. Doing so, the network is encouraged to reuse features and, in some setups, similar performances to ResNet are achieved using 3× less parameters. Following [He+16], Szegedy *et al*. introduced residual connections in Inception, resulting in Inception-ResNet [Sze+17].

The convergence of a deep network training depends greatly on the initialization of the model parameters as shown by Simonyan *et al*. [SZ14]. Random initialization showed to be under-performing. Glorot *et al*. [GB10] proposed to construct initialization in order to control the variance of activations and avoid for a signal to explode or vanish. They introduced the "Xavier" initialization, which scales a uniform distribution to a layer size. He *et al*. [He+15] adapted [GB10] to the presence of non-linearity ReLU in convolutional networks. Training can be further improved by initializing weights using transfer learning methods (discussed in Section 2.2). In order to simplify deep network training, Ioffe and Szegedy proposed to normalize layer inputs using Batch Normalization (BN) [Chr15]. BN allows to use a higher learning rate and sometimes it can replace dropout by acting as a regularizer.

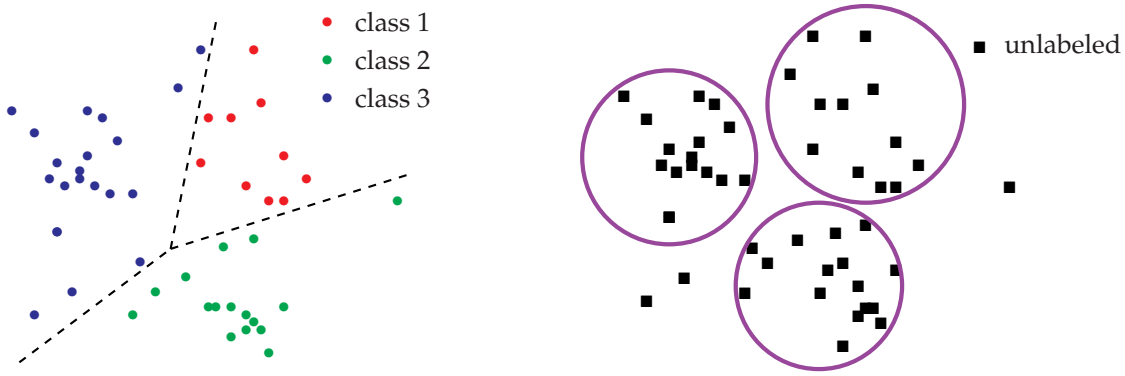
Computer vision classification benchmarks are currently dominated by CNNs, which proved to be very efficient in other tasks such as object detection, segmentation, image retrieval, language and sound processing.

2.2 Minimizing supervision

We have seen that very deep CNN models achieve state-of-the-art results in many computer vision tasks. They are commonly trained in a supervised manner on labeled datasets. Most tasks require training sets that include images and their associated labels (a class, a set of box/class couples or a value for every pixel). This image labeling process commonly requires human labor, which is tedious and alienating. However, the ability of CNN to generalize to new images depends directly on the number and diversity of images seen at training time. In this section, we investigate different solutions to train a CNN for the classification task whilst minimizing supervision.

Unsupervised learning. A first question to consider, is what information can be extracted from unlabeled data and how to perform learning without labels.

Clustering, [JMF99], can be divided into hierarchical [GWW01; GK78; Kur91] and partitional clustering. The latter includes k-means clustering [Llo82] , expectation maximization [DLR77], or spectral clustering [NJW+02; SM97; Nad+05; ZP04]. Efforts have been made to integrate unsupervised clustering to deep networks; in particular, [Bau+16; Dos+14; Lia+16; XGF16; Yan+16] learned jointly CNN features and image clusters using clustering losses. At a large scale, DeepCluster [Car+18a] updated network parameters based on grouped features using k-means clustering in an alternating fashion with model training, while [CN12] performed a sequential bottom-up learning.



(a) Supervised learning (b) Unsupervised learning

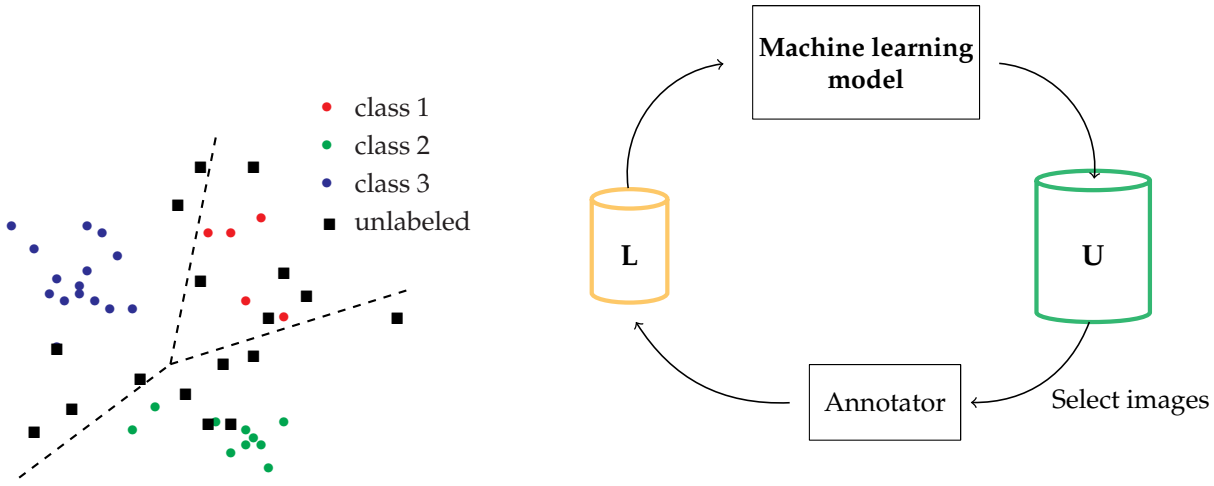
FIGURE 2.2 – Illustration of learning with all labeled data (a) and no labeled data (b).

Self-supervision. Hinton *et al*. [HS06] and Goodfellow *et al*. [Goo+14] showed that data representation can be learned by training a network on a "dummy" task that yields supervision without human involvement. Among others, it is possible to predict relative position of patches [DGE15], to rearrange patches of jigsaw puzzles [NF16], to guess missing pixels [Pat+16], to rotate images [**KoGi18**; FXT19]–helping the network to deal with rotation invariance–, or to predict blur [Alv+19]. Wang *et al*. [WG15] proposed to use videos by training a siamese [CHL05] network on patches extracted by a tracking system on a video.

Generative models. In order to learn image representations, Hinton *et al*. [HS06] proposed to train an *autoencoder* to learn encoding/decoding an image to/from a feature space. Ranzato *et al*. [Ran+07] forced the generated representation to be sparse, while Vincent *et al*. [Vin+08] trained the network with augmented data to be more robust to altered inputs. Alternatively, Generative Adversarial Networks (GANs) [Goo+14; Spr16] simultaneously learn two networks on two competing tasks. The generator learns to generate images that should fool the discriminator classifier, which itself learns to discriminate between "fake" and "real" images. GANs and autoencoder networks can also be associated [DKD16; Dum+16]. Alternatively, [BJ17; Boj+17] proposed to use a reconstruction loss to learn a mapping between random vectors and images in a database.

Transfer learning. As discussed previously, CNN weight initilization impacts greatly performance (in Section 2.1). A line of work proposed to initialize weights using unsupervised methods such as clustering [HOT06; Car+18a; Ser+13b]. Alternatively, it is possible to take advantage of previous efforts of training by performing *transfer learning*. A network *pre-trained* on a supervised task with a large-scale dataset can be successively fine-tuned on another supervised task described by less labeled data [Gir+14; He+17; Sun+17; Mah+18]. As a result, pre-training a network on the large dataset ILSVRC [Rus+15] has become a standard practice in classification [Gir+14; He+16], as well as in other tasks [Ren+15; RTC16; He+17]. Using a larger source domain than ILSVRC (*e.g.* $\times 6$ [He+17], $\times 300$ [Sun+17] or $\times 3000$ [Mah+18]) showed improvements, however suffered from diminishing returns. He *et al*. [HGD18] showed that a long enough training allows to avoid pre-training.

Semi-supervised learning. When a limited amount of labeled data is available, it is often possible to make use of unlabeled images, which are usually easy to find. Not annotated data can be added directly to a supervised training setup [See01; ZG09] as shown in Figure 2.3a. A review of semi-supervised methods before 2009 can be found in [Zhu+06; ZG09]. Semi-supervised learning can be performed using either transductive or inductive



(a) Semi-supervised learning (b) Active learning

FIGURE 2.3 – Illustration of learning with different levels of supervision. Semi-supervised learning combines labeled and unlabeled data (a). Figure (b) presents the process of active learning.

techniques. The last requires a predictive model that can be applied on unseen data, while the latter only focuses on already available unlabeled data and computes label inference based on data statistics [Zho+03a; ZLG03a] *e.g.* using clustering or graphs.

Pseudo-labels. Using a classifier trained in a supervised manner, it is possible to predict labels for unlabeled images using its softmax output [Lee13; Shi+18] or with *label propagation* [ZG02; Zho+03a] on the dataset graph [Isc+19b]. Label propagation has been extensively explored [ZLG03a; Zho+03a; Lon+08]. It takes advantage of a graph structure and the assumption that "similar" images have a high chance to share labels. Given the resulting *pseudo-labels*, the novel class training set is augmented. To reduce the noise, Shi *et al.* [Shi+18] discarded unlabeled images for which a confidence level was too low, while Iscen *et al.* [Isc+19b] proposed a soft selection of unlabeled data by weighting the loss using an uncertainty score.

Unsupervised regularization. Unlabeled data can also be used to regularize a supervised training. In 2008, Weston *et al.* [WRC08] used non-linear embedding algorithms to act as regularizer on different layers of a network. Rasmus *et al.* [Ras+15] adopted a reconstruction loss between one clean and one stochastically-corrupted forward pass of the same input using auto-encoders. Miyato *et al.* [Miy+16] defined the distributional smoothness against local random perturbation as an unsupervised penalty. Chen *et al.* [CZG18] proposed to integrate a memory-assisted module that produces on-the-fly an unsupervised memory loss. A line of work modeled the regularization using a *consistency* loss. Sajjadi *et al.* [SJT16] and Laine *et al.* [LA16a] forced the network to produce consistent outputs compared to outputs of training samples randomly transformed and to the temporal average of outputs, respectively. Tarvainen *et al.* [TV17] focused on the averaged network parameter distribution rather than the averaged outputs. Shi *et al.* [Shi+18] added a contrastive loss to the consistency loss. Inspired by GANs [Goo+14], Miyato *et al.* [Miy+18] proposed a virtual adversarial loss that evaluated local smoothness of the label distribution. Finally, Qiao *et al.* [Qia+18] prevented several networks trained simultaneously to collapse into each other using adversarial examples.

Active learning. When only few images can be labeled, it is beneficial to choose the most "informative" images. *Active learning* requires a system to optimize the selection of examples from an unlabeled pool to be labeled by humans, as shown in Figure 2.3b. Acquisition functions can be designed to find diverse samples in the feature space. The core set technique [GE17; SS18] relies on geometry and uses the Euclidean distance to find the samples that are furthest away from labeled and selected samples. The uncertainty of a classifier is also a relevant information when choosing images [Yan+15b; JPP09; LG13]. One way to define the uncertainty is to measure the entropy of the classifier output probabilities. Following adversarial learning, [GS18; SED19] proposed to use a binary classifier to predict if an image is from the labeled or unlabeled pool. [DP18] added adversarial examples to the labeled set, while [MT18] matched them to the nearest unlabeled example. Different works observed that the difference between results obtained with recent acquisition functions are not significant [GS18; CAL19; Sim+20].

Ensemble and Bayesian methods [GIG17; Bel+18; CAL19] target representing model uncertainty, which can then be used by different acquisition functions. This idea is orthogonal to acquisition strategies. In fact, Beluch *et al*. [Bel+18] and Chitta *et al*. [CAL19] show that the gain of ensemble models is more pronounced than the gain of any acquisition strategy. Of course, ensemble and Bayesian methods are more expensive than single models. Approximations include for instance a single model producing different outputs by dropout [GIG17].

Semi-supervised active learning has a long history [MN98; MMK02; ZLG03a; ZCJ04; Lon+08]. A recent deep learning approach acquires the *least* certain unlabeled examples for labeling and at the same time assigns predicted *pseudo-labels* to *most* certain examples [Wan+17]. This does not always help [DP18]. In some cases, semi-supervised algorithms are incorporated as part of an active learning evaluation [Li+19; SS18]. A comparative study suggests that semi-supervised learning does not significantly improve over active learning, despite its additional cost due to training on more data [GIG17]. However, we showed (Chapter 3) that this is clearly not the case, using the semi-supervised method [Isc+19a].

Few-shot learning (or low-shot learning) [Thr95; MMV00; Lak+11; Koc15] is similar to transfer learning in that knowledge is transferred from a task with large-scale data and supervision to a task with few labeled data. However, even the raw data in the new task are so few (*e.g*. 5-10 examples per class) that fine-tuning a pre-trained model is challenging. If only a single new labeled image is available, the task is called *one-shot* learning[FFP06]. *Zero-shot* learning [Soc+13] differs in that instead of a label, a description is given per image - *e.g*. a binary attribute or text.

Sharing priors. Li Fei-Fei *et al*. [LFP06] and George *et al*. [Geo+17] found priors that can be shared between classes. Designed for simple tasks, Lake *et al*. [LST13; LST15] focused on strokes on the "Omniglot" dataset of handwritten characters [LST13], while Wong *et al*. [WY15] were finding similar patches of MNIST digits. On a more semantic level, [LNH09] focused on finding common attributes between objects of the same class. Alternatively, [TC09] proposed to weight a descriptor to get closer to vectors of well-known class images.

Meta-learning. An ensemble of works [Men+12; Vin+16; SSZ17] were inspired by metric learning (which will be discussed later). In particular, Vinyals *et al*. [Vin+16] argued that using a similar training scenario to a test scenario helped. They trained a neural network, equivalent to a weighted k-NN classifier, on small mini-batches called *episodes*. This formed

the introduction of *meta-learning* [TP98] in the context of CNNs. In each episode, a meta-learner samples a small training and test set from the labeled pool and feeds it to a learner that outputs a classifier. The loss produced by the classifier on the test set is used to train the meta-learner. The episodic few-shot learning process is therefore directly optimized. Also performing metric learning, Snell *et al*. [SSZ17] proposed to construct a prototype per class and to classify a query to its nearest prototype. A line of works focused on optimization [FAL17; FXL18; RL17]. Alternatively, Bertinetto *et al*. [Ber+16] were attempting to predict the parameters of a learner.

Image generation. Another idea is that new labeled images can be generated using dataset statistics. For instance, Miller *et al*. [MMV00] learned a density that arises from a set of transforms. Dixit *et al*. [Dix+17] proposed to perform an attribute-guided augmentation given a pose and attribute labeled dataset. Hariharan *et al*. [HG17] performed feature hallucination, by applying it on novel class image transformations learned on pairs of images. Wang *et al*. [Wan+18] also combined meta-learning with a hallucinator and improved the process by an end-to-end network.

There are also combinations of different tasks. Douze *et al*. [Dou+18] bridged semi-supervised and few-shot learning by using additional unlabeled data in few-shot learning. We have discussed above how semi-supervised can be combined with active learning [Lon+08; Wan+17] and we further explore this combination in Chapter 3. Rebuffi *et al*. [Reb+19] have recently investigated unsupervised pre-training in the context of semi-supervised learning and their findings are consistent with ours (Chapter 3).

2.3 Localization in activation maps

Training on semi-supervised and unsupervised tasks allows to greatly reduce supervision when using CNNs. In this section, we discuss how localization information can be extracted while training on a simple classification task. A CNN designed for classification is often composed of a backbone network - a stack of convolutional layers - and additional layers specific to the task. CNNs give excellent results on classification tasks [KSH12; SZ14; He+16], but how and why is still not well understood. Convolutional layers output feature maps that are activated by some patterns. Their properties vary with depth as shown by Zeiler *et al*. [ZF14] and Springenberg *et al*. [Spr+14]: the first activation maps have small receptive fields and focus on simple patterns while the last maps are coarser and can detect complex concepts. Additionally, convolutional layers are invariant to translation, and when the receptive field is large enough, they are invariant to scale change up to a certain extent [ZF14].

Understanding classification predictions. While trying to improve the performance of CNNs, Zeiler *et al*. [ZF14] proposed to visualize parts of an image that were most significant for a given neuron using a deconvolution [ZTF11] approach. They also discovered discriminative parts of an image to a prediction by occluding the image using a sliding gray patch on the image and observing the class probabilities depending on the patch position. From both those experiments, they showed that it was possible to visualize where the network would focus to make a prediction. Springenberg *et al*. [Spr+14] further investigated visualization and proposed a guided back-propagation approach, in which the back-propagation signal is guided by the deconvolution signal, preventing negative gradients to flow backwards. Results are presented in Figure 2.4. Alternatively, Mahendran *et al*. [MV15] experimented with

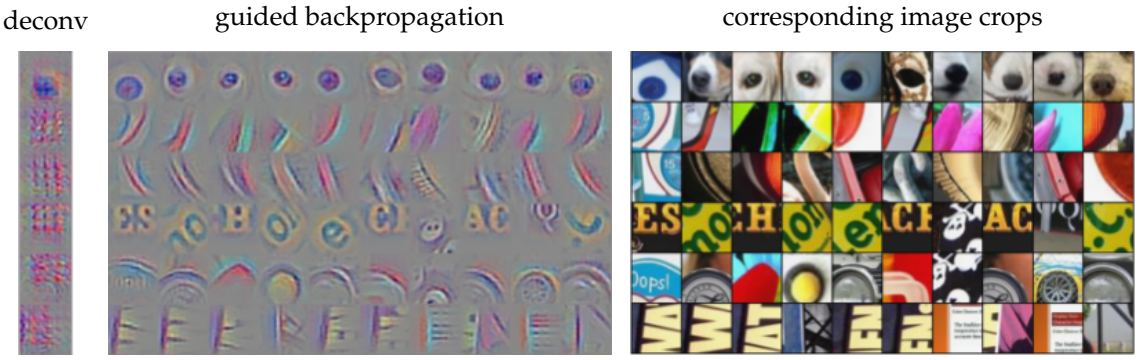


FIGURE 2.4 – Visualization of learned patterns proposed by Springenberg *et al*. [Spr+14]. Most active patches are shown.

the reconstruction of an image given a representation, showing therefore what the representation encodes and what information is preserved by the network. Bazzani *et al*. [Baz+14] localized objects by hiding parts of an image and discovered the one causing the most difference of activation.

Zeiler *et al*. [ZF14] and Springenberg *et al*. [Spr+14] used visualization to interpret CNN results and to improve their network architecture. While doing so, they showed that convolutional kernels are acting as concept detectors even if trained only using class labels.

Weakly supervised localization. Following this discovery, Zhou *et al*. [Zho+15] proposed to train a network on the scene recognition task and perform classification and detection of objects present in a single forward pass. One year later, Zhou *et al*. [Zho+16] exploited visualization efforts to perform accurate localization. They introduced Class Activation Mapping (CAM) that performs a linear combination of the last convolutional layer outputs using weights learned before a fully-connected layer specific to the classification task. Contrary to Zeiler *et al*. [ZF14] and Zhou *et al*. [Zho+15], they used a global average pooling layer in order to maintain spatial information while using fully-connected layers. The generated class activation maps highlight the class-discriminative regions that were used to make predictions, and gave good localization results. Grad-CAM [Sel+17] generalizes this idea to any network architecture and allows visualization at any layer by a similar linear combination on the gradient signal instead. Similarly, Oquab *et al*. [Oqu+15] trained a network on class-level labels and localized objects in class feature maps. However, they used max-pooling as opposed to average-pooling in CAM [Zho+16], which limits localization to a single point rather than encouraging a network to discover the entire object. The selection of a pooling scheme is important and will be latter discussed in Section 2.5 in the context of image retrieval.

Sparsity and localization. Wang *et al*. [Wan+15] performed further experiments on feature maps, and observed the following.

> *"Although the receptive field of CNN feature maps is large, the activated feature maps are sparse and localized. The activated regions are highly correlated to the regions of semantic objects."*

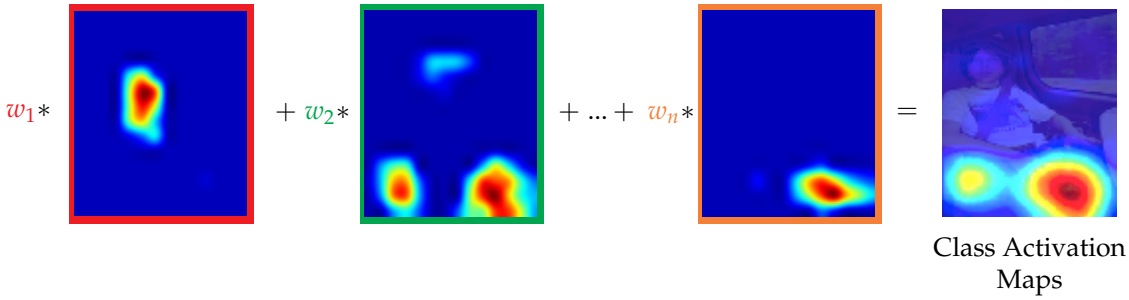


FIGURE 2.5 – CAM [Zho+16]. Extracting localization information.

They constructed a visual tracking system that selects feature maps from convolutional layers using a convolutional attention layer. They noted that the last convolutional layer was responding to semantic objects while earlier layers are focusing on more coarse, but detailed, information (*e.g.* texture) that could help discriminating between semantically equivalent objects. In a class-agnostic manner, Kalantidis *et al.* [KMO16] took advantage of the sparsity of activation maps. Following the intuition that a sparse feature map has a higher chance to focus on a particular object rather than on clutter, they proposed to weight activation map pixels using this information, as well as applying a spatial weight based on pooling over channels.

Local features. CAM [Zho+16] and Wang *et al.* [Wan+15] require the network to be trained on specific classes that will be localized, while in [KMO16] localization is coarse and does not include semantic information. Also, pooling over channels, performed in [Zho+16; KMO16], leads to the construction of a saliency map which is less sparse than feature maps themselves. Dusmanu *et al.* [Dus+19] and Cieslewski *et al.* [CBS19] and us (Chapter 4) discovered semantic information within feature maps without requiring a specific training. All three methods rely on extracting local features that were repeatable from one image to another and could be used for spatial matching. As Cieslewski *et al.* [CBS19], we showed that the semantic information of feature maps was significant enough to represent an image by a set of local features without descriptors; additionally we proved that no specific training was required.

2.4 Graph representation

So far, we have discussed how CNN models can solve recognition problems, giving an answer for one image at a time. However, it may be useful to consider a dataset as a whole and to look at it globally. Indeed, a database of images can be structured in a graph, in which an algorithm can *walk* and discover how images are related. Largely used in web applications, graphs allow the measurement of the importance of each node w.r.t. to the others, as well as comparing them. In particular, given a query image and a database structured in a graph, similar images to the query can be found while walking in the graph. We focus on *k*-NN graphs and introduce concepts that are used in the following chapters.

K-Nearest Neighbor Graph. We consider a graph with n vertices $V = \{\mathbf{v}_1, ..., \mathbf{v}_n\}$, corresponding to n images, with $\mathbf{v}_i$ being the descriptor of the i^{th} image. The graph edges connect k-nearest neigbhors and are weighted using a similarity measure $s(\mathbf{u}, \mathbf{v})$ where $\mathbf{u}, \mathbf{v} \in \mathbb{R}^d$ and $s : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$. The similarity measure is symmetric and positive, so the graph is *undirected*. *k*-NN graphs are particularly suited for image retrieval as a graph can be computed offline (the whole dataset is known). Additionally, there is no need to exhaustively define

relations between images–but only between close/similar ones following k-NN–, processes described later will help discover more accurate .

Iscen *et al*. [Isc+17] proposed a *mutual* k-NN graph, which edges are weighted using the *mutual* k-NN similarity $s_k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$

$$s_k(\mathbf{v}_i, \mathbf{v}_j) = \min(s_k(\mathbf{v}_i|\mathbf{v}_j), s_k(\mathbf{v}_j|\mathbf{v}_i)) \tag{2.1}$$

where $i, j \in [n]$, $(\mathbf{v}_i, \mathbf{v}_j) \in V$, and,

$$s_k(\mathbf{v}_i|\mathbf{v}_j) = \begin{cases} s(\mathbf{v}_i, \mathbf{v}_j) & \text{if } \mathbf{v}_i \in NN_k(\mathbf{v}_j) \\ 0 & \text{otherwise} \end{cases} \tag{2.2}$$

where $NN_k(\mathbf{v})$ are the k nearest neighbors of $\mathbf{v}$ in V. Any other symmetric operation could be used to replace min in 2.1.

Graph representation. An undirected k-NN graph can be represented by the *adjacency matrix* $W \in \mathbb{R}^{n \times n}$ which elements are the edge weights. Considering the $n \times n$ *degree matrix* $D := \text{diag}(W\mathbf{1})$ where $\mathbf{1} \in \mathbb{R}^n$, it is possible to define the *symmetrically normalized adjacency matrix*

$$\mathcal{W} := D^{-1/2} W D^{-1/2}, \tag{2.3}$$

with the convention $0/0 = 0$. The graph $n \times n$ Laplacian and normalized Laplacian are defined as

$$L := D - W \tag{2.4}$$

and

$$\mathcal{L} := D^{-1/2} L D^{-1/2}. \tag{2.5}$$

Iscen *et al*. [Isc+17] introduced a regularized version of the Laplacian,

$$L_\alpha := D - \alpha W \tag{2.6}$$

and its normalized form

$$\mathcal{L}_\alpha := D^{-1/2} L_\alpha D^{-1/2} = I_n - \alpha \mathcal{W}, \tag{2.7}$$

where $0 < \alpha < 1$. Both L_α and $\mathcal{L}_\alpha$ are positive-definite [Chu97; Isc+17; Isc+18a], in particular $\mathcal{L}_\alpha = \alpha\mathcal{L} + (1-\alpha) I_n \succ \alpha\mathcal{L} \succeq 0$.

Scalability. The construction of k-NN graph by brute force is only practicable for small datasets as it has a $O(n^2)$ cost. Indeed, k-NNs have to be found for every object of the graph. Scalability can be achieved using parallel algorithm *e.g*. data partioning schemes [CFS09] or space filling curves [CK10] when using specific similarity measures. Alternatively, Dong *et al*. [DCL11] proposed an approximate k-NN graph, fast to implement and applicable to any similarity measure.

Social network analysis. Social network analysis is a related field of study where graphs have been used to model social relations. Important contributions to the field in the 50's [Kat53; Hub65; See49] have later been applied to other fields including machine learning [SC04] and computer vision. Newman [New10] gathered different measures describing a vertex *importance* in a graph. In particular, he introduced the concept of *centrality* which would help answer the question "What are the most important or *central* vertices in a network?". Such information was primordial when performing social network analysis, but

is also useful in computer vision. We further detail significance measures and a possible exploitation in Chapter 5.

Queries. For the following part of this section, we consider a set of query descriptors in V, represented by the binary vector $\mathbf{g} = (\mathbf{g}_i) \in \mathbb{R}^n$ where $\mathbf{g}_i = 1$ if $\mathbf{v}_i$ for which the system should retrieve all images containing the same object in V.

Diffusion. In the context of image retrieval, the centrality measure is not enough to retrieve elements similar to a query. In particular, Richardson *et al*. [RD01] added the notion of query to the centrality algorithm *PageRank* [Pag+99], by modeling the probability of relevance of a page to a query. Alternatively, similarities can be *diffused* starting from queries, performing *diffusion*. Different diffusion schemes were revisited in [DB13] by Donoser and Bischof; they focused on iterative solutions and argued that closed forms were not computationally doable due to the need to inverse large matrices. Diffusion has been early investigated in the context of semi-supervised classification [ZGL03; Zho+03a] and segmentation [Gra06; KLL08]. We focus this section on the process proposed by Zhou *et al*. [Zho+03b; Zho+03a].

Given a manifold, represented by a graph, and the set of queries Zhou *et al*. [Zho+03b] proposed to diffuse *ranking scores* defined by a ranking score vector $\mathbf{u} \in \mathbb{R}^n$. The iterative diffusion processed was based on the following equation

$$\mathbf{u}^t = \alpha \mathcal{W} \mathbf{u}^{t-1} + (1 - \alpha)\mathbf{g} \tag{2.8}$$

where $\mathcal{W} \in \mathbb{R}^{n \times n}$ is the normalized adjency matrix of the graph, α is a parameter and $\mathbf{g} \in \mathbb{R}^n$.

This iterative process is equivalent to performing a random walk on the graph when $\mathcal{W}$ is a transition matrix and $\mathbf{g}$ is a l^1 unit vector. Indeed, given the probability α the process jumps to an adjacent vertex following $\mathcal{W}$, and with $1 - \alpha$ to a query point selected uniformly at random. In our case however, $\mathcal{W}$ is symmetric.

The system 2.8 converges towards the solution $\mathbf{u}^*$ as shown by [Zho+03a; Zho+03b], with

$$\mathbf{u}^* = (1 - \alpha)\mathcal{L}_\alpha^{-1}\mathbf{g} \tag{2.9}$$

assuming that $0 < \alpha < 1$.

Closed form solution. Iscen *et al*. [Isc+17; Isc+18a] introduced a fast iterative approximate solution of the closed form. The authors proposed to construct a graph using *mutual* k-NN similarity (2.1) implying performing a locally constrained random walk [KDB09]. They speed up the process by approximating the solution

$$\mathcal{L}_\alpha \mathbf{f} = (1 - \alpha)\mathbf{g} \tag{2.10}$$

using Conjugate Gradient (CG) [NW06], which takes advantage of $\mathcal{L}_\alpha$ being positive-definite.

Additionally, Iscen *et al*. augmented the diffusion to use queries not included in the database V. They did not augment the existing graph at query time, as proposed by Zhang *et al*. [Zha+12], but found the k-NN of the query and performed the search with those neighbors. They also modified diffusion so that it could be applied to regional representation, using a diffusion process starting from several queries.

It can be noticed that depending on the form of $\mathbf{g}$, it is possible to perform diffusion ($\mathbf{g} \in \mathbb{R}^n$) (used in Chapter 4, 5), computation of centrality ($\alpha = 0$, $\mathbf{g} = \mathbf{1} \in \mathbb{R}^n$) discussed in Subsection 5.3.7 and label propagation (using matrices $U \in \mathbb{R}^{n \times n}$ and $G \in \mathbb{R}^{n \times n}$ instead of vectors $\mathbf{u}$ and $\mathbf{g}$) [Isc+19b] (used in Chapter 3). This diffusion process has been used in many different scenarios including retrieval [Isc+17; Isc+18a], semi-supervised learning [Isc+19b] and our active learning framework presented in Chapter 3.

2.5 Image retrieval

In this section, we focus on the instance-level image retrieval task. We discuss different solutions from early hand-crafted features to CNN-based methods. In this task, we are given an image, called a *query*, which depicts an object, or a scene. Given this query image and no other information, the system must retrieve from a database, a ranked list of images sorted by similarity to the query. An important difficulty lies in the fact that queried object of interest may vary from one image to another in viewpoint, lighting conditions or can be occluded.

Evaluation metric. The mean Average Precision (mAP) [Phi+07] (average of Average Precision (AP) over all queries) is commonly used to evaluate retrieval performances. The AP measure corresponds to the area under the precision/recall curve and allows to describe results combining the accuracy of results (*precision*) and checking how many of the expected elements are returned (*recall*). Additionally, the mean Precision at rank K (mP@K) [Rad+18] can be used to evaluate the sorting precision.

Hand-crafted features. For several decades, images have been represented by hand-crafted features, which are computed directly on the pixels. Those features are computed in 2 to 3 steps. First, local features are *detected* in the image, they are *described* by a descriptor and finally they may be *aggregated* into a single descriptor.

Local feature detection. Early detectors were focusing on corner and edges [Mor80; HS88] looking for maximum of change between pixels. The Difference of Gaussians (DoG) [Low99] detector efficiently approximates the Laplacian of Gaussian (LoG) [Lin98] to detect local features at different scales. Mikolajczyk *et al.* [MS01] proposed the *Harris-Laplace* detector, an extension of the Harris detector [HS88] that integrates a scale selection using LoG [Lin98]. In order to deal with images varying by affine transformation, Mikolajczyk and Schmid proposed the *Harris-Affine* [MS02] which includes an additional shape adaptation following [LG97] to every detected keypoint. The detector has been further improved by using the Hessian matrix, resulting in the *Hessian-Laplace* [MS04]. Maximally Stable Extremal Region (MSER) [Mat+02] differs from previously presented work as it is not based on pixel changes, but rather on pixel intensity.

Local descriptors. One widely used representation method is SIFT [Low99] which computes for each keypoint a canonical orientation and a 128-descriptor constructed by concatenating orientation histograms of subpatches around the keypoint. This work has been followed by improvements including Root-SIFT [AZ12] or the faster scheme SURF [Bay+08], descriptors based on histogram of gradients as HOG [DT05] or CHOG [Cha+09], or binary descriptions including BRIEF [Cal+10].

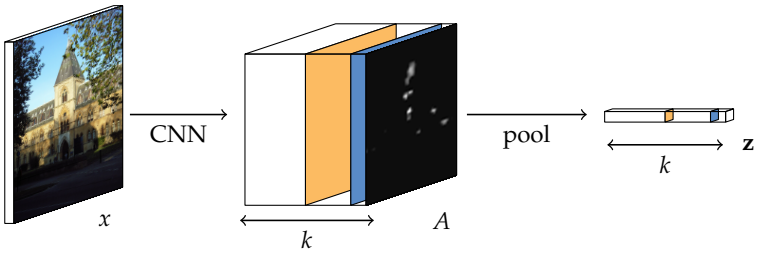


FIGURE 2.6 – Pooling. We consider an input image x processed by a fully convolutional backbone network. The resulting tensor $A \in \mathbb{R}^{h \times w \times k}$ can be pooled in a global descriptor $\mathbf{z} \in \mathbb{R}^k$ using either Sum-pooled Convolutional (SPoC) [BL15], Maximum Activation of Convolutions (MAC) [Azi+14; TSJ16] or Generalized-Mean (GeM) [RTC18] pooling.

Aggregation. In order to perform a fast global search, each image needs to be represented by a unique global descriptor. The local descriptors presented above must be aggregated into a global representation. State-of-the-art solutions require to construct a dictionary of *visual words*, called a *codebook*, using for instance k-means clustering [Llo82], Gaussian Mixture Model (GMM) or Expanding Gaussian Mixture (EGM) [AK12]. The BoW technique was first introduced in the context of text understanding [SM83], and then adapted to computer vision [SZ03; Csu+04]. A histogram of assigments to the codebook is constructed and used as image representation. Vector of Locally Aggregated Descriptors (VLAD) [Jég+10] and Fisher Vectors (FVs) [PD07] integrate more statistics of the codewords, and in particular the divergence of each visual word from the assigned descriptors, using respectively 1^{st} and a combination of 1^{st} and 2^{nd} moments. Among many improvements of those aggregation techniques, the cell of a BoW visual word is further devided in Hamming Embeddings [JDS08; JDS10]. In particular, local descriptors assigned to the same visual word are compared using the Hamming distance after being quantized using a binary signature. While Aggregated Selective Match Kernel (ASMK) [TAJ13; TAJ16] is an improved version of VLAD which inludes non-linearity.

CNN in retrieval. Following the breakthrough of CNNs on the classification task and qualitative results of deep networks on image retrieval [KSH12], the retrieval community has introduced convolutional networks in their pipeline. The first positive experiments with CNNs for retrieval were performed using the fully-connected layer outputs [Bab+14; Gon+14], allowing for one descriptor per image (or patch). Further experiments on convolutional outputs showed convolutional representations to give better performances [Raz+16; Moh+16; CMV15], as well as allowing to extract either global or local features. Local descriptors obtained with CNNs can be aggregated using classic aggregation methods such as VLAD [Gon+14], FV [CMV15] or BoW [Moh+16].

Retrieval at large scale is challenging both in terms of search and memory footprint. Global descriptors are therefore very interesting for the community. Hand-crafted global descriptors can be computed [OT01; TFW08] directly from pixels, but they were outperformed by aggregated local descriptors. Alternatively, is it possible to construct CNN global representations that achieve better results than hand-crafted aggregated descriptors (*e.g.* VLAD) with the same dimensions and memory cost [TSJ16; RTC18].

Global pooling. Feature maps can be compressed into a global representation using simple pooling schemes as shown in Figure 2.6. We denote by $A \in \mathbb{R}^{h \times w \times k}$ a tensor containing k feature maps of spatial resolution (h, w). The tensor can be represented by a descriptor $\mathbf{z} = [z_1 \cdots z_j \cdots z_k] \in \mathbb{R}^k$ with $j \in [k]$ where each z_j summarizes information from channel

$A^{\bullet j}$. We define $P := [h] \times [w]$ the set of positions in a feature map. In particular, following SPoC [BL15] it is possible to average each channel into a value using

$$z_j^{\text{SPoC}} = \frac{1}{|P|} \sum_{p \in P} A^{pj} \tag{2.11}$$

with $j \in [k]$. Better results were achieved using max-pooling MAC [Azi+14; TSJ16]

$$z_j^{\text{MAC}} = \max_{p \in P} A^{pj} \tag{2.12}$$

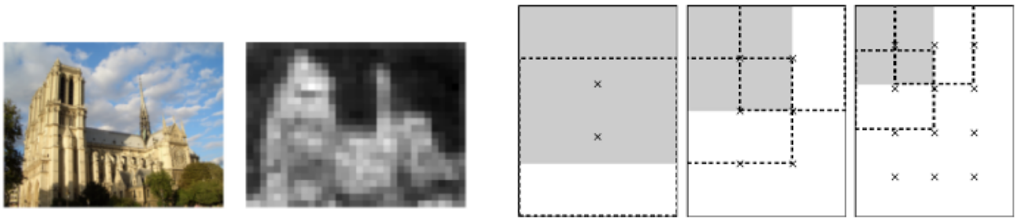
or GeM pooling [RTC18]

$$z_j^{\text{GeM}} = \left(\frac{1}{|P|} \sum_{p \in P} A^{pj^\omega} \right)^{\frac{1}{\omega}}, \tag{2.13}$$

which depends on a parameter ω learned a training. Average pooling has the issue of giving the same power to each pixel, therefore background and objects contribute equivalently to the pooling, while MAC focuses on a single pixel value per feature map. GeM is an intermediate solution, leveraging high value pixels, more or less depending on p_k. Alternatively, the NetVLAD network [Ara+16] aggregates all vectors $A^{p\bullet} \in \mathbb{R}^k$ of the tensor A with $p \in P$ using a differentiable VLAD layer [Jég+10]; a soft-assignment is used in order for the layer to be differentiable.

Region-based pooling. Another line of work is using regions. In Regional Maximum Activation of Convolutions (R-MAC) [TSJ16], Tolias *et al.* divide the image in R sampled regions following Figure 2.7b. A representation is computed per region by applying MAC pooling on the cropped feature maps. Resulting local descriptors are l^2-normalized and whitened (discussed later). The regions are then summed into a global representation. Other works have replaced uniformly sampled regions by Regions of Interest (RoI) produced by the class-agnostic Region Proposal Network (RPN) [Sal+16; Gor+16a; Son+17]. The RPN is part of the Faster-Region-CNN (R-CNN) detector [Ren+15]; it produces for every image hundreds of RoI proposals with an objectness score. Teichmann *et al.* [Tei+19] proposed to train a model to predict few bounding boxes, as opposed to the hundreds produced by a RPN, and aggregated regions into a single descriptor. At a higher cost, it is possible to perform regional *cross-matching*; Tolias *et al.* [TSJ16] and Salvador *et al.* [Sal+16] both accumulated the highest similarity scores when comparing all database local image descriptors to every region of every query.

Saliency-based pooling. An image does not only depict the objects of interest, but also many irrelevant elements to the search, such as sky, grass or background. An ensemble of works tried to eliminate such information by using saliency maps. Cross-dimensional Weighting and Pooling (CroW) [KMO15] is a weighted pooling scheme which uses data statistics and successfully removes a major part of image noise (Figure 2.7a). Channels are weighted using *sparsity* - a sparse feature map has a higher chance to focus on a particular object, rather than on clutter, and pixels are weighted by their importance over channels. Laskar *et al.* [LK17] used a similary scheme to weigh each R-MAC region. Our work, GOD (Chapter 5) includes the computation of a saliency map per image combining CroW and object repetition information. Candidate regions are gathered in a graph and repeating objects are spotted using the centrality measure. It is also possible to learn a saliency map within a CNN network by training an attention map following DEep Local Features (DELF) [Noh+17].



(a) (b)

FIGURE 2.7 – Figure (a) presents an image and the resulting saliency map when applying CroW [KMO15] weighting scheme. Figure (b) presents R-MAC [TSJ16] sampling scheme.

Architectures and losses. There are several solutions to train a CNNs for the instance-level retrieval task. On one hand, it is possible to train a classifier, using a cross-entropy loss, on a training set with images and labels specifically selected to correspond to retrieval benchmarks [Bab+14]. On the other hand, a line of work is inspired by *metric learning* [Xin+02] which consists in discovering what objects are *similar* or *dissimilar*. This is based on Chopra *et al*. [CHL05], who used a siamese architecture to learn on positive and negative pairs and [HCL06] who introduced the constrative loss. Hoffer *et al*. [HA14] and Wang *et al*. [Wan+14a] augmented the two images architecture to an *anchor*, *negative*, *positive* triple architecture. Applicable directly to image retrieval, Gordo *et al*. [Gor+16a; Gor+17] used a triplet network tuned using a *triplet loss*. In order to deal with a noisy training set, Arandjelović *et al*. [Ara+16] used a weakly supervised triplet loss. Radenović *et al*. [RTC16; RTC18] showed that a contrastive loss gives better performances because overfitting less.

Training sets. Previously described networks are trained in a supervised manner. They require datasets composed of a large number of *matching* and *non-matching* images. Interestingly, it is possible to construct such annotated dataset with minimal human effort. Arandjelović *et al*. [Ara+16] extracted images from Google Street View and labeled them using GPS information. This dataset, although cheap to construct, is noisy: a picture with a certain GPS coordinate can depict one side of a street or another. Gordo *et al*. [Gor+16a] used the processed Landmark [Bab+14] dataset. They constructed pairs of images by performing spatial matching, following the first-to-second neighbor ratio rule [Low99], using SIFT [Low99] and Hessian-Affine [MS02] features. Radenović *et al*. [RTC16] constructed a dataset, from unlabeled images, following Schönberger *et al*. [Sch+15] that coupled BoW [SZ03] with Structure from Motion (SfM) [HZ03]. They constructed clusters and a *3D* model per cluster, allowing to select *hard positives* as inspired by [Sim+14]. D2-NET [Dus+19] was trained on MegaDepth [LS18], a dataset of scenes constructed using internet images and COLMAP [SF16a]. Teichmann *et al*. [Tei+19] proposed a dataset of landmarks labeled by bounding boxes focusing on the objects of interest allowing to train detectors.

Whitening. One of the major constraints in image retrieval is memory. Indeed, representations must be as *compact* as possible. In the work on VLAD [Jég+10], Jégou *et al*. showed that applying Principal Component Analysis (PCA) [FRS01] not only allows to reduce descriptors dimension, but also improves retrieval performances. Inspired by this work, Jégou *et al*. [JC12] theoretically explained how PCA improves a representation by down-weighting co-occurences. This post-processing step has become standard practice in image retrieval, in particular for CNN-based descriptors [TSJ16; BL15; RTC16]. Whitening can be integrated in a deep network [Gor+16a] modeled using a fully-connected layer. Alternatively, Radenović *et al*. [RTC18] discussed the instability of PCA whitening [JC12] and proposed to learn *explicitly* a robust whitening using the same supervision as in network learning.

Query expansion. A retrieval search is dependent on the quality of the query image and its description. If the image is cluttered, if the object is occluded, or the lighting conditions are bad, then the descriptor may be of low quality resulting in a poor search. One way to overcome this problem is to *expand* a query representation using reliable additional information. QE has been first introduced by Manning *et al*. [MRS08] for information retrieval. In 2007, Chum *et al*. [Chu+07] applied it to the visual domain and proposed the Average Query Expansion (AQE) method: a query descriptor is aggregated with descriptors of trusted similar images chosen using BoW and geometric verification (see Section 4.2). Moreover, several efforts have been made to improve the algorithm [JB09; Chu+11; AZ12]. Getting rid of costly spatial verification, Qin *et al*. [Qin+11] proposed Hello Neighbors (HN) and Tolias and Jégou introduced Hamming Query Expansion (HQE) [TJ14] which relies on Hamming Embedding (HE) [JDS10] quantization. Following [TAJ13], a global descriptor was computed per image but expanded to reliable image local descriptors.

Initially computed with hand-crafted features, AQE has also been applied to CNN features [TSJ16; KMO16; Gor+16a]. Moreover, the high quality of CNN global representations has allowed to perform more extension methods. In particular, Iscen *et al*. [Isc+17; Isc+18b] proposed a diffusion process on a graph of global descriptors (details are given in Section 2.4). In a nutshell, similarity scores are diffused starting from the query images through neighbors. Chang *et al*. [Cha+19] also used a graph: they proposed a process that alternates exploitation–retrieve k-NN to the query which edge weights are higher than a threshold–and exploitation–traverse neighbors of retrieved images–of a graph.

Chapter 3

Revisiting active learning

Contents

We first try to minimize the need for supervision by selecting the most interesting images for training a CNN, using *active learning*. Active learning typically focuses on training a model on few labeled examples alone, while unlabeled ones are only used for acquisition. In this chapter, we depart from this setting by using both labeled and unlabeled data during model training across active learning cycles. We do so by using unsupervised feature learning at the beginning of the active learning pipeline and semi-supervised learning at every active learning cycle, on all available data. The former has not been investigated before in active learning, while the study of latter in the context of deep learning is scarce and recent findings are not conclusive with respect to its benefit. The idea developed here is orthogonal to acquisition strategies by using more data, much like ensemble methods use more models. By systematically evaluating on a number of popular acquisition strategies and datasets, we find that the use of unlabeled data during model training brings a spectacular accuracy improvement in image classification, compared to the differences between acquisition strategies. We thus explore smaller label budgets, even one label per class.

We introduce our work [Sim+20] in Section 3.1 and present a background in Section 3.2. We detail our proposition to integrate unlabeled data in Section 3.3 and study relations between acquisition function and label propagation in Section 3.4. We present experiments in Section 3.5 and propose a study of the difference between acquisition functions in Section 3.6. Finally, we dicuss our work in Section 3.7.

3.1 Introduction

Active learning [Set09] is an important pillar of machine learning but it has not been explored much in the context of *deep learning* until recently [GIG17; Bel+18; Wan+17; GE17;

SS18]. The standard active learning scenario focuses on training a model on few labeled examples alone, while unlabeled data are only used for acquisition, *i.e.*, performing inference and selecting a subset for annotation. This is the opposite of what would normally work well when learning a deep model from scratch, *i.e.*, training on a lot of data with some loss function that may need labels or not. At the same time, evidence is being accumulated that, when training powerful deep models, the difference in performance between acquisition strategies is small [GS18; CAL19; Bel+18].

In this work, focusing on image classification, we revisit active deep learning with the seminal idea of using all data, whether labeled or not, during model training at each active learning cycle. This departs from the standard scenario in that unlabeled data are now directly contributing to the cost function being minimized and to subsequent parameter updates, rather than just being used to perform inference for acquisition, whereby parameters are fixed. We implement our idea using two principles: *unsupervised feature learning* and *semi-supervised learning*. While both are well recognized in deep learning in general, we argue that their value has been unexplored or underestimated in the context of deep active learning.

Unsupervised feature learning or *self-supervised learning* is a very active area of research in deep learning, often taking the form of pre-training on artificial tasks with no human supervision for representation learning, followed by supervised fine-tuning on different target tasks like classification or object detection [DGE15; WG15; GSK18; Car+18b]. To our knowledge, all deep active learning research so far considers training deep models from scratch. In this work, we perform unsupervised feature learning on all data once at the beginning of the active learning pipeline and use the resulting parameters to initialize the model at each active learning cycle. Relying on Caron *et al*. [Car+18b], we show that such unsupervised pre-training improves accuracy in many cases at little additional cost.

Semi-supervised learning [CSZ06] and active learning can be seen as two facets of the same problem: the former focuses on *most* certain model predictions on unlabeled examples, while the latter on *least* certain ones. Combined approaches appeared quite early [MN98; ZLG03b]. In the context of deep learning however, such combinations are scarce [Wan+17] and have even been found harmful in cases [DP18]. It has also been argued that the two individual approaches have similar performance, while active learning has lower cost [GIG17]. In the meantime, research on deep semi-supervised learning is very active, bringing significant progress [TV17; LA17; Isc+19a; Ver+19]. In this work, we use semi-supervised learning on all data at every active learning cycle, replacing supervised learning on labeled examples alone. Relying on Iscen *et al*. [Isc+19a], and contrary to previous findings Wang *et al*. [Wan+17] and Gal *et al*. [GIG17], we show that this consistently brings a dramatic accuracy improvement. Related work has been discussed in Section 2.2.

Since Iscen *et al*. [Isc+19a] uses *label propagation* [Zho+03a] to explore the manifold structure of the feature space, an important question is whether it is the manifold similarity or the use of unlabeled data during model training that actually helps. We address this question by introducing a new acquisition strategy that is based on label propagation.

In summary, we make the following contributions:

- We systematically benchmark a number of existing acquisition strategies, as well as a new one, on a number of datasets, evaluating the benefit of unsupervised pre-training and semi-supervised learning in all cases.

- Contrary to previous findings, we show that using unlabeled data during model training can yield a dramatic gain compared to differences between acquisition strategies.
- Armed with this finding, we explore a smaller budget (fewer labeled examples) than prior work, and we find that the random baseline may actually outperform all other acquisition strategies by a large margin in cases.

3.2 Problem formulation and background

Problem. We are given a set $X := \{\mathbf{x}_i\}_{i\in\mathcal{I}} \subset \mathcal{X}$ of *n examples* where $\mathcal{I} := [n] := \{1, \dots, n\}$ and, initially, a collection $\mathbf{y}_0 := (y_i)_{i\in L_0}$ of *b labels* $y_i \in C$ for $i \in L_0$, where $C := [c]$ is a set of *c* classes and $L_0 \subset \mathcal{I}$ a set of indices with $|L_0| = b \ll n$. The goal of *Active Learning (AL)* [Set09] is to train a classifier in *cycles*, where in cycle $j = 0, 1, \dots$ we use a collection $\mathbf{y}_j$ of labels for training, and then we *acquire* (or *sample*) a new batch S_j of indices with $|S_j| = b$ to label the corresponding examples for the next cycle $j + 1$. Let $L_j := L_{j-1} \cup S_{j-1} \subset \mathcal{I}$ be the set of indices of *labeled* examples in cycle $j \geq 1$ and $U_j := \mathcal{I} \setminus L_j$ the indices of the *unlabeled* examples for $j \geq 0$. Then $\mathbf{y}_j := (y_i)_{i\in L_j}$ are the labels in cycle j and $S_j \subset U_j$ is selected from the unlabeled examples. To keep notation simple, we will refer to a single cycle in the following, dropping subscripts j.

Classifier learning. The *classifier* $f_\theta : \mathcal{X} \to \mathbb{R}^c$ with parameters θ, maps new examples to a vector of probabilities per class. Given $\mathbf{x} \in \mathcal{X}$, its *prediction* is the class of maximum probability

$$\pi(\mathbf{p}) := \arg\max_{k\in C} p_k, \tag{3.1}$$

where p_k is the k-th element of vector $\mathbf{p} := f_\theta(\mathbf{x})$. As a by-product of learning parameters θ, we have access to an *embedding function* $\phi_\theta : \mathcal{X} \to \mathbb{R}^d$, mapping an example $\mathbf{x} \in \mathcal{X}$ to a feature vector $\phi_\theta(\mathbf{x})$. For instance, f_θ may be a linear classifier on top of features obtained by ϕ_θ.

In a typical AL scenario, given a set of indices L of labeled examples and labels $\mathbf{y}$, the *parameters* θ of the classifier are learned by minimizing the cost function

$$J(X, L, \mathbf{y}; \theta) := \sum_{i\in L} \ell(f_\theta(\mathbf{x}_i), y_i), \tag{3.2}$$

on labeled examples $\mathbf{x}_i$ for $i \in L$, where *cross-entropy* $\ell(\mathbf{p}, y) := -\log p_y$ for $\mathbf{p} \in \mathbb{R}^c_+$, $y \in C$.

Acquisition. Given the set of indices U of unlabeled examples and the parameters θ resulting from training, one typically acquires a new batch by initializing $S \leftarrow \emptyset$ and then greedily updating by

$$S \leftarrow S \cup \{a(X, L \cup S, U \setminus S, \mathbf{y}; \theta)\} \tag{3.3}$$

until $|S| \geq b$. Here a is an *acquisition* (or *sampling*) function, each time selecting one example from $U \setminus S$. For each $i \in S$, the corresponding example $\mathbf{x}_i$ is then given as query to an *oracle* (often a human expert), who returns a label y_i to be used in the next cycle.

Geometry. Given parameters θ, a simple acquisition strategy is to use the geometry of examples in the feature space $\mathcal{F}_\theta := \phi_\theta(\mathcal{X})$, without considering the classifier. Each example

$\mathbf{x}_i$ is represented by the feature vector $\phi_\theta(\mathbf{x}_i)$ for $i \in \mathcal{I}$. One particular example is the function [GE17; SS18]

$$a(X, L, U, \mathbf{y}; \theta) := \arg\max_{i \in U} \min_{k \in L} \|\phi_\theta(\mathbf{x}_i), \phi_\theta(\mathbf{x}_k)\|, \tag{3.4}$$

each time selecting the unlabeled example in U that is the most distant to its nearest labeled or previously acquired example in L. Such geometric approaches are inherently related to *clustering*. For instance, k-means++ [AV07] is a probabilistic version of (3.4).

Uncertainty. A common acquisition strategy that considers the classifier is some measure of uncertainty in its prediction. Given a vector of probabilities $\mathbf{p}$, one such measure is the *entropy*

$$H(\mathbf{p}) := -\sum_{k=1}^{c} p_k \log p_k, \tag{3.5}$$

taking values in $[0, \log c]$. Given parameters θ, each example $\mathbf{x}_i$ is represented by the vector of probabilities $f_\theta(\mathbf{x}_i)$ for $i \in \mathcal{I}$. Then, acquisition is defined by

$$a(X, L, U, \mathbf{y}; \theta) := \arg\max_{i \in U} H(f_\theta(\mathbf{x}_i)), \tag{3.6}$$

effectively selecting the b *most uncertain* unlabeled examples for labeling.

Pseudo-labels. It is possible to use more data than the labeled examples while learning. In Wang *et al*. [Wan+17] for example, given indices L, U of labeled and unlabeled examples respectively and parameters θ, one represents example $\mathbf{x}_i$ by $\mathbf{p}_i := f_\theta(\mathbf{x}_i)$, selects the *most certain* unlabeled examples

$$\hat{L} := \{i \in U : H(\mathbf{p}_i) \leq \epsilon\}, \tag{3.7}$$

and assigns pseudo-label $\hat{y}_i := \pi(\mathbf{p}_i)$ by (3.1) for $i \in \hat{L}$. The same cost function J defined by (3.2) can now be used by augmenting L to $L \cup \hat{L}$ and $\mathbf{y}$ to $(\mathbf{y}, \hat{\mathbf{y}})$, where $\hat{\mathbf{y}} := (\hat{y}_i)_{i \in \hat{L}}$. This augmentation occurs once per cycle in Wang *et al*. [Wan+17]. This is an example of active *semi-supervised* learning.

Transductive label propagation [Zho+03a] refers to graph-based, semi-supervised learning. Following notations presented in Section 2.4, a nearest neighbor graph of the dataset X is used, represented by a symmetric non-negative $n \times n$ *adjacency* matrix W with zero diagonal. This matrix is symmetrically normalized as $\mathcal{W} := D^{-1/2} W D^{-1/2}$ (2.3), where $D := \operatorname{diag}(W\mathbf{1})$ is the *degree* matrix and $\mathbf{1}$ is the all-ones vector. The given labels $\mathbf{y} := (y_i)_{i \in L}$ are represented by a $n \times c$ zero-one matrix $Y := \chi(L, \mathbf{y})$ where row i is a c-vector that is a *one-hot* encoding of label y_i if example $\mathbf{x}_i$ is labeled and zero otherwise,

$$\chi(L, \mathbf{y})_{ik} := \begin{cases} 1, & i \in L \wedge y_i = k, \\ 0, & \text{otherwise} \end{cases} \tag{3.8}$$

for $i \in \mathcal{I}$ and $k \in C$. Given the normalized regularized Laplacian $\mathcal{L}_\alpha := I_n - \alpha\mathcal{W}$ (2.7) where $\alpha \in [0, 1)$ is a parameter, Zhou *et al*. [Zho+03a] define the $n \times c$ matrix $P := \eta[h(Y)]$[1], where

$$h(Y) := (1 - \alpha)\mathcal{L}_\alpha^{-1} Y. \tag{3.9}$$

[1] We denote by $\eta[A] := \operatorname{diag}(A\mathbf{1})^{-1} A$ and $\eta[\mathbf{a}] := (\mathbf{a}^\top \mathbf{1})^{-1} \mathbf{a}$ the (row-wise) ℓ_1-normalization of nonnegative matrix A and vector $\mathbf{a}$ respectively.

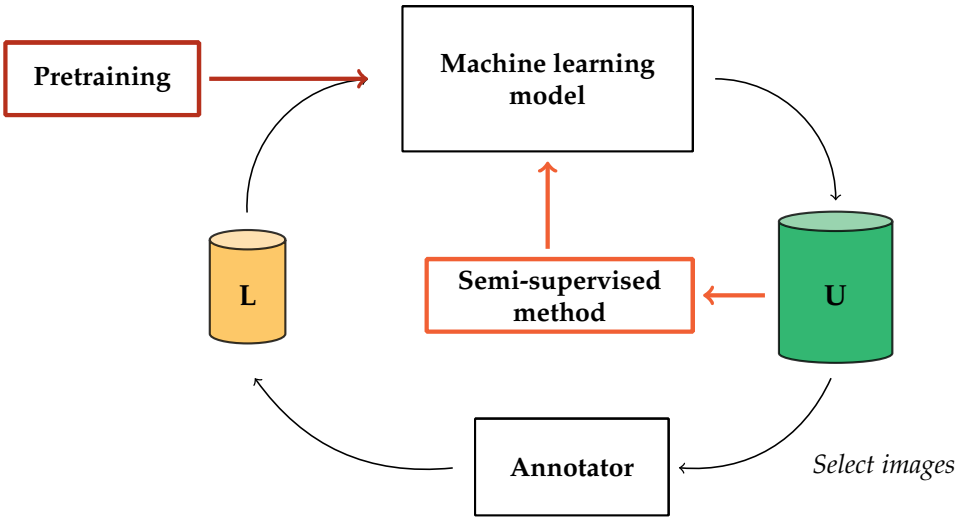


FIGURE 3.1 – Illustration of the active learning process including our addition of unsupervised and semi-supervised learning.

The i-th row $\mathbf{p}_i$ of P represents a vector of class probabilities of unlabeled example $\mathbf{x}_i$, and a prediction can be made by $\pi(\mathbf{p}_i)$ (3.1) for $i \in U$. This method is *transductive* because it cannot make predictions on previously unseen data without access to the original data X.

Inductive label propagation. Although the previous methods do not apply to unseen data by themselves, the predictions made on X can again be used as pseudo-labels to train a classifier. This is done in [Isc+19a], applied to semi-supervised learning. Like Wang *et al.* [Wan+17], a pseudo-label is generated for unlabeled example $\mathbf{x}_i$ as $\hat{y}_i := \pi(\mathbf{p}_i)$ by (3.1), only now $\mathbf{p}_i$ is the i-th row of the result P of label propagation according to (3.9) rather than the classifier output $f_\theta(\mathbf{x}_i)$. Unlike Wang *et al.* [Wan+17], *all* unlabeled examples are pseudo-labeled and an additional cost term $J_w(X, U, \hat{\mathbf{y}}; \theta) := \sum_{i \in U} w_i \ell(f_\theta(\mathbf{x}_i), \hat{y}_i)$ applies to those examples, where $\hat{\mathbf{y}} := (\hat{y}_i)_{i \in U}$ and $w_i := \beta(\mathbf{p}_i)$ is a weight reflecting the *certainty* in the prediction of $\hat{y}_i$:

$$\beta(\mathbf{p}) := 1 - \frac{H(\mathbf{p})}{\log c}. \tag{3.10}$$

Unlike Wang *et al.* [Wan+17], the graph and the pseudo-labels are updated *once per epoch* during learning in Iscen *et al.* [Isc+19a], where there are no cycles.

Unsupervised feature learning. Finally, it is possible to train an embedding function in an unsupervised fashion. A simple method that does not make any assumption on the nature or structure of the data is Caron *et al.* [Car+18b]. Simply put, starting by randomly initialized parameters θ, the data $\phi_\theta(X)$ are *clustered* by k-means, each example is assigned to the nearest centroid, clusters and assignments are treated as classes C and *pseudo-labels* $\hat{\mathbf{y}}$ respectively, and learning takes place according to $J(X, \mathcal{I}, \hat{\mathbf{y}}, \theta)$ (3.2). By updating the parameters θ, $\phi_\theta(X)$ is updated too. The method therefore alternates between clustering/pseudo-labeling and feature learning, typically once per epoch.

3.3 Training the model on unlabeled data

We argue that acquiring examples for labeling is not making the best use of unlabeled data: unlabeled data should be used during model training, appearing in the cost function

that is being minimized. We choose two ways of doing so: unsupervised feature learning and semi-supervised learning, as shown in Figure 3.1. As outlined in Algorithm 1, we follow the standard active learning setup, adding unsupervised pre-training at the beginning and replacing supervised learning on L by semi-supervised learning on $L \cup U$ at each cycle. The individual components are discussed in more detail below.

Algorithme 1 : Semi-supervised active learning

Data : data X, indices of labeled examples L, labels $\mathbf{y}$, batch size b

$U \leftarrow \mathcal{I} \setminus L$ ▷ indices of unlabeled examples
$\theta_0 \leftarrow \text{PRE}(X)$ ▷ unsupervised pre-training
for $j \in \{0, \ldots\}$ **do** ▷ active learning cycles
  $\theta \leftarrow \text{SUP}(X, L, \mathbf{y}; \theta_0)$ ▷ supervised learning on L only
  for $e \in \{1, \ldots\}$ **do** ▷ epochs
    $(\hat{\mathbf{y}}, \mathbf{w}) \leftarrow \text{LP}(X, L, \mathbf{y}, \theta)$ ▷ pseudo-labels $\hat{\mathbf{y}}$ and labels $\mathbf{w}$
    $\theta \leftarrow \text{SEMI}(X, L \cup U, (\mathbf{y}, \hat{\mathbf{y}}), \mathbf{w}; \theta)$ ▷ semi-supervised learning on all data
  $S \leftarrow \emptyset$
  while $|S| < b$ **do** ▷ acquire a batch $S \subset U$ for labeling
    $S \leftarrow S \cup a(X, L \cup S, U \setminus S, \mathbf{y}; \theta)$
  $\mathbf{y} \leftarrow (\mathbf{y}, \text{LABEL}(S))$ ▷ obtain true labels on S by oracle
  $L \leftarrow L \cup S; U \leftarrow U \setminus S$ ▷ update indices

Unsupervised pre-training (PRE) takes place at the beginning of the algorithm. We follow Caron *et al*. [Car+18b], randomly initializing θ and then alternating between clustering the features $\phi_\theta(X)$ by k-means and learning on cluster assignment pseudo-labels $\hat{\mathbf{y}}$ of X according to $J(X, \mathcal{I}, \hat{\mathbf{y}}, \theta)$ (3.2). The result is a set of parameters θ_0 used to initialize the classifier at every cycle.

Learning per cycle follows inductive label propagation [Isc+19a]. This consists of supervised learning followed by alternating label propagation and semi-supervised learning on all examples $L \cup U$ at every epoch. The *supervised learning* (SUP) is performed on the labeled examples L only using labels $\mathbf{y}$, according to $J(X, L, \mathbf{y}, \theta)$ (3.2), where the parameters θ are initialized by θ_0.

Label propagation (LP) involves a reciprocal k-nearest neighbor graph on features $\phi_\theta(X)$ according to (2.1), (2.2). The resulting adjacency matrix W is normalized as $\mathcal{W} := D^{-1/2} W D^{-1/2}$ (2.3). Label propagation is then performed according to $P = \eta[h(Y)]$ (3.9), by solving the corresponding linear system using the *conjugate gradient* (CG) method [Isc+19a]. The label matrix $Y := \chi(L, \mathbf{y})$ (3.8) is defined on the true labeled examples L that remain fixed over epochs but grow over cycles. With $\mathbf{p}_i$ being the i-th row of P, a pseudo-label $\hat{y}_i = \pi(\mathbf{p}_i)$ (3.1) and a weight $w_i = \beta(\mathbf{p}_i)$ (3.10) are defined for every $i \in U$ [Isc+19a].

Semi-supervised learning (SEMI) takes place on all examples $L \cup U = \mathcal{I}$, where examples in L have true labels $\mathbf{y}$ and examples in U pseudo-labels $\hat{\mathbf{y}} := (\hat{y}_i)_{i \in U}$. Different than Iscen *et al*. [Isc+19a], we minimize the standard cost function $J(X, L \cup U, (\mathbf{y}, \hat{\mathbf{y}}), \theta)$ (3.2), but we do take weights $\mathbf{w} := (w_i)_{i \in U}$ into account in mini-batch sampling, ℓ_1-normalized as $\eta[\mathbf{w}]$. In particular, part of each mini-batch is drawn uniformly at random from L, while the other part is drawn with replacement from the discrete distribution $\eta[\mathbf{w}]$ on U: an example may be drawn more than once per epoch or never.

Discussion. The above *probabilistic weighting* decouples the size of the epoch from n and indeed we experiment with epochs smaller than n, accelerating learning compared to Iscen *et al*. [Isc+19a]. It is similar to *importance sampling*, which is typically based on loss values [KF17; Che+18] or predicted class probabilities [Yan+15a]. Acceleration is important as training on all examples is more expensive than just the labeled ones, and is repeated at every cycle. On the contrary, unsupervised pre-training only occurs once at the beginning.

The particular choice of components is not important: any unsupervised representation learning could replace Caron *et al*. [Car+18b] in line 2 and any semi-supervised learning could replace Iscen *et al*. [Isc+19a] in lines 4-7 of Algorithm 1. We keep the pipeline as simple as possible, facilitating comparisons with more effective choices in the future.

3.4 Investigating manifold similarity in the acquisition function

Label propagation [Zho+03a; Isc+19a] is based on the manifold structure of the feature space, as captured by the normalized affinity matrix $\mathcal{W}$. Rather than just using this information for propagating labels to unlabeled examples, can we use it in the acquisition function as well? This is important in interpreting the effect of semi-supervised learning in Algorithm 1: is any gain due to the use of manifold similarity, or to training the model on more data?

Joint label propagation (jLP), introduced here, is an attempt to answer these questions. It is an acquisition function similar in nature to the geometric approach (3.4), with Euclidean distance replaced by manifold similarity. In particular, the n-vector $Y\mathbf{1}_c$, the row-wise sum of $Y = \chi(L, \mathbf{y})$ (3.8), can be expressed as $Y\mathbf{1}_c = \delta(L) \in \mathbb{R}^n$, where

$$\delta(L)_i := \begin{cases} 1, & i \in L, \\ 0, & \text{otherwise} \end{cases} \tag{3.11}$$

for $i \in \mathcal{I}$. As discussed in Section 2.4, in the terminology of *manifold ranking*, vector $Y\mathbf{1}_c$ represents a set of *queries*, one for each example $\mathbf{x}_i$ for $i \in L$, and the i-th element of the n-vector $u^* = h(Y)\mathbf{1}_c$ expresses the *manifold similarity* of $\mathbf{x}_i$ to the queries for $i \in \mathcal{I}$ as in (2.9). Similar to (3.4), we acquire the example in U that is the *least similar* to examples in L that are labeled or previously acquired:

$$a(X, L, U, \mathbf{y}; \theta) := \arg\min_{i \in U} (h(\delta(L)))_i. \tag{3.12}$$

This strategy is only geometric and bears similarities to *discriminative active learning* [GS18], which learns a binary classifier to discriminate labeled from unlabeled examples and acquires examples of least confidence in the "labeled" class.

3.5 Experiments

3.5.1 Experimental setup

Datasets. We conduct experiments on four datasets that are most often used in deep active learning: MNIST [LeC+98], SVHN [Net+11], CIFAR-10 and CIFAR-100 [Kri09]. Table 3.1 presents statistics of the datasets. Following [TV17; Isc+19a], we augment input images by 4×4 random translations and random horizontal flips.

	Data size train / test	Image size	Batch size w/o SEMI / SEMI	Budget	Total labels
MNIST	60000 / 10000	28×28	10 / 64	10	50
SVHN	73257 / 26032	32×32	32 / 128	100	500
CIFAR-10	50000 / 10000	32×32	32 / 128	100	500
CIFAR-10	50000 / 10000	32×32	32 / 128	1000	5000
CIFAR-100	50000 / 10000	32×32	32 / 128	1000	5000

TABLE 3.1 – Datasets used in this chapter, including the batch sizes used in training with and without SEMI, acquisition sizes at each active learning step and the total number of labeled images.

Networks and training. For all experiments we use a 13-layer convolutional network used previously in Laine *et al*. [LA16b]. We train the model from scratch at each active learning cycle, using SGD with momentum of 0.9 for 200 epochs. An initial learning rate of 0.2 is decayed by cosine annealing [LH17], scheduled to reach zero at 210 epochs. The mini-batch size is 32 for standard training and 128 when SEMI is used, except for MNIST where the size of the mini-batch 10 and 64 with SEMI. All other parameters follow Tarvainen *et al*. [TV17].

Unsupervised pre-training. We use k-means as the clustering algorithm and follow the settings of Caron *et al*. [Car+18b]. The model is trained for 250 epochs on the respective datasets.

Semi-supervised learning. Following Iscen *et al*. [Isc+19a], we construct a reciprocal k-nearest neighbor graph on features $\phi_\theta(X)$, with $k = 50$ neighbors and similarity function $s(\mathbf{u}, \mathbf{v}) := [\hat{\mathbf{u}}^\top \hat{\mathbf{v}}]_+^3$ for $\mathbf{u}, \mathbf{v} \in \mathbb{R}^d$, where $\hat{\mathbf{u}}$ is the ℓ_2-normalized counterpart of $\mathbf{u}$, while $\alpha = 0.99$ in (3.9). We follow [Isc+19a] in splitting mini-batches into two parts: 50 examples (10 for MNIST) are labeled and the remaining pseudo-labeled. For the latter, we draw examples using normalized weights as a discrete distribution. The epoch ends when $\frac{1}{2}|U|$ pseudo-labels have been drawn, that is the epoch is 50% compared to Iscen *et al*. [Isc+19a]. Given that $|L| \ll |U|$ in most cases, the labeled examples are typically repeated more than once.

Acquisition strategies. We evaluate our new acquisition strategy jLP along with the following baselines: (a) *Random*; (b) *Uncertainty* based on entropy (3.5); (c) *CEAL* [Wan+17], combining entropy with pseudo-labels (3.7); (d) the greedy version of *CoreSet* (3.4) [SS18; GE17].

Baselines. For all acquisition strategies, we show results of the complete Algorithm 1 as well as the the *standard baseline*, that is without pre-training and only fully supervised on labeled examples L, and *unsupervised pre-training* (PRE) alone without semi-supervised. In some cases we show *semi-supervised* (SEMI) alone. For instance, in the scenario of 100 labels per class, the effect of pre-training is small, especially in the presence of semi-supervised. CEAL [Wan+17] is a baseline with its own pseudo-labels, so we do not combine it with semi-supervised. The length of the epoch is fixed for Algorithm 1 and increases with each cycle for the baselines.

Label budget and cycles. We consider three different scenarios, as shown in Table 3.1. In the first we use an initial balanced label set L_0 of 10 labels per class, translating into a total of 100 for CIFAR-10 and SVHN and 1000 for CIFAR-100. We use the same values as label budget b for all cycles. In the second, we use initially 100 labels per class in CIFAR-10 with

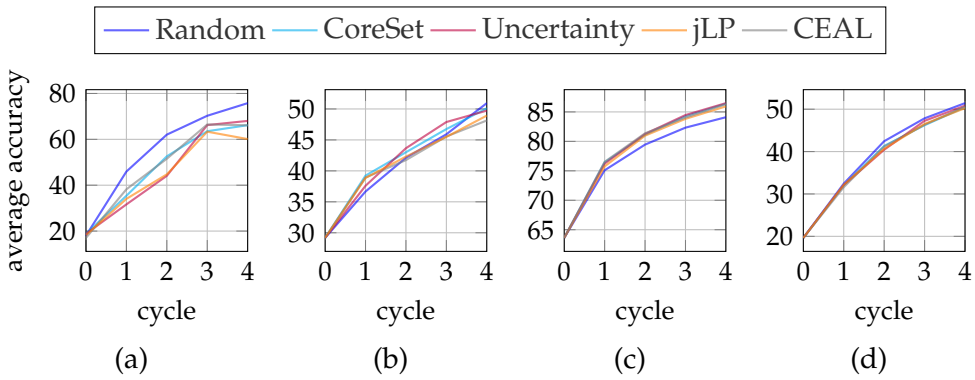


FIGURE 3.2 – Average accuracy *vs.* cycle on different setups and acquisition strategies on SVHN with a budget ($b = 100$)(a), CIFAR-10 with a budget of 100 and 1000 ((b) and (c) respectively), and CIFAR-100 with $b = 1000$ (d).

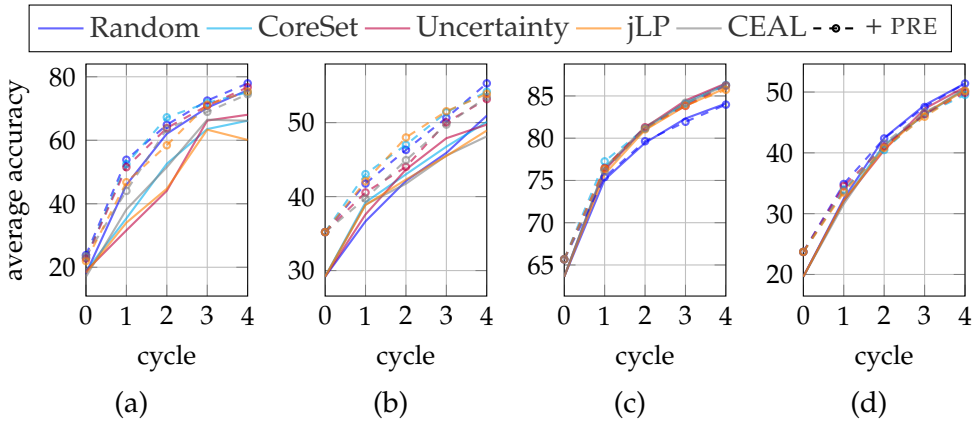


FIGURE 3.3 – Average accuracy *vs.* cycle on different setups and acquisition strategies with PRE on SVHN with a budget ($b = 100$)(a), CIFAR-10 with a budget of 100 and 1000 ((b) and (c) respectively), and CIFAR-100 with $b = 1000$ (d).

$b = 1000$ per cycle; this is not interesting for CIFAR-100 as it results in complete labeling of the training set after 4 cycles. Finally, we investigate the use of one label per class both as the initial set and the label budget, on MNIST, translating to 10 labels per cycle. All experiments are carried out for 5 cycles and repeated 5 times using different initial label sets L_0. We report *average accuracy* and *standard deviation*.

3.5.2 Investigating the agreement of acquisition strategies

We first evaluate acquisition functions without using any unlabeled data. Figure 3.2 presents results on SVHN, CIFAR-10 and CIFAR-100. The differences between acquisition functions are not significant, except when compared to Random. On SVHN, Random appears to be considerably better than the other acquisition functions and worse on CIFAR-10 with $b = 1000$. All the other acquisition functions give near identical results; in particular, there is no clear winner in the case of 10 labels per class on CIFAR-10 and CIFAR-100 (Figure 3.2(b) and (d), respectively).

This confirms similar observations made in Gissin *et al.* [GS18] and Chitta *et al.* [CAL19]. Our jLP is no exception, giving similar results to the other acquisition functions. We study

METHOD	CIFAR-10	CIFAR-100
BUDGET	$b = 100$	$b = 1000$
CYCLE 0		
jLP	29.17±1.62	19.63±0.99
+ PRE	35.20±2.26	23.71±0.86
+ SEMI	36.73±5.70	25.06±1.44
+ PRE + SEMI	**38.05±2.92**	**27.04±0.78**
CYCLE 1		
jLP	38.86±1.36	32.16±1.98
+ PRE	42.07±0.74	33.48±0.52
+ SEMI	46.76±3.27	37.99±2.47
+ PRE + SEMI	**48.66±2.64**	**40.30±1.53**
CYCLE 2		
jLP	42.30±1.61	40.65±1.21
+ PRE	47.99±1.17	40.81±0.40
+ SEMI	**51.53±3.02**	46.39±1.49
+ PRE + SEMI	51.18±1.80	**47.03±0.47**

TABLE 3.2 – Ablation study. Evaluation of results obtained with Random while adding PRE and/or SEMI.

this phenomenon in Section 3.6. In summary, we find that while the ranks of examples according to different strategies may be uncorrelated, the resulting predictions of label propagation mostly agree. Even in cases of disagreement, the corresponding examples have small weights, hence their contribution to the cost function is small. Since those predictions are used as pseudo-labels in Iscen *et al*. [Isc+19a], this can explain why the performance of the learned model is also similar in the presence of semi-supervised learning.

3.5.3 The effect of unsupervised pre-training

As shown in Figure 3.3, pre-training can be beneficial. PRE by itself brings substantial gain on SVHN and CIFAR-10 with $b = 100$, up to 6%, while the improvements on CIFAR-100 are moderate. In addition, numerical results in Table 3.2 for our acquisition strategy jLP show that PRE is beneficial with or without SEMI in most cases. Pre-training provides a relatively easy and cost-effective improvement. It is performed only once at the beginning of the active learning process. While Caron *et al*. [Car+18b] was originally tested on large datasets like ImageNet or YFCC100M, we show that it can be beneficial even on smaller datasets like CIFAR-10 or SVHN.

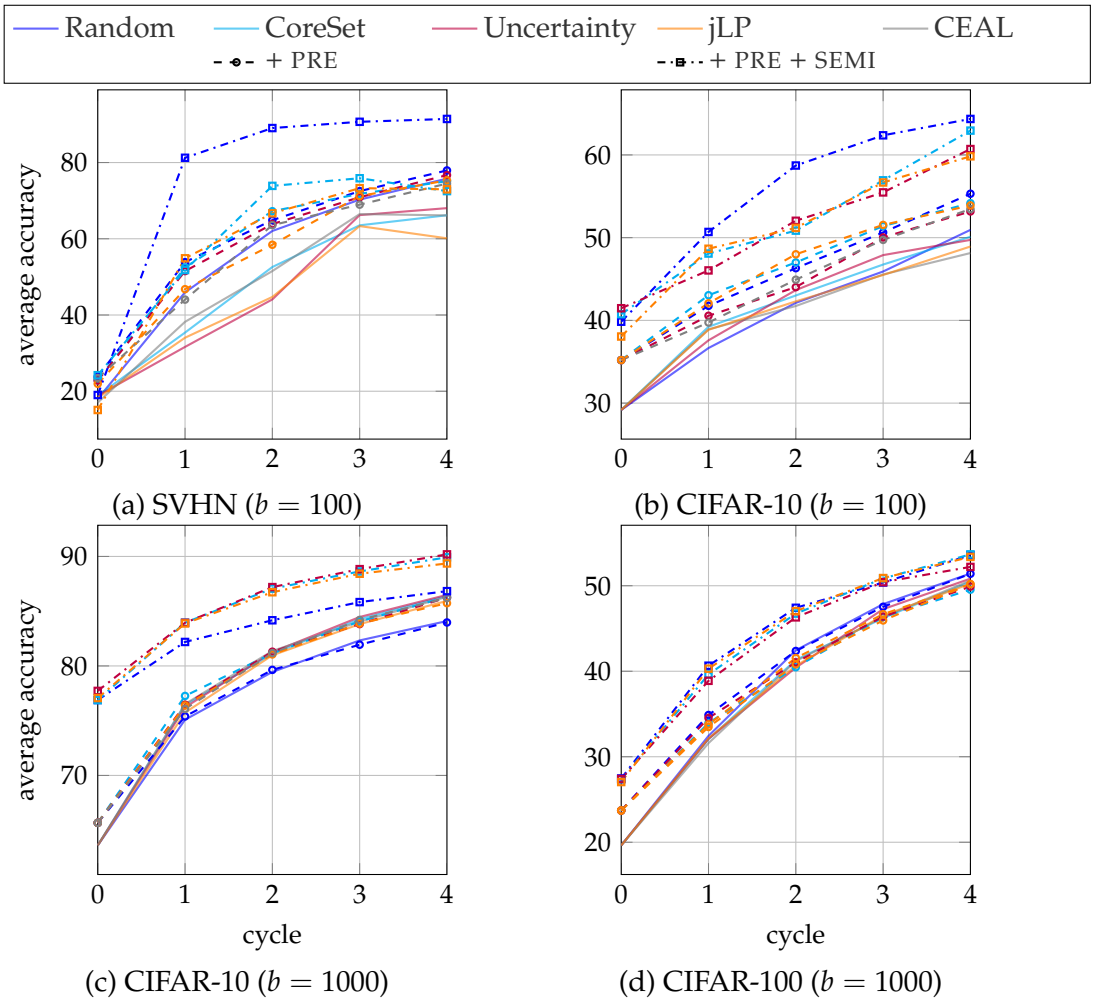
3.5.4 The effect of semi-supervised learning

Figure 3.4 shows results on different datasets and acquisition strategies like Figure 3.3, but including both PRE and PRE + SEMI. For the purpose of reproducibility, numeric results, including average and standard deviation measurements, are given in Table 3.3. We describe results obtained with the five methods Random, Uncertainty, CEAL, CoreSet and jLP presented before. We evaluate them on CIFAR-10 with 10 and 100 labels per class (budget $b = 100$ and $b = 1000$ respectively), CIFAR-100 with $b = 1000$ and SVHN with $b = 100$ in Table 3.3.

The combination PRE + SEMI yields a further significant improvement over PRE and the standard baseline, on all acquisition functions and datasets. For instance, on CIFAR-10 with

a budget of 100, the most noticeable improvement comes from Random, where the improvement of PRE + SEMI is around 15% over the standard baseline at all cycles. The improvement is around 10% in most other cases, which is by far greater than any potential difference between the acquisition methods. Also, noticeably, in the case of SVHN, Random with PRE + SEMI reaches nearly the fully supervised accuracy after just 2 cycles (300 labeled examples in total).

The gain from semi-supervised learning is dramatic in the few-labels regime of CIFAR-10 with $b = 100$. A single cycle with PRE + SEMI achieves the accuracy of 4 cycles of the standard baseline in this case, which translates to a significant reduction of cost for human annotation.



(a) SVHN ($b = 100$) (b) CIFAR-10 ($b = 100$)

(c) CIFAR-10 ($b = 1000$) (d) CIFAR-100 ($b = 1000$)

FIGURE 3.4 – Average accuracy *vs.* cycle on different setups and acquisition strategies: Baseline, PRE and PRE +SEMI. PRE, SEMI and PRE + SEMI scenarios are represented using different dashed lines as presented in the legend. For reference, the full training accuracy is 96.97% for SVHN, 94.84 % for CIFAR-10 and 76.43 % for CIFAR-100.

In Table 3.2 we present the effect of all four combinations: with/without PRE and with/without SEMI. We focus on our jLP acquisition strategy, which has similar performance as all other strategies and uses manifold similarity just like SEMI. In most cases, PRE improves over SEMI alone by around 2%. The use of PRE appears to be particularly beneficial in the first cycles, while its impact decreases as the model performance improves.

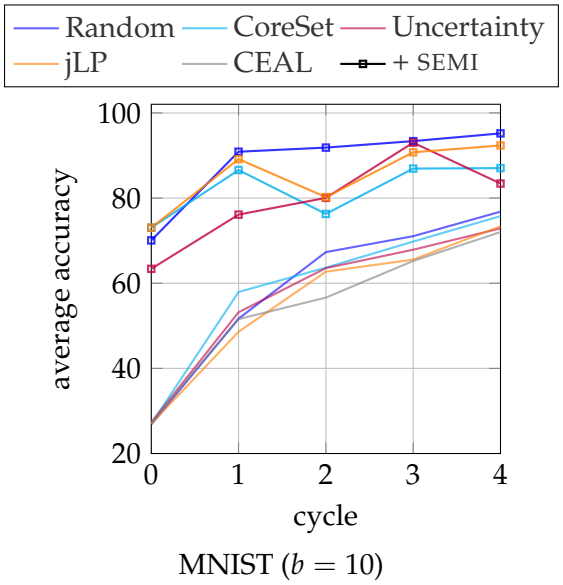


FIGURE 3.5 – Average accuracy *vs.* cycle on different setups and acquisition strategies.

It is worth noting that CEAL, which makes use of pseudo-labels, has a low performance. This has been observed before Ducoffe *et al.* [DP18] and can be attributed to the fact that it is using the same set of pseudo-labels in every epoch. By contrast, pseudo-labels are updated in every epoch in our case.

3.5.5 Label propagation with one label per class

Here we demonstrate the effectiveness of SEMI in combination with acquisition functions against the functions alone. To that end we use MNIST as a benchmark. We investigate a relatively difficult scenario in which the AL budget at each cycle is equal to the number of classes. Figure 3.5 shows the 5 strategies with and without SEMI. As noted before, there is no consistent winner among the selection strategies alone, and results with 50 labels (5 cycles) remain below 80%. However, Random with SEMI arrives at 90.89% accuracy with only 20 labeled examples which is 40% better than without. We also investigated the scenario of 2 labels per class, but did not extend the study as we achieve 94.39% with 40 labels and 97.86% with 60 (2 cycles and 3 cycles, respectively) combining Random and SEMI. This results are close to the 99.44% accuracy achieved with a full training over the 50000 labeled images. Detailed results are presented in Table 3.3.

3.6 Studying the agreement of acquisition strategies

It has been observed that most acquisition strategies do not provide a significant improvement over standard uncertainty when using deep neural networks; for instance, all strategies perform similarly on CIFAR-10 and CIFAR-100 according to Gissin *et al.* [GS18] and Chitta *et al.* [CAL19]. To better understand the differences, the ranks of examples acquired by different strategies are compared pairwise by Gissin *et al.* [GS18]. We make a step further in this direction, using label propagation as a tool.

In particular, after the classifier is trained at any cycle using any reference acquisition function a, we apply two different acquisition functions, say $a^{(1)}$ and $a^{(2)}$, followed by labeling of acquired examples and label propagation, obtaining two different sets of predicted

METHOD	MNIST, $b = 10$		SVHN, $b = 100$			CIFAR-10, $b = 100$			CIFAR-10, $b = 1000$		CIFAR-100, $b = 1000$		
PRE				✓	✓		✓	✓				✓	✓
SEMI		✓			✓			✓		✓			✓
CYCLE 0	10 LABELS			100 LABELS			100 LABELS		1K LABELS			1K LABELS	
Random	26.83±4.15	**70.06±12.87**	**18.00±2.47**	**23.83±4.63**	**19.01±5.61**	29.17±1.62	35.20±2.26	39.84±2.63	63.61±1.42	78.85±0.86	19.63±0.99	23.71±0.86	27.46±0.52
CYCLE 1	20 LABELS			200 LABELS			200 LABELS		2K LABELS			2K LABELS	
Random	51.68±2.72	**90.89±4.84**	**45.95±1.97**	**53.87±5.43**	**81.25±4.82**	36.66±1.08	41.76±1.32	**50.69±2.95**	75.09±0.51	83.49±0.81	**32.44±1.69**	**34.88±0.90**	**40.65±0.63**
Uncertainty	53.18±5.88	76.12±11.07	31.63±8.75	51.52±2.36	37.84±21.00	37.59±1.93	40.56±2.21	46.04±2.78	76.22±0.68	84.94±0.35	32.09±1.50	34.54±0.70	38.88±1.11
CoreSet	**57.94±7.16**	86.59±10.98	35.39±7.16	52.49±5.76	51.80±10.62	**39.23±1.17**	**43.04±0.92**	48.08±1.64	76.44±0.34	**84.98±0.19**	32.05±1.40	33.95±0.57	39.63±0.70
CEAL	51.57±3.18	–	38.21±2.70	44.04±4.56	–	38.92±2.00	39.74±1.72	–	**76.52±0.73**	–	31.59±0.93	33.78±0.39	–
jLP (ours)	48.60±3.15	89.16±5.53	34.04±4.75	46.78±5.18	54.88±22.90	38.86±1.36	42.07±0.74	48.66±2.64	75.74±0.39	84.62±0.47	32.16±1.98	33.48±0.52	40.30±1.53
CYCLE 2	30 LABELS			300 LABELS			300 LABELS		3K LABELS			3K LABELS	
Random	**67.31±5.19**	**91.86±3.89**	**62.05±3.23**	64.88±4.93	**89.05±2.07**	42.12±1.83	46.31±1.40	**58.72±4.04**	79.45±0.56	85.33±0.42	**42.45±0.90**	**42.37±0.53**	**47.42±0.53**
Uncertainty	63.55±2.67	80.05±13.29	44.09±13.49	63.85±3.55	64.14±6.36	**43.66±1.57**	44.02±1.73	52.04±2.46	**81.26±0.30**	**87.65±0.29**	40.43±0.63	41.04±0.27	46.30±1.12
CoreSet	63.66±3.84	76.28±15.38	52.59±9.20	**67.23±3.01**	73.88±13.94	43.01±2.14	47.00±2.57	50.85±4.23	81.11±0.61	87.21±0.31	41.32±0.70	40.47±0.38	46.74±1.00
CEAL	56.62±7.05	–	51.53±5.93	63.58±2.80	–	41.74±1.15	44.92±2.09	–	81.37±0.54	–	41.19±0.41	41.55±0.45	–
jLP (ours)	62.71±2.82	80.23±4.11	44.74±17.50	58.43±9.82	66.68±13.91	42.30±1.61	**47.99±1.17**	51.18±1.80	80.97±0.40	87.16±0.44	40.65±1.21	40.81±0.40	47.03±0.47
CYCLE 3	40 LABELS			400 LABELS			400 LABELS		4K LABELS			4K LABELS	
Random	**71.05±1.66**	**93.38±3.99**	**70.28±1.67**	**72.50±2.05**	**90.69±0.73**	45.91±1.63	50.63±0.59	**62.37±1.41**	82.33±0.21	86.66±0.21	**47.85±0.84**	**47.54±0.63**	50.38±0.25
Uncertainty	67.87±3.26	93.03±4.88	66.21±3.68	70.90±2.48	56.60±5.69	**47.89±1.78**	50.03±1.38	55.47±2.10	**84.47±0.49**	**89.32±0.24**	47.26±0.79	46.39±0.81	50.42±0.24
CoreSet	69.79±3.36	86.93±7.62	63.53±6.34	71.79±3.58	75.88±6.95	46.75±2.41	51.40±1.99	56.93±2.90	84.27±0.36	88.75±0.45	46.22±0.39	46.34±0.92	50.85±0.32
CEAL	65.24±7.43	–	66.48±2.80	68.95±2.06	–	45.55±2.39	49.73±1.82	–	84.05±0.44	–	46.34±0.44	46.67±0.38	–
jLP (ours)	65.55±4.01	90.75±5.76	63.33±9.59	71.20±2.93	73.28±11.69	45.49±1.71	**51.54±1.24**	56.67±2.58	83.82±0.02	88.85±0.38	46.52±0.99	45.94±0.44	**50.90±0.67**
CYCLE 4	50 LABELS			500 LABELS			500 LABELS		5K LABELS			5K LABELS	
Random	**76.81±2.19**	**95.20±3.61**	**75.78±1.90**	**77.93±1.55**	**91.44±0.80**	**50.94±1.75**	**55.31±1.28**	**64.35±1.37**	84.10±0.10	87.23±0.21	**51.43±0.56**	**51.40±0.47**	53.58±0.64
Uncertainty	72.88±5.82	83.42±5.93	68.04±6.58	76.70±1.11	55.42±10.49	49.73±2.29	53.17±1.52	60.71±2.77	**86.49±0.19**	**90.42±0.28**	50.83±0.31	49.90±0.82	52.20±0.50
CoreSet	75.76±3.93	87.04±6.44	66.17±16.11	75.11±3.40	72.51±9.99	50.11±1.40	54.17±0.40	62.94±2.41	86.39±0.36	90.33±0.13	50.48±0.84	49.54±0.95	**53.67±1.29**
CEAL	72.02±7.96	–	66.14±14.42	74.48±1.98	–	48.14±1.24	53.46±1.27	–	86.31±0.23	–	50.62±0.28	50.18±0.60	–
jLP (ours)	73.36±4.43	92.37±5.38	60.12±20.06	75.33±1.44	72.98±12.01	48.93±2.22	53.89±1.42	59.83±4.02	85.94±0.38	89.91±0.28	50.24±0.93	50.20±0.44	53.37±0.64

TABLE 3.3 – Average accuracy and standard deviation for different label budget b and cycle on MNIST, SVHN, CIFAR-10 and CIFAR-100. Following Algorithm 1, we show the effect of unsupervised pre-training (PRE) and semi-supervised learning (SEMI) compared to the standard baseline.

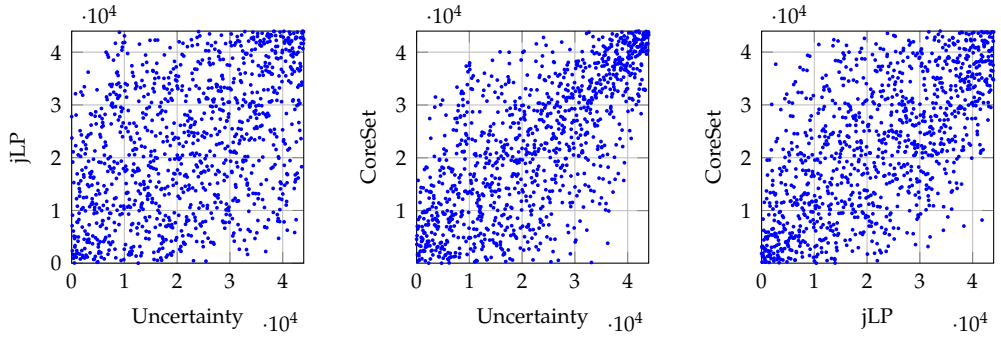


FIGURE 3.6 – Ranks of examples obtained by one acquisition strategy *vs*. the ranks of another on CIFAR-10 with $b = 1000$ after cycle 1. A random 5% subset of all examples is shown.

pseudo-labels $\hat{\mathbf{y}}^{(1)}$ and $\hat{\mathbf{y}}^{(2)}$ and weights $\mathbf{w}^{(1)}$ and $\mathbf{w}^{(2)}$ on the unlabeled examples U. We define the *weighted accuracy*

$$A_{U,\mathbf{w}}(\mathbf{z}, \mathbf{z}') = \sum_{i \in U} \eta [\mathbf{w}]_i \delta_{z_i, z'_i} \tag{3.13}$$

for $\mathbf{z}, \mathbf{z}' \in \mathbb{R}^{|U|}$, where δ is the Kronecker delta function. Using the average weights $\mathbf{w} := \frac{1}{2}(\mathbf{w}^{(1)} + \mathbf{w}^{(2)})$, we then measure the weighted accuracy $A_{U,\mathbf{w}}(\mathbf{y}^{(1)}, \mathbf{y}^{(2)})$, expressing the *agreement* of the two strategies, as well as the weighted accuracy $A_{U,\mathbf{w}}(\mathbf{y}^{(k)}, \mathbf{t})$ of $a^{(k)}$ relative to the true labels $\mathbf{t}$ on U for $k = 1, 2$. More measurements include weighted accuracies relative to true labels on subsets of U where the two strategies agree or disagree. This way, assuming knowledge of the true labels on the entire set X, we evaluate the quality of pseudo-labels used in semi-supervised learning in each cycle, casting label propagation as an efficient surrogate of the learning process.

Results. Here we take a closer look at the similar performance of most acquisition strategies. We show results on CIFAR-10 with $b = 1000$ in this study. Following the experiments of [GS18], we first investigate the correlation of the ranks of unlabeled examples obtained by two acquisition functions. As shown in Figure 3.6(a), Uncertainty and jLP are not as heavily correlated compared to, for example, *CoreSet* and *Uncertainty* in Figure 3.6(b). The correlation between *jLP* and *CoreSet* is also quite low as shown in Figure 3.6(c).

It may of course be possible that two strategies with uncorrelated ranks still yield models of similar accuracy. To investigate this, we follow the methodology described in Section 3.6. Results are shown in Table 3.4. Uncertainty is used as a reference strategy, *i.e*. we train the model for a number of cycles using Uncertainty and then measure agreement and disagreement of another strategy to Uncertainty. After cycle 1, any two methods agree on around 80% of the pseudo-labels, while the remaining 20% have on average smaller weights compared to when the methods agree.

We reach the same conclusions from a similar experiment where we actually train the model rather than performing label propagation. Hence, although examples are ranked differently by different strategies, their effect on prediction, either by training or label propagation, is small.

CYCLE	1					2				
MEASURE	%agree	accuracy (3.13)		avg weights		%agree	accuracy (3.13)		avg weights	
AGREE?		$=$	$\neq$	$=$	$\neq$		$=$	$\neq$	$=$	$\neq$
Random	79.98	79.97	38.39	0.32	0.17	86.98	88.07	39.77	0.46	0.28
CoreSet	80.58	79.52	44.57	0.27	0.16	87.32	87.94	43.80	0.45	0.29
jLP (ours)	80.24	80.03	48.79	0.27	0.15	86.96	88.12	45.55	0.43	0.27

TABLE 3.4 – Agreement results between acquisition strategies on CIFAR-10 with $b = 1000$ after cycles 1 and 2. All strategies are compared to Uncertainty as reference, which is also employed in the previous cycles. *%agree* is percentage of pseudo-labels agreeing to the reference. Accuracy is weighted according to (3.13) and weights are according to (3.10). Measurements denoted by $=$ ($\neq$) refer to the set of pseudo-labels that agree (disagree) with the reference.

3.7 Discussion

In this work, we have shown the benefit of using both labeled and unlabeled data during model training in deep active learning for image classification. This leads to a more accurate model while requiring less labeled data, which is in itself one of the main objectives of active learning. We have used two particular choices for unsupervised feature learning and semi-supervised learning as components in our pipeline. There are several state of the art methods that could be used for the same purpose, for instance Tarvainen *et al*. [TV17], Verma *et al*. [Ver+19], Berthelot *et al*. [Ber+19], and Rebuffi *et al*. [Reb+19] for semi-supervised learning. Our pipeline is as simple as possible, facilitating comparisons with more effective choices, which can only strengthen our results. While the improvement coming from recent acquisition strategies is marginal in many scenarios, an active learning approach that uses unlabeled data for training and not just acquisition appears to be a very good option for deep network models. Our findings can have an impact on how deep active learning is evaluated in the future. For instance, the relative performance of acquisition strategies changes in the presence of pre-training and semi-supervised learning.

Chapter 4

Deep Spatial Matching

Contents

In this chapter, we introduce Deep Spatial Matching (DSM) [SAC19], a novel way of extracting localization information from feature maps of convolutional networks and using this information to perform spatial matching for large-scale image retrieval. Initial ranking is based on image descriptors extracted from convolutional neural network activations by global pooling, as in recent state-of-the-art work. However, the same sparse 3D activation tensor is also approximated by a collection of local features. These local features are then robustly matched to approximate the optimal alignment of the tensors. This happens without any network modification, additional layers or training. No local feature detection happens on the original image. No local feature descriptors and no visual vocabulary are needed throughout the whole process.

We experimentally show that the proposed method achieves the state-of-the-art performance on standard benchmarks across different network architectures and different global pooling methods. The highest gain in performance is achieved when diffusion on the nearest-neighbor graph of global descriptors is initiated from spatially verified images.

We introduce our work in Section 4.1. After discussing background in Section 4.2, we develop DSM, in Section 4.3. Experimental results are reported in Section 4.4. In Section 4.5 we discuss how a network training can take advantage of the feature detection and spatial matching. Conclusions are drawn in Section 4.6.

4.1 Introduction

Image and specific object retrieval is commonly addressed as large scale image matching: a query is matched against the database images and the final ranking is given by the matching score. In the early retrieval days, methods based on local features were dominating [SZ03; NS06]. The matching score was first approximated by a similarity of bag of words [SZ03] or aggregated descriptors [Jég+10], and then re-ranked by efficient spatial verification [Phi+07; PCM09].

Recently, image retrieval is dominated by CNNs [Gor+17; RTC18]. Image representation is derived from the output of the CNN, which can be interpreted as a collection of 2D response maps of pattern detectors. The position of the response indicates the location of the pattern in the image, the size of the pattern is limited by the receptive field, and the value of the response indicates the confidence in the presence of the pattern.

Images of corresponding objects or object parts have similar response in all channels. It is known that the image-to-image mapping can be recovered by correlating the response tensors of the two images [LZD14; Cho+16; RAS18].

In general, the CNN activation tensor size depends on the number of channels and the image size. It is is too large to be stored, especially for large-scale applications. To construct a descriptor of a fixed and reasonable size, vectors obtained by global pooling are extracted instead, for instance mean pooling [BL15], max pooling [TSJ16], generalized-mean pooling [RTC18], and others [KMO16; TSJ16]. If the CNN-response tensors are matching, the statistics obtained after the global pooling should be matching too.

Global pooling not only reduces the size of the descriptor, but also injects view-point invariance. In fact, the global pooling is, similarly as bag of features, invariant to a very broad class of transformations. Thus, some information, namely geometric consistency, is lost.

In this work we introduce a very simple way of extracting from the CNN activations a representation that is suitable for geometric verification, which we apply to re-ranking. Ideally, one would estimate the geometric transformation to align the activation tensors and compare. Nevertheless, as stated previously, this would be impractical. We propose to approximate this process, exploiting two properties of the activations: high values are more important and the activations are sparse. Therefore each channel can be well approximated by a small number of extremal regions.

4.2 Background

As discussed in Section 2.5, the power of CNN representations of one or very few regional descriptors per image allowed reducing image retrieval to nearest neighbor search and extending previous query expansion [Chu+07; TJ14] into efficient online exploration of the entire nearest neighbor graph of the dataset by *diffusion* [Isc+17]. The main drawback of these compact representations is that they are not compatible with spatial verification. However, using spatial verification ensures accuracy of the top ranking results and boosts greatly retrieval results. We discuss in this section *spatial verification*, also called *geometric verification*.

[1] Answer: (b) [this work].

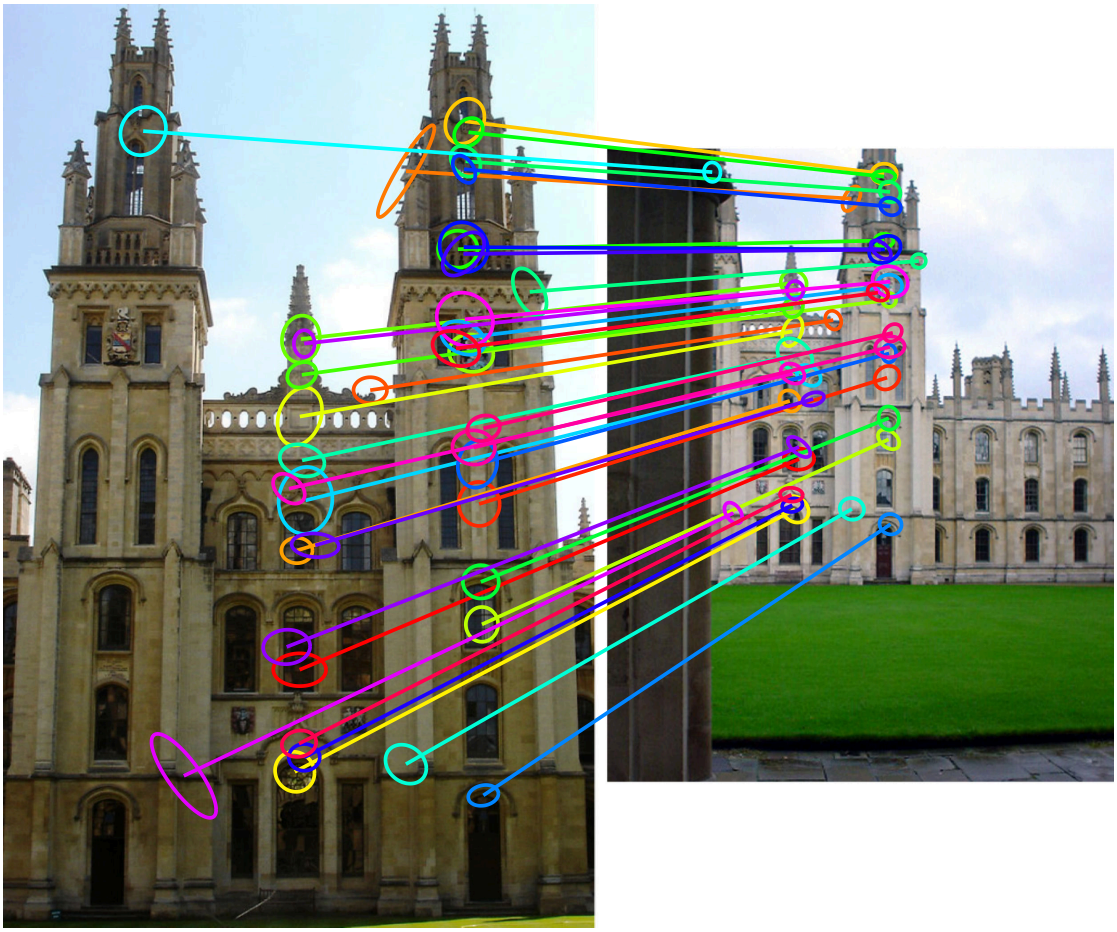


FIGURE 4.1 – Fast spatial matching [Phi+07] finds a linear geometric transformation between two views of an object based on a local feature representation. This is used for spatial verification in large-scale image retrieval. Inlier correspondences shown, colored by visual word. What is the underlying representation?[1]

(a) *SIFT [Low99] descriptors on Hessian-affine [MS04] local features.*

(b) *Descriptors on detected patches by an end-to-end differentiable pipeline using patch pair labels [Yi+16].*

(c) *A subset of convolutional features at locations selected by an attention mechanism learned on image-level labels [Noh+17].*

(d) *Local maxima on each channel of a vanilla feature map. No vocabulary needed.*

Geometric verification The well-known RANdom SAmple Consensus (RANSAC) algorithm, by Fischler *et al*. [FB81], performs robust estimation of a geometric model on experimental data contaminated with outliers. Models are iteratively tested against the full dataset and evaluated using a metric *e.g*. the number of *inliers*. Hartley *et al*. [HZ04] compared more model hypothesis metrics. The process stops when the likelihood of finding a better model is too low. There are several improvements of RANSAC by adding prior knowledge, for instance MLE-SAC [TZ00], or NAPSAC [Mya+02].

Chum *et al*. [CMK03] showed that the quality of an hypothesis was increasing with the number of samples used to generate it and proposed Locally Optimized RANSAC (LO-RANSAC). They added at the end of a RANSAC pass the re-estimation of the model parameters from all inliers. The locally optimized model is then tested against the full data and a new set of inliers is computed. As a result, LO-RANSAC proved to be faster than RANSAC.

In *FSM* [Phi+07], Philbin *et al*. proposed to remove the randomness in RANSAC by enumerating all hypothesis, which would result in better performances. To do so, they proposed on one side to consider a restricted class of transformations at hypothesis generation and on the other side to exploit the affine-invariant properties of regions (features). Doing so, they could generate hypothesis using only one pair of regions. As a result, the number of hypothesis would be significantly reduced, speeding up the matching process.

The Generalized Hough Transform (GHT) [Bal81] can be used to find promising transformation hypotheses by a voting process in the transformation space. Lowe [Low99] used a version of GHT where each correspondence casts a single vote, followed by a verification process similar to RANSAC. Jégou *et al*. [JDS10] incorporated a voting process in the indexing process independently for relative scales and orientations rather than full transformations, which still requires geometric verification. Tolias *et al*. [TA11] used a hierarchical partition of the transformation space, which allows computing a global matching score without verification of hypotheses, making the matching process linear in the number of correspondences. Li *et al*. [LLH15] followed [TA11] and used pairs of correspondences too.

Spatial verification using CNNs Rocco *et al*. [RAS17; RAS18] proposed networks trained to perform geometric matching in an end-to-end fashion. Feature extraction, matching and transformation estimation are performed using three different differentiable layers. Rocco *et al*. [Roc+18] matched two images by learning to discover neighborhood consensus patterns in a 4D space representing all correspondences between pairs of images. This exhaustive comparison doesn't require to estimate a geometric model. Although relevant, those works are too expensive for image retrieval.

Noh *et al*. [Noh+17] and Dusmanu *et al*. [Dus+19] applied RANSAC on *keypoints* descriptors detected and extracted from CNN activation maps. The performance of DELF [Noh+17] was remarkable in benchmark [Rad+18]. DELF combines the power of CNN features with the conventional pipeline of hundreds of local descriptors per image found using an attention map, followed by encoding against a vocabulary, searching by inverted files and spatial matching by RANSAC. Local features are only described by coordinates, hence efficient matching by FSM does not apply. More relevant work is D2-NET [Dus+19], published concurrently with ours. The method includes the discovery of local features within feature maps–one per map–without additional training. As in DELF [Noh+17], a high-dimensional descriptor per feature is stored in order to perform spatial matching. Considering that FSM [Phi+07] can compute a transformation hypothesis using two affine regions, we investigate

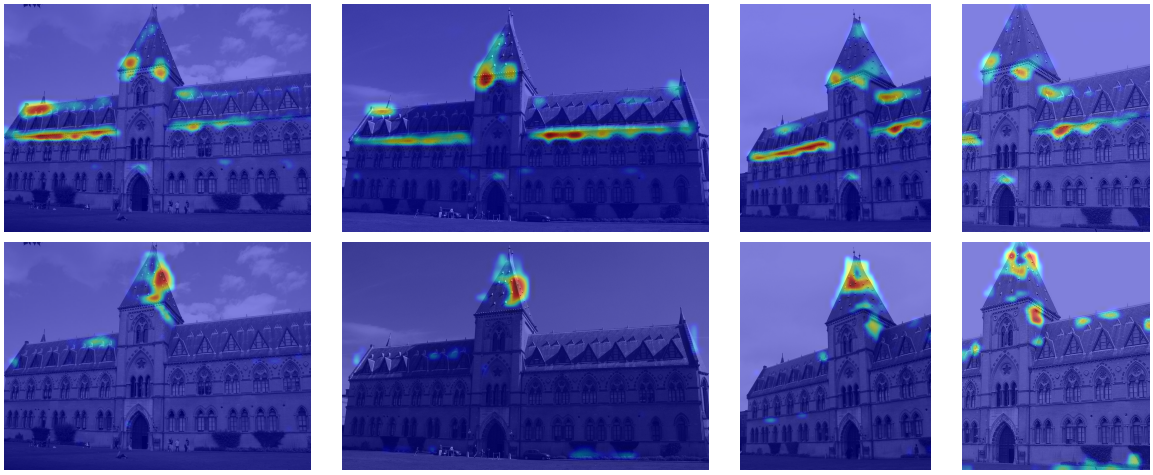


FIGURE 4.2 – Four views (columns) of the Museum of natural history in the $\mathcal{R}$Oxford dataset, overlaid with two different feature maps (rows) of the last convolutional layer of the VGG16 [SZ14] network. The filter kernel in each channel is responding to similar image structures in all images. All activations are naturally sparse and nonzero responses agree in both location and local shape between all images.

in this work how to reduce storage cost by not using expensive descriptors. Similar to us, Cieslewski *et al*. [CBS19] also used no descriptors, however they only used one local feature per channel and did not report results without training.

RANSAC [FB81] is not differentiable. There have been attempts [Bra+17; RAS18] to propose an approximate and differentiable version that could be integrated to CNN architecture. Differentiable SAmple Consensus (DSAC) [Bra+17] uses a probabilistic selection of highest scoring model hypotheses. Rocco *et al*. [RAS18] proposed an alignment network that estimates one geometric transformation which is then verified using a soft-inlier count. Alternatively, classic RANSAC has been improved by using neural guidance in [BR19].

4.3 Deep spatial matching

We begin by motivating our approach, and then present the proposed architecture, followed by the main ideas, including feature detection and representation from CNN activations, spatial matching and re-ranking.

4.3.1 Motivation

Given a convolutional neural network ending in global average pooling, objects of a given class can be localized by CAM [Zho+16], even if the network has only been trained for classification on image-level labels. These maps are linear combinations of individual feature maps (channels) of the last convolutional layer. Grad-CAM [Sel+17] generalizes this idea to any network architecture and allows visualization at any layer by a similar linear combination on the gradient signal instead. Without any class semantics, another linear combination produces a saliency map used for spatial pooling in CroW [KMO15; KMO16]. The latter weighs channels according to *sparsity*, but in all cases the linear combinations only provide coarse localization of objects of a given class or class-agnostic salient regions.

Experiments in [TSJ16] have shown MAC to be superior to other spatial pooling schemes, at least for image retrieval. This can be connected to the sparsity of the activations. More

interestingly, looking at the positions of the maxima in channels contributing most to image similarities, one can readily identify correspondences between two images [TSJ16]. The same has been observed in person re-identification [Alm+18]. Later, GeM [RTC18] was shown to outperform max-pooling. This can be attributed to the fact that it allows for more than one locations contributing to the representation, while still being more selective than average pooling.

Following the above observations, we investigate the responses of the last convolutional layer of VGG on several matching images of the $\mathcal{R}$Oxford dataset. This time we do not limit ourselves to the channels that are contributing most to image similarity (assuming *e.g.* global max-pooling and cosine similarity), but we rather observe all channels. We find out that, as illustrated in Figure 4.2, for two example channels, in most cases the responses to all images are not just sparse but consistent too: the filters respond to the same structures in the images, and there are responses at consistent locations with consistent local shape. The responses exhibit translation and scale covariance to some extent. The deep spatial matching proposed in this work is motivated by the following ideas.

Instead of just reducing each channel to a single scalar, why not keep all the peaks of the responses in each channel along with geometric information (coordinates and local shape)? Instead of attaching an entire descriptor to each such geometric entity, why not just attach the channel it was extracted from, as if it was a visual word?

We propose a method in-between two commonly used approaches, taking the best of the two worlds. One is conventional representations of thousands of local features per image, each with its own descriptor, suitable for inverted files and spatial verification. The other relies on a single global or few regional descriptors per image, leading to compact storage, efficient nearest neighbor search, and graph-based re-ranking. The proposed approach is applicable to any network fine-tuned for retrieval, without requiring any network adaptation, even without any training. It needs no vocabulary and it is trivially related to the global descriptors that dominate the state of the art.

4.3.2 Method overview

The preceding ideas give rise to the Deep Spatial Matching (DSM) network architecture that we introduce in this work, illustrated in Figure 4.3. We consider a fully convolutional backbone network architecture that maintains as much as possible spatial resolution. We denote by f the *network function* that maps an input image to the feature tensor of the last convolutional layer. We assume that the backbone network f, when followed by a pooling mechanism *e.g.* MAC [TSJ16] or GeM [RTC18], extracts a global descriptor that is used *e.g.* for retrieval [Gor+17; RTC18].

As shown in Figure 4.3, two input images x_1, x_2 are processed by a network into 3-dimensional *feature tensors* $A_1 := f(x_1), A_2 := f(x_2)$ where $A_i \in \mathbb{R}^{w_i \times h_i \times k}$, $w_i \times h_i$ is the spatial resolution of A_i for $i = 1, 2$ and k is the number of channels (features). Using the two feature tensors is standard practice in image registration [LZD14; Cho+16], optical flow [Dos+15a] or semantic alignment [Kim+17; RAS18], but here we use an entirely different way of working with the tensors.

In particular, similarly to local feature detection from a single feature tensor [Noh+17], most registration/flow/alignment methods see a feature tensor $A \in \mathbb{R}^{w \times h \times k}$ as a $w \times h$ array of k-dimensional vector descriptors. Then, given two feature tensors, most consider

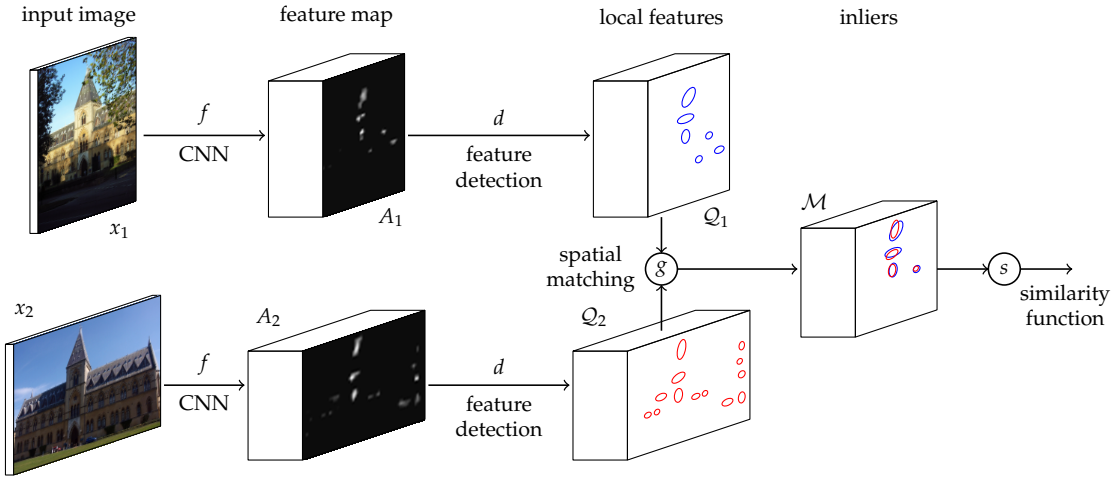


FIGURE 4.3 – Deep Spatial Matching (DSM) network architecture. Two input images x_1, x_2 are mapped by network f to feature tensors A_1, A_2 respectively. Sparse *local features* $\mathcal{Q}_1, \mathcal{Q}_2$ extracted by *detector* d undergo *spatial matching* g, resulting in a collection of inliers $\mathcal{M}$. Similarity function s applies to this collection. Local features are detected and matched independently per channel, with channels playing the role of *visual words*. This takes place without any additional learning and without adapting the backbone network. In retrieval, only local features $\mathcal{Q}_1, \mathcal{Q}_2$ are stored and g applies directly at re-ranking.

the correlation of the two 2-dimensional arrays, seeking dense correspondences. By contrast, from each feature tensor A_1, A_2 we extract a sparse collection of *local features* $\mathcal{Q}_1 := d(A_1), \mathcal{Q}_2 := d(A_2)$ respectively. The feature detector d, discussed in Subsection 4.3.3, operates independently per channel and each local feature collection $\mathcal{Q}$ is a list of sets, one per channel. Local features are represented as discussed in Subsection 4.3.4.

Then, the two local feature collections $\mathcal{Q}_1, \mathcal{Q}_2$ undergo *spatial matching*, denoted as g and discussed in Subsection 4.3.5, returning a collection of inliers $\mathcal{M}$ and a geometric transformation T. We fit a linear motion model to a collection of tentative *correspondences*, *i.e.*, pairs of local features from the two images, which are formed again independently per channel. This implicitly assumes that the "appearance" of each local feature is *quantized* according to channel where it was detected, hence channels play the role of *visual words*, without any descriptor vectors ever being computed. The output collection of *inlier* correspondences $\mathcal{M}$ is again given as a list of sets, one per channel. Finally, *similarity function* s applies to $\mathcal{M}$.

The entire feature detection and matching mechanism takes place without adapting the backbone network in any way and without any additional learning. When applied to image retrieval, this architecture assumes that local features have been precomputed and are the representation of the database images, that is, feature tensors are discarded. Based on this representation, spatial matching g applies directly for geometric verification and *re-ranking*.

Even though this functionality comes "for free", we optionally have the opportunity to use it at learning too, which we call *spatial training*. In particular, as discussed in Section 4.5, inliers $\mathcal{M}$ are split into two collections of local features per image. Then, *attention function* ϕ combines each collection with the corresponding feature tensor to yield the representation of one image given the other. Explicit spatial attention mechanisms are again ubiquitous but we rather focus attention on one image guided by another.

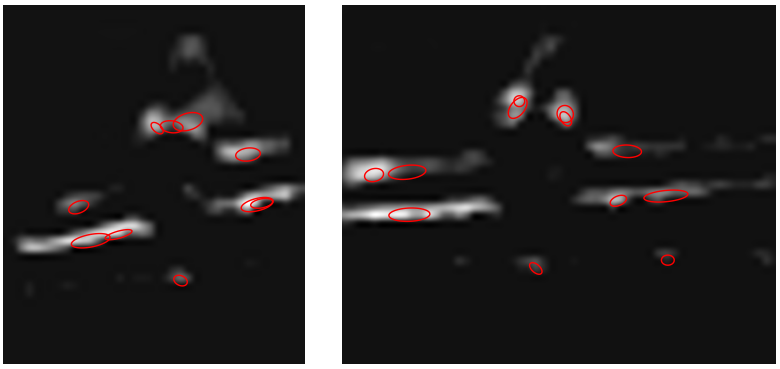


FIGURE 4.4 – Feature maps from one channel for two different views of a building in the $\mathcal{R}$Oxford dataset. Ellipses are fitted to the local features detected by MSER.

4.3.3 Local feature detection

To detect local features in each feature channel we use MSER by Matas *et al*. [Mat+02]. MSERs are defined over a 2-dimensional input, in our case over feature map $A^{\bullet j}$ of feature tensor A independently for each channel $j = 1, \ldots, k$. The extractor finds continuous regions R with all interior points having strictly higher response value than neighboring outer points. Regions satisfying a stability criterion [Mat+02] and passing location non-maxima suppression are selected. These features are appropriate for regions of arbitrary shape, including localized peaks, blobs, elongated or even nested regions.

When MSERs are used as image features, the response value is either the image intesity (MSER$^+$) or the reverse intensity (MSER$^-$). In our case, only regions of high CNN activations in sparse feature maps are of interest, and hence only one type of MSERs are extracted directly over the feature map responses.

4.3.4 Local feature representation

For each MSER R detected in channel j we compute a scalar value v representing strength. It is pooled over the spatial support of R in feature map $A^{\bullet j}$ as $v := \text{pool}_{r \in R}\, A^{\bullet j}(r)$. Here pool can be any pooling operation like max, mean, or generalized mean. We also fit an ellipse by matching its first and second moments, *i.e*. its 2×1 mean (position) vector μ and 2×2 covariance matrix (local shape) Σ. For instance, Figure 4.4 shows an example of ellipses fitted to the MSER detected on feature maps of one channel for two views of the Oxford Museum of Natural History. Ellipses are well aligned in the two views. The local feature corresponding to R is then represented by tuple $q := (\mu, \Sigma, v)$. Finally, we collect local features $\mathcal{Q} = (Q^{(1)}, \ldots, Q^{(k)})$ where $Q^{(j)}$ contains the local features q found in channel j. The entire operation is denoted by $\mathcal{Q} := d(A)$.

To treat feature channels as visual words, we assume that features are uncorrelated, which does not hold in practice as indicated by the fact that whitening boosts performance. The same filter may respond to a variety of input patterns and worse, several filters may respond to the same pattern. This can increase the level of interference in negative image pairs. For this reason we apply Non Maximum Suppression (NMS) over all channels on the detected regions of each database image. Because local features are often small, we set a low Intersection over Union (IoU) threshold. We do not apply NMS to the query image in order to allow matches from any channel.

FIGURE 4.5 – Examples of our DSM between images from $\mathcal{R}$Oxford and $\mathcal{R}$Paris benchmarks. Inlier features (ellipses) and correspondences (lines) shown in different colors.

4.3.5 Spatial matching

Given the local features $\mathcal{Q}_1, \mathcal{Q}_2$ of two images x_1, x_2, we use FSM [Phi+07] to find the geometric transformation T between the two images and the subsets of $\mathcal{Q}_1, \mathcal{Q}_2$ that are consistent with this transformation. Matching is based on *correspondences*, *i.e.* pairs of local features (q_1, q_2) from the two images. We allow pairs only between local features of the same channel, that is, q_1, q_2 are in $\mathcal{Q}_1^{(j)}, \mathcal{Q}_2^{(j)}$ respectively for some channel j. We thus treat channels as *visual words*, as if local features were assigned descriptors that were vector-quantized against a vocabulary and matched with the discrete metric. We begin with the *tentative correspondences* that is the set of all such pairs, $\mathcal{C} := (\mathcal{Q}_1^{(1)} \times \mathcal{Q}_2^{(1)}, \dots, \mathcal{Q}_1^{(k)} \times \mathcal{Q}_2^{(k)})$.

FSM is a variant of RANSAC [FB81] that generates a transformation hypothesis from a single correspondence. We adopt the linear 5-dof transformation which allows for translation, anisotropic scale and vertical shear but no rotation, assuming images are in "upright" orientation. Given a correspondence of two features $q_1 = (\mu_1, \Sigma_1, v_1)$ and $q_2 = (\mu_2, \Sigma_2, v_2)$, one finds from the two ellipses $(\mu_1, \Sigma_1), (\mu_2, \Sigma_2)$ the transformations T_1, T_2 that map them to the unit circle while maintaining the y-direction, and defines the transformation hypothesis $T = T_2^{-1} T_1$.

A hypothesis is evaluated based on the number of *inliers*, that is, correspondences that are consistent with it. Because tentative correspondences are not too many, all possible hypotheses are enumerated. Following [Phi+07], we are using LO-RANSAC [CMK03], which iteratively evaluates promising hypotheses by fitting a full transformation to inliers by least squares. The transformation T with the most inliers $\mathcal{M}$ is returned. The operation is denoted by $(\mathcal{M}, T) := g(\mathcal{Q}_1, \mathcal{Q}_2)$ and $\mathcal{M} = (\mathcal{M}^{(1)}, \dots, \mathcal{M}^{(k)})$ is a list of sets of inliers, one per channel.

4.3.6 Retrieval and re-ranking

In an image retrieval scenario, n database images $X = \{x_1, \dots, x_n\}$ are given in advance. For each image x_i with feature tensor A_i, its local features $\mathcal{Q}_i := d(A_i)$ are computed along with a global descriptor z_i spatially pooled directly from A_i again *e.g.* by max (2.12) or GeM pooling (2.13); A_i is then discarded. At query time, given query image x with feature tensor A, local features $\mathcal{Q} := d(A)$ and global descriptor z, we first rank $\{z_1, \dots, z_n\}$ by cosine similarity to z, and then the top-ranking images undergo spatial matching against $\mathcal{Q}$ according to $(\mathcal{M}_i, T_i) := g(\mathcal{Q}, \mathcal{Q}_i)$ and are re-ranked according to *similarity function* $s(\mathcal{M}_i)$. The most common choice, which we also follow in this work, is the number of inliers found, $s(\mathcal{M}_i) := \sum_{j=1}^{k} |\mathcal{M}_i^{(j)}|$.

In order to improve the performance, we follow a *multi-scale* approach where we compute feature tensors and local features from each input image at 3 different scales, but still keeping a fixed number of local features from all scales according to strength. During re-ranking, we then perform spatial matching on all 9 combinations of query and database image scales and keep the combination with maximum similarity. Matching examples are shown in Figure 4.5. As post-processing, we apply *supervised whitening* to global descriptors as in [RTC18] (discussed in Section 2.5) and query-time *diffusion* [Isc+17]. The latter, discussed in Section 2.4, is based on a nearest neighbor graph of the entire dataset X and is a second re-ranking process applied after spatial re-ranking. The precision of top-ranking images is important for diffusion [Rad+18], so spatial re-ranking is expected to help more its presence.

Method	Medium		Hard	
	$\mathcal{R}$Oxford	$\mathcal{R}$Paris	$\mathcal{R}$Oxford	$\mathcal{R}$Paris
R-MAC*	64.0	75.5	36.7	53.2
R-MAC*↑	63.9	75.5	35.6	53.3
R-GeM[RTC18]	64.7	77.2	38.5	56.3
R-GeM[RTC18]↑	65.3	77.3	39.6	56.6
R-MAC*+D	73.7	89.5	45.8	80.5
R-MAC*↑+D	73.9	89.9	45.6	81.0
R-GeM[RTC18]+D	69.8	88.9	40.5	78.5
R-GeM[RTC18]↑+D	70.1	89.1	41.5	78.9

TABLE 4.1 – Impact of ResNet (R) activation upsampling (↑) on mAP in $\mathcal{R}$Oxford and $\mathcal{R}$Paris [Rad+18]. MAC: max-pooling [TSJ16]; GeM: generalized-mean pooling [RTC18]; D: diffusion [Isc+17]. All results with supervised whitening [MM07]. Citation specifies the origin of the network or *: our re-training.

4.4 Experiments

In this section we evaluate the benefits of different parts of our DSM and compare our results with the state of the art on standard benchmarks.

4.4.1 Experimental setup

Test sets. We use the medium and hard setups of the revisited $\mathcal{R}$Oxford and $\mathcal{R}$Paris benchmarks [Rad+18]. We also use the large-scale benchmarks $\mathcal{R}$Oxf+$\mathcal{R}$1M and $\mathcal{R}$Par+$\mathcal{R}$1M, which are a combination of a set of 1M distractor images with the two small ones. We resize all images to a maximum size of 1024×1024. We evaluate performance by *mean average precision* (mAP) and *mean precision at 10* (mean Precision (mP)@10), as defined by the protocol [Rad+18].

Networks. We use VGG16 [SZ14] and Resnet101 [He+16], denoted simply as VGG (ResNet), or V (R) for short. In particular we use the versions trained by Radenovic *et al*. [RTC18] with GeM pooling. We also re-train them with max-pooling, on the same dataset of 120k Flickr images and the same structure-from-motion pipeline [RTC18]. Max-pooling is denoted by MAC [TSJ16] and re-training by *. ResNet has a resolution 4 times smaller than VGG. Therefore we remove the stride in the first *conv*5 convolutional layer and add a dilation factor of 2 in all following layers. We thus preserve the feature space while upsampling by 2. This upsampling requires no re-training and is denoted by ↑.

Global image representation. To rank images based on cosine similarity, we compute the multi-scale global representation described in Subsection 4.3.5. We extract descriptors at three different scales, related by factors $1, 1/\sqrt{2}$, and $1/2$, and pooled from the last activation maps using MAC [TSJ16] or GeM [RTC18]. The descriptors are pooled over scales into a single representation by either GeM for networks using GeM, or average for networks using MAC.

Local feature detection. We use the MSER implementation of VLFEAT [VF08] to detect regions in the last activation map of the network. We set the minimum diversity to 0.7 and maximum variation to 0.5. We observed that the step Δ needs adjusting according to the network/dataset used. We do this by setting Δ to 60% of the cumulative histogram of the activation values over the dataset.

Method	Medium				Hard			
	$\mathcal{R}$Oxford		$\mathcal{R}$Paris		$\mathcal{R}$Oxford		$\mathcal{R}$Paris	
	mAP	mP@10	mAP	mP@10	mAP	mP@10	mAP	mP@10
V	44.8	63.3	65.7	95.0	18.4	31.2	41.0	79.1
V+DSM	51.1	77.3	66.2	96.9	25.3	40.3	41.0	81.7
R↑	44.4	64.2	69.0	96.4	17.7	31.2	46.5	85.3
R↑+DSM	49.6	74.0	69.7	98.4	21.7	37.6	46.7	87.0
V+D	48.4	65.2	81.4	95.6	24.8	37.1	67.1	93.0
V+DSM+D	61.6	81.0	82.8	97.6	35.5	48.1	68.7	95.9
R↑+D	53.8	69.0	85.6	96.3	29.8	38.1	72.1	94.1
R↑+DSM+D	60.2	78.9	86.3	96.9	33.1	42.0	72.8	95.0

TABLE 4.2 – Impact of the proposed (DSM) on mAP and mP@10 on $\mathcal{R}$Oxford and $\mathcal{R}$Paris [Rad+18] with *off-the shelf* (pre-trained on Imagenet [Don+09]) VGG (V) and ResNet (R). ↑: upsampling; D: diffusion [Isc+17]. DSM: this work. All results with GeM pooling and supervised whitening.

Local image representation. To spatially verify images, we compute the multi-scale local representation introduced in Subsection 4.3.5. We fit an ellipse to each MSER region and for each ellipse we keep the covariance matrix, center position, channel id and maximum value. We discard activation maps with more than 20 features detected on query images, and 10 on database images. We apply NMS to features of database images with IoU threshold 0.2, which is restrictive enough even for small features. We rank features over all scales according to activation value and we select the top-ranking 512 features on VGG and 2048 on ResNet.

Re-ranking. After initial ranking by cosine similarity, we perform spatial matching between the query and the 100 top-ranked images as described in Subsection 4.3.5. Tentative correspondences originate from the same channels. We set the error threshold to 2 pixels (in the activation channel, not the image) and the maximal scale change to 3. Finally, we use the number of inliers to re-rank the top 100 images.

Spatially verified diffusion. We use diffusion [Isc+17], denoted by D, as a second post-processing step after spatial verification. It is based on a nearest neighbor graph of the global descriptors of the entire dataset, which is computed off-line. It starts from the top ranked images and finds more similar images according to manifold similarity. Diffusion is very powerful but sensitive to the quality of the initial top-ranked results. Thanks to our spatial matching, these results are more accurate. We take our 10 top-ranking spatially verified images and we compute a new score that is the product of the number of inliers and the descriptor similarity scores. We select the top 5 of them to initiate diffusion.

4.4.2 Ablation experiments

Upsampling. Table 4.1 shows the effect of upsampling on retrieval. This is not significant on MAC pooling. On GeM however, it results in perfomance increase by up to 1 mAP point on the hard setup of $\mathcal{R}$Oxford, both with and without diffusion. This can be explained by the higher resolution of the activation maps.

Off-the-shelf networks. Our re-ranking can be applied to any network, even as pre-trained on Imagenet [Don+09] (*off-the-shelf*). We use GeM pooling, which is better than MAC on such networks [Rad+18]. Table 4.2 shows the effect of DSM on $\mathcal{R}$Oxford and $\mathcal{R}$Paris medium

Method	Medium				Hard			
	$\mathcal{R}$Oxford		$\mathcal{R}$Paris		$\mathcal{R}$Oxford		$\mathcal{R}$Paris	
	mAP	mP@10	mAP	mP@10	mAP	mP@10	mAP	mP@10
V*	55.2	78.1	61.3	96.1	25.0	38.6	35.8	77.4
V*+DSM	58.2	83.4	61.9	98.9	28.4	46.6	36.2	80.4
V*+W	59.1	81.3	66.8	97.7	31.5	49.0	41.7	82.3
V*+W+DSM	60.0	84.3	67.0	98.6	32.5	53.1	42.0	82.3
R*↑	54.0	75.7	70.6	97.0	24.2	36.6	44.4	84.6
R*↑+DSM	57.4	80.4	70.9	98.7	28.4	42.6	44.3	84.9
R*↑+W	63.9	85.2	75.5	98.4	35.6	52.6	53.3	89.6
R*↑+W+DSM	62.7	83.7	75.7	98.7	35.4	51.6	53.1	88.6

TABLE 4.3 – Impact of the *supervised whitening* (W) [MM07] on mAP and mP@10 on $\mathcal{R}$Oxford and $\mathcal{R}$Paris [Rad+18]. Results with VGG (V) and ResNet (R), both with MAC pooling; ↑: upsampling; D: diffusion [Isc+17]; DSM: this work; *: our network re-training.

and hard setup. We improve results with and without diffusion. The gain is significant on $\mathcal{R}$Oxford, up to 13 mAP points on VGG-GeM with diffusion, medium setup. It is much smaller on $\mathcal{R}$Paris, where the perfomance is already 20 to 40 mAP points higher than on $\mathcal{R}$Oxford.

Whitening. We investigate the efficiency of our re-ranking with multi-scale global descriptors that are whitened or not. We use supervised whitening as in [MM07; RTC16], denoted by W. This is more powerful than PCA whitening [JC12]. As shown in Table 4.3, we improve significantly on non-whitened descriptors with both networks on $\mathcal{R}$Oxford. We gain 3 to 4 mAP points, as well as increasing mP@10. On the other hand, whitening boosts cosine similarity search, and gains 5 to 10 mAP points. Our improvement is more marginal or we lose up to one mAP point in this case.

Inliers. To evaluate the quality of matching, we check how many inliers are found for positive and negative images. In particular, Figure 4.6 shows the distribution of the number of inliers to all queries of $\mathcal{R}$Oxford with VGG-MAC for both positive and negative images. The distribution is similar over different networks and datasets. Negative images can be easily discriminated by having few inliers, but this may result in loosing positive ones. Contrary to conventional spatial matching, we do not use local descriptors. This is positive in terms of memory, but comes necessarily with lower quality of matches. However, the top-ranking spatially verified images *per query* are indeed accurate as indicated by mP@10, which is enough to initiate a better diffusion.

4.4.3 Comparison with the state-of-the-art

We conduct an extensive comparison of our method with baselines and additional state-of-the-art methods. All methods are tested on $\mathcal{R}$Oxford, $\mathcal{R}$Oxf+$\mathcal{R}$1M, $\mathcal{R}$Paris and $\mathcal{R}$Par+$\mathcal{R}$1M. We collect all results in Table 4.4.

Most baselines are improved by re-ranking, and all experiments on $\mathcal{R}$Oxford show consitent increase in performance. However, re-ranking is not perfect, as seen in Figure 4.6. In few cases the performance drops after re-ranking by up to one mAP point on $\mathcal{R}$Paris, in particular with the upsampled ResNet-GeM. We attribute the loss to two factors. One is a limited "vocabulary", based only on 512 or 2048 activation maps. The other is the fact that activation maps are highly correlated. This is exploited by whitening of the global descriptors, but tends to create correlated features.

Method	Medium								Hard							
	$\mathcal{R}$Oxford		$\mathcal{R}$Oxf+$\mathcal{R}$1M		$\mathcal{R}$Paris		$\mathcal{R}$Par+$\mathcal{R}$1M		$\mathcal{R}$Oxford		$\mathcal{R}$Oxf+$\mathcal{R}$1M		$\mathcal{R}$Paris		$\mathcal{R}$Par+$\mathcal{R}$1M	
	mAP	mP@10	mAP	mP@10	mAP	mP@10	mAP	mP@10	mAP	mP@10	mAP	mP@10	mAP	mP@10	mAP	mP@10
"DELF-ASMK*+SP" [Rad+18]	67.8	87.9	53.8	81.1	76.9	99.3	57.3	98.3	43.1	62.4	31.2	50.7	55.4	93.4	26.4	75.7
R-RMAC[Gor+17] [Rad+18]	60.9	78.1	39.3	62.1	78.9	96.9	54.8	93.9	32.4	50.0	12.5	24.9	59.4	86.1	28.0	70.0
V-MAC[RTC16]	58.4	81.1	39.7	68.6	66.8	97.7	42.4	92.6	30.5	48.0	17.9	27.9	42.0	82.9	17.7	63.7
V-MAC*	59.1	81.3	40.2	68.1	66.8	97.7	42.1	92.0	31.5	49.0	17.8	28.4	41.7	82.3	17.4	63.6
V-MAC*+DSM	60.0	84.3	42.2	71.0	67.0	98.6	42.5	94.7	32.5	53.1	19.4	31.6	42.0	82.3	17.7	66.0
R-MAC*↑	63.9	85.2	43.2	69.6	75.5	98.4	50.1	95.3	35.6	52.6	17.7	31.4	53.3	89.6	22.4	71.6
R-MAC*↑+DSM	62.7	83.7	44.4	72.3	75.7	98.7	50.4	96.4	35.4	51.6	20.6	32.3	53.1	88.6	22.7	72.1
V-GeM[RTC18]	61.9	82.7	42.6	68.1	69.3	97.9	45.4	94.1	33.7	51.0	19.0	29.4	44.3	83.7	19.1	64.9
V-GeM[RTC18]+DSM	63.0	85.5	43.9	72.9	69.2	98.4	45.4	94.7	34.5	54.0	19.9	32.9	43.9	82.7	19.5	67.6
R-GeM[RTC18]	64.7	84.7	45.2	71.7	77.2	98.1	52.3	95.3	38.5	53.0	19.9	34.9	56.3	89.1	24.7	73.3
R-GeM[RTC18]↑	65.3	86.3	46.1	73.4	77.3	98.3	52.6	95.4	39.6	54.6	22.2	36.4	56.6	89.4	24.8	73.6
R-GeM[RTC18]↑+DSM	65.3	87.1	47.6	76.4	77.4	99.1	52.8	96.7	39.2	55.3	23.2	37.9	56.2	89.9	25.0	74.6
Diffusion																
"DELF-HQE+SP" [Rad+18]	73.4	88.2	60.6	79.7	84.0	98.3	65.2	96.1	**50.3**	67.2	37.9	56.1	69.3	93.7	35.8	69.1
"DELF-ASMK*+SP"→D† [Rad+18]	75.0	87.9	68.7	83.6	**90.5**	98.0	**86.6**	98.1	48.3	64.0	39.4	55.7	**81.2**	95.6	**74.2**	94.6
V-MAC*+D	67.7	86.1	56.8	78.6	85.6	97.6	78.6	96.4	39.8	51.1	29.4	46.0	73.9	94.1	62.4	91.9
V-MAC*+DSM+D	72.0	90.6	59.2	80.1	86.4	98.9	79.3	97.1	43.9	56.0	32.0	47.4	75.1	95.4	63.4	92.9
R-MAC*↑+D	73.9	87.9	61.3	80.6	89.9	96.1	83.0	95.1	45.6	62.2	31.9	48.4	81.0	94.3	68.6	91.9
R-MAC*↑+DSM+D	**76.9**	90.7	65.7	83.9	90.1	96.4	84.0	95.3	49.4	64.7	35.7	51.3	**81.2**	93.3	70.1	92.6
V-GeM[RTC18]+D	69.6	84.7	60.4	79.4	85.6	97.1	80.7	97.1	41.1	51.1	33.1	49.6	73.9	93.7	65.3	93.1
V-GeM[RTC18]+DSM+D	72.8	89.0	63.2	83.7	85.7	96.1	80.1	95.7	45.4	57.1	35.4	53.7	74.2	93.3	65.2	91.9
R-GeM[RTC18]+D	69.8	84.0	61.5	77.1	88.9	96.9	84.9	95.9	40.5	54.4	33.1	48.2	78.5	94.6	71.6	93.7
R-GeM[RTC18]↑+D	70.1	84.3	67.5	79.0	89.1	97.3	85.0	96.6	41.5	54.4	39.6	53.0	78.9	95.1	72.0	94.1
R-GeM[RTC18]↑+DSM+D	75.0	89.6	**70.2**	84.5	89.3	97.1	84.8	95.3	46.2	60.6	**41.9**	54.9	79.3	95.1	72.0	93.4

TABLE 4.4 – mAP and mP@10 *state-of-the-art* on the full benchmark [Rad+18]. We use VGG (V) and ResNet (R), with MAC or GeM pooling. ↑: upsampling; *: our re-training; D: diffusion [Isc+17]. DSM: this work. Results citing [Rad+18] are as reported in that work and are combining DELF [Noh+16], ASMK* [TAJ16] and HQE [TJ14]. SP: spatial matching [Phi+07]; D†: diffusion on the graph obtained by [Gor+17]. The remaining citations specify where we took the trained network from.

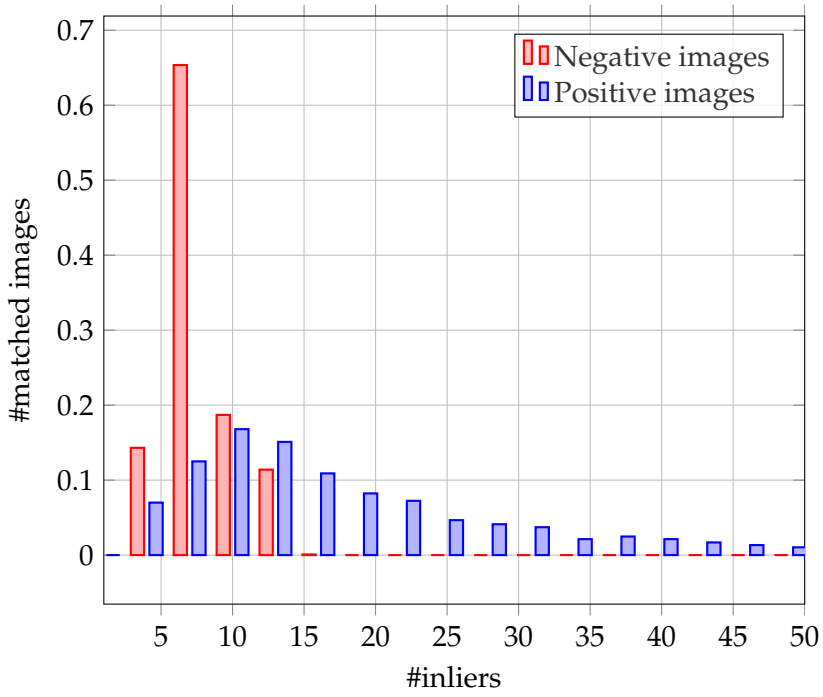


FIGURE 4.6 – Distribution of number of inliers for positive and negative database images over all queries of $\mathcal{R}$Oxford, using VGG-MAC.

The performance is improved significantly when diffusion is initiated from top-ranked spatially verified images. Diffusion only needs few relevant images, and we are able to provide these images thanks to spatial matching. We improve on most datasets, networks and pooling options in this case. The gain is more pronounced on $\mathcal{R}$Oxford, and is up to 5 mAP or 6 mP@10 points.

Finally, the proposed method with spatially verified diffusion outperforms approaches based on deep local features in a number of cases. In particular, we compare with the best performing and expensive version of DELF [Noh+16] proposed and evaluated by [Rad+18]. Apart from spatial verification by [Phi+07] on the 100 top-ranking images, this version is using two independent representations. One is ASMK* [TAJ16], based on 128-dimensional descriptors of 1000 DELF features per image, and used for initial ranking. Another is a global descriptor obtained by ResNet-R-MAC [Gor+17], and used for diffusion (D†) after spatial verification as in this work. By contrast, our global and local representations are obtained from the same activation tensor, and we do not use any local descriptors or their quantized versions.

4.5 Spatial training

We have shown that our approach DSM can be applied on pre-trained networks. The used networks are trained for image retrieval following [RTC18], with a siamese architecture. This setting is appropriate for spatial matching as well. Indeed, our approach can be added in the training and help discovering where matching objects are located given a pair of images. In this section we present experiments on spatial training that have not been presented in our published work.

4.5.1 Method

Although spatial matching can apply to any backbone network without any change in the architecture, it can be used at learning too. For image retrieval, it is typical to start with networks pre-trained on ImageNet and fine-tune them on a new dataset for *manifold learning*, for instance using the contrastive [RTC18] or triplet [Gor+16a] loss on image-level labels obtained automatically from data. We follow the pipeline of [RTC18], where pairs of images are compared by cosine similarity on global descriptors, obtained by either max or GeM pooling. As shown in Figure 4.7, our architecture is siamese like [RTC18], the only difference being that we apply spatial matching before computing a similarity.

The idea is that spatial matching finds what is common in the two images and then *focuses attention* to the common part in each image. In a positive image pair, the common part is what should really be matching because anything else is really irrelevant and trying to strengthen similarity on irrelevant input can be damaging. In a negative pair on the other hand, whatever is found by spatial matching is the part of the two images that looks most similar according to the current representation, so it should be the hardest. Thus, spatial matching can provide for a finer selection of input to the learning process than entire images.

In particular, inliers $\mathcal{M}$ are split into two collections of local features $\mathcal{Q}_{1|2}, \mathcal{Q}_{2|1}$, where $\mathcal{Q}_{i|j}$ refers to local features of image x_i verified against x_j. Then, *attention function* ϕ combines each collection $\mathcal{Q}_{i|j}$ with the corresponding feature tensor A_i to yield the representation $A_{i|j} := \phi(A_i, \mathcal{Q}_{i|j})$ of image x_i with given x_j, for $i, j = 1, 2$ and $i \neq j$. The simplest form of function $\phi(A, \mathcal{Q})$, which we follow in this work, is to pool A into a global descriptor. This is the same max or GeM pooling operation, only over the regions of the local features $\mathcal{Q}$; in channels without features (*i.e.*, inliers), we pool globally instead. Since pooling over each region has been done already, this is the same as pooling the local feature strengths. Representations $A_{1|2}, A_{2|1}$ are then matched with *learning similarity function* s, which is cosine similarity in this case. Finally, loss function ℓ applies, which is contrastive as in [RTC18].

The above pooling operation is also the same as masking every feature map $A^{(j)}$ according to the MSER regions of $\mathcal{Q}^{(j)}$ and then pooling over the masked tensor. This is possible at learning only because otherwise the regions are not stored in $\mathcal{Q}$. We apply this approach in practice as it is efficient on GPU and differentiable with respect to A. While of course it is not differentiable with respect to the masks that result from MSER detection and spatial matching. The argument is that there are two streams multiplied here, one of which ($\mathcal{Q}$) is an attention signal to the other (A), in which case just one (A) being differentiable is enough. In contrast to works that attempt to come up *e.g.* with differentiable version of RANSAC [Roc+18; Bra+17], which necessarily needs to be simplified in many aspects, we are thus allowed to use a well-tested robust version.

4.5.2 Experiments

Experimental setup. We re-use experimental setup presented in Section 4.4 while investigating the impact of *spatial training* on results. We consider a trained network following [RTC18] and we re-train it for 30 epochs with spatial matching (presented 4.5.1) using the pipeline given by [RTC18]. We detect features on both images of a training pair, using MSER [Mat+02], and match them with FSM [Phi+07].

Baseline. For fair comparison, we also re-train a network for 30 epochs without spatial matching. Results on VGG16 [SZ14] and Resnet101 [He+16] trained following [TSJ16; RTC18]

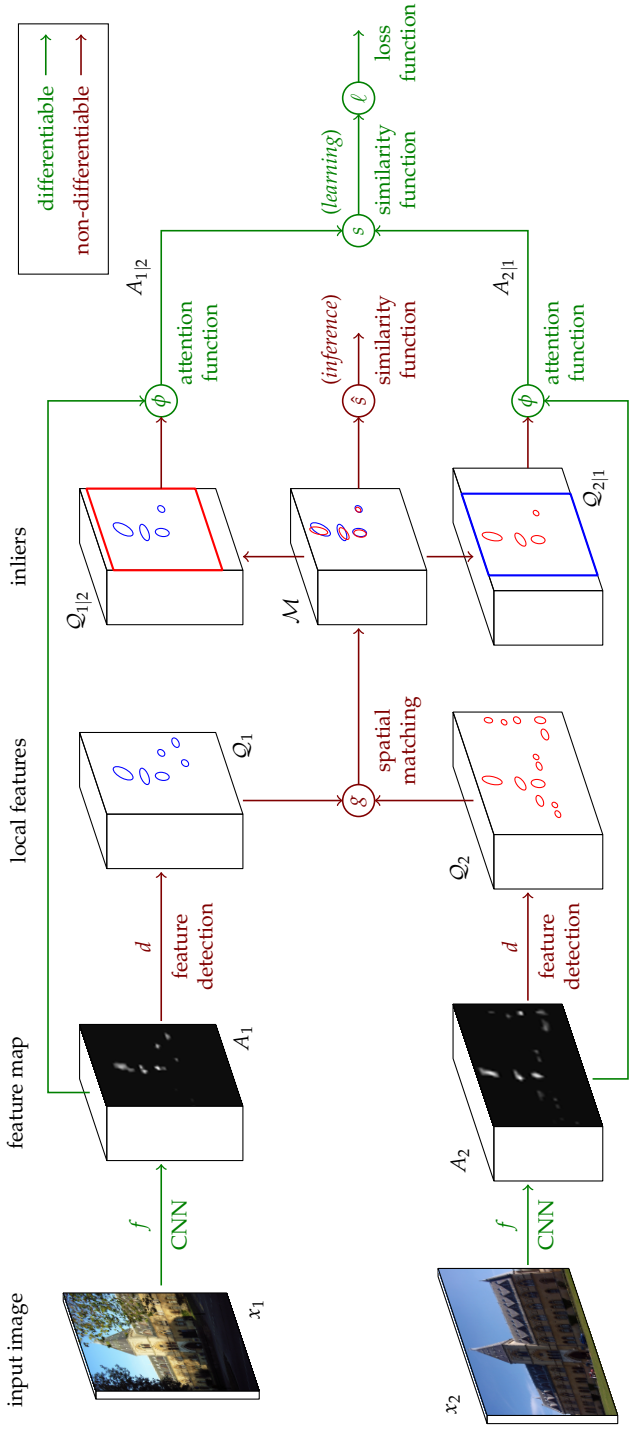


FIGURE 4.7 – Spatial training architecture. Two input images x_1, x_2 are mapped by network f to feature tensors A_1, A_2 respectively. Sparse *local features* $\mathcal{Q}_1, \mathcal{Q}_2$ extracted by *detector* d undergo *spatial matching* g, resulting in a collection of inliers $\mathcal{M}$. At inference, the similarity function $\hat{s}$ applies to this collection. While training, matching features $\mathcal{Q}_{1|2}, \mathcal{Q}_{2|1}$ are used to mask feature tensors A_1, A_2, respectively, following the *attention function* ϕ. Resulting feature tensors $A_{1|2}, A_{2|1}$ are compared using similarity function s. Contrastive loss ℓ is applied following [RTC18]. The spatial matching operation g is not differentiable.

Method	Medium		Hard	
	$\mathcal{R}$Oxford	$\mathcal{R}$Paris	$\mathcal{R}$Oxford	$\mathcal{R}$Paris
V-MAC*	59.1	66.8	31.5	41.7
V-MAC*+T_R	59.7	65.7	31.5	40.0
V-MAC*+T_{SP}	60.0	65.7	31.7	40.2
R-MAC*↑	63.9	75.5	35.6	53.3
R-MAC*↑+T_R	63.9	74.8	36.4	52.7
R-MAC*↑+T_{SP}	64.2	74.7	37.6	52.5
R-GeM[RTC18]↑	65.3	77.3	39.6	56.6
R-GeM[RTC18]↑+T_R	64.6	77.0	39.6	55.7
R-GeM[RTC18]↑+T_{SP}	64.6	76.9	39.5	55.7

TABLE 4.5 – Impact of re-training with spatial training +T_{SP} and without +T_R on mAP in $\mathcal{R}$Oxford and $\mathcal{R}$Paris [Rad+18]. MAC: max-pooling [TSJ16]; GeM: generalized-mean pooling [RTC18]; D: diffusion [Isc+17]. All results with supervised whitening [MM07]. Citation specifies the origin of the network or *: our re-training.

are presented in Table 4.5 (denoted by +T_R). We observe that performing re-training doesn't significantly alter trained network, between -0.7 mAP (on R-GeM[RTC18]↑) to +0.3 mAP (on R-MAC*↑).

Results. Re-training with spatial matching (+T_{SP}) improves over re-training without (+T_R) by 0.3 mAP when MAC pooling, and doesn't affect results when using GeM pooling. The impact of spatial re-training alone is therefore not significant. We additionally evaluate if spatial training helps our spatial matching method DSM, previously presented.

In Table 4.6 we observe consistent gain using spatial training on VGG16 with MAC pooling, both with/without diffusion and with/without DSM, by 0.8 to 1.9 mAP improvement on $\mathcal{R}$Oxford to the baseline without training. Notably, re-training without matching can hurt DSM results (-0.8 mAP on V-MAC*) or not improve (on V-MAC*+ DSM). In Table 4.7, we observe that spatial training also improves results obtained with Resnet101 and MAC pooling in all setups except when using both diffusion and DSM–although $\mathcal{R}$Oxford always benefits from our spatial training.

When using GeM pooling, results are improved only when using diffusion as showed in Table 4.8. The most significant gain is achieved on $\mathcal{R}$Oxford in the hard setup, obtaining +3 mAP. However gains are equivalent, and sometimes better, when re-training without spatial matching. So it appears that ResNet101 benefits from a longer training when trained using GeM pooling.

4.6 Discussion

Our experiments validate that the proposed representation for spatial verification achieves state-of-art performance across a number of different datasets, networks and pooling mechanisms. This representation arises naturally in the existing convolutional activations of off-the-shelf or fine-tuned networks, without any particular effort to detect local features or extract local descriptors on image patches. It does not require any network modification or retraining. It is a significant step towards bridging the gap between global descriptors, which are efficient for initial ranking using nearest neighbor search, and local representations, which are compatible with spatial verification. We also show that spatial training can

Method	Medium		Hard	
	$\mathcal{R}$Oxford	$\mathcal{R}$Paris	$\mathcal{R}$Oxford	$\mathcal{R}$Paris
V-MAC*	59.1	66.8	31.5	41.7
V-MAC*+T_R	59.7	65.7	31.5	40.0
V-MAC*+T_{SP}	60.0	65.7	31.7	40.2
V-MAC*+ DSM	60.0	67.0	32.5	42.0
V-MAC*+T_R+ DSM	59.2	65.5	31.3	39.6
V-MAC*+T_{SP}+ DSM	61.1	65.4	33.3	39.6
V-MAC*+ D	67.7	85.6	39.8	73.9
V-MAC*+T_R+ D	69.3	86.1	43.2	74.8
V-MAC*+T_{SP}+ D	69.6	86.4	43.4	75.4
V-MAC*+ DSM+ D	72.0	86.4	43.9	75.1
V-MAC*+T_R+ DSM+ D	72.0	85.7	45.5	74.7
V-MAC*+T_{SP}+ DSM+ D	73.2	86.4	46.2	75.7

TABLE 4.6 – Impact ofre-training with spatial training (+T_{SP}) and without (+T_R) on on mAP in $\mathcal{R}$Oxford and $\mathcal{R}$Paris [Rad+18]. MAC: max-pooling [TSJ16]; D: diffusion [Isc+17]. All results with supervised whitening [MM07]. Citation specifies the origin of the network or *: our re-training.

Method	Medium		Hard	
	$\mathcal{R}$Oxford	$\mathcal{R}$Paris	$\mathcal{R}$Oxford	$\mathcal{R}$Paris
R-MAC*↑	63.9	75.5	35.6	53.3
R-MAC*↑+T_R	63.9	74.8	36.4	52.7
R-MAC*↑+T_{SP}	64.2	74.7	37.6	52.5
R-MAC*↑+ DSM	62.7	75.7	35.4	53.1
R-MAC*↑+T_R+ DSM	63.1	75.0	35.5	52.5
R-MAC*↑+T_{SP}+ DSM	63.1	74.8	35.4	52.1
R-MAC*↑+ D	73.9	89.9	45.6	81.0
R-MAC*↑+T_R+ D	74.1	89.6	47.8	80.3
R-MAC*↑+T_{SP}+ D	74.5	89.5	48.3	80.2
R-MAC*↑+ DSM+ D	76.9	90.1	49.4	81.2
R-MAC*↑+T_R+ DSM+ D	76.8	89.8	50.5	80.6
R-MAC*↑+T_{SP}+ DSM+ D	76.6	89.7	50.8	80.5

TABLE 4.7 – Impact of re-training with spatial training (+T_{SP}) and without (+T_R) on on mAP in $\mathcal{R}$Oxford and $\mathcal{R}$Paris [Rad+18]. MAC: max-pooling [TSJ16]; D: diffusion [Isc+17]. All results with supervised whitening [MM07]. Citation specifies the origin of the network or *: our re-training.

Method	Medium		Hard	
	$\mathcal{R}$Oxford	$\mathcal{R}$Paris	$\mathcal{R}$Oxford	$\mathcal{R}$Paris
R-GeM[RTC18]↑	65.3	77.3	39.6	56.6
R-GeM[RTC18]↑+T_R	64.6	77.0	39.6	55.7
R-GeM[RTC18]↑+T_{SP}	64.6	76.9	39.5	55.7
R-GeM[RTC18]↑+ DSM	65.3	77.4	39.2	56.2
R-GeM[RTC18]↑+T_R+ DSM	64.6	77.0	37.5	55.4
R-GeM[RTC18]↑+T_{SP}+ DSM	64.7	77.1	38.2	55.4
R-GeM[RTC18]↑+ D	70.1	89.1	41.5	78.9
R-GeM[RTC18]↑+T_R+ D	72.2	88.7	44.5	78.3
R-GeM[RTC18]↑+T_{SP}+ D	72.1	88.6	44.6	78.1
R-GeM[RTC18]↑+ DSM+ D	75.0	89.3	46.2	79.3
R-GeM[RTC18]↑+T_R+ DSM+ D	76.4	88.8	49.8	78.7
R-GeM[RTC18]↑+T_{SP}+ DSM+ D	75.5	88.8	48.4	78.6

TABLE 4.8 – Impact of re-training with spatial training (+T_{SP}) and without (+T_R) on mAP in $\mathcal{R}$Oxford and $\mathcal{R}$Paris [Rad+18]. GeM: generalized-mean pooling [RTC18]; D: diffusion [Isc+17]. All results with supervised whitening [MM07]. Citation specifies the origin of the network or *: our re-training.

help improving activation maps from which more robust local features can be discovered following our DSM.

Of course, the activation channels are not the most appropriate by construction to replace a visual vocabulary. This means that our representation, while being very compact, is not as powerful as storing *e.g.* hundreds of local descriptors per image. Nonetheless, we still demonstrate that it is enough to provide high-quality top-ranking images to initiate diffusion, which then brings excellent results.

Chapter 5

Object Discovery and retrieval

Contents

In Chapter 4 we have introduced a way to exploit information contained in CNN feature maps. In this chapter we investigate another way: discovering objects of interest in an entire image collection and using them to define a robust global representation. Our motivation is to address the challenge of severe background clutter in large-scale image retrieval. Global descriptors, that are popular due to their memory and search efficiency, are especially prone to corruption by such a clutter. Eliminating the impact of the clutter on the image descriptor increases the chance of retrieving relevant images and prevents topic drift due to actually retrieving the clutter in the case of query expansion. Here we introduce a novel salient region detection method that captures, in an unsupervised manner, patterns that are both discriminative and common in an image dataset. Saliency is based on a centrality measure of a nearest neighbor graph constructed from regional CNN representations of dataset images. The proposed method [Sim+19] exploits recent CNN architectures trained for object retrieval to construct the image representation from the salient regions. We improve particular object retrieval on challenging datasets containing small objects.

We introduce the context of our work in Section 5.1. Section 5.2 discusses our contributions against related work. Section 5.3 describes our methodology including our pooling scheme in Subsection 5.3.3 and our object discovery approach in Subsection 5.3.8. We present experimental results in Section 5.4 and draw conclusions in Section 5.5.

5.1 Introduction

Particular object retrieval becomes very challenging when the object of interest is covering a small part of the image. In this case, the amount of relevant information is significantly reduced. Large objects might be partially occluded, while small objects are on a

FIGURE 5.1 – The saliency map (right) computed for an input image (left) based on common-structure analysis on *Instre* dataset. Background clutter and objects not relevant for this dataset are automatically removed. The image is represented only by the region detected on the saliency map.

background that covers most of the image. A combination of both, occlusion and cluttered background, is not rare either. These conditions naturally arise from image acquisition and make naive approaches fail, including global template matching or semi-robust template matching [OT06].

Ideally, image descriptors should be extracted only from the relevant part of the image, suppressing the irrelevant clutter and occlusions. In this paper, we attempt to determine the regions containing the relevant information, as shown in Figure 5.1, in a fully unsupervised manner.

Methods based on robust matching of *hand-crafted local features* are naturally insensitive to occlusion and background clutter. The locality of the features allows to match small parts of images in regions containing the object of interest, while the incorrect matches are typically removed by robust geometric consistency check [Phi+07]. Methods based on efficient matching of vector-quantized local-feature descriptors were introduced in context of image retrieval by Sivic and Zisserman [SZ03].

Retrieval methods based on descriptors extracted by CNNs have become popular because they combine good precision and recall, efficiency of the search, and reasonable memory footprint [Bab+14; Raz+16]. Deep neural networks are capable of learning, to some extent, what information in the image is relevant, which results in a good performance even with global descriptors [TSJ16; BL15; KMO15]. However, if the signal to noise ratio is low, *e.g.* the object is relatively small, multiple objects are present, *etc.*, the global CNN descriptors fail [Isc+17; Isc+18a].

A class of methods inspired by *object detection* have recently emerged. Instead of attempting to match the whole image to the query, the problem is changed to finding a rectangular region in the image that best matches the query [TSJ16; Sal+16]. An inefficient search by sliding window is intractable for large collections of images. The exhaustive enumeration is approximated by similarity evaluation on a number of pre-selected regions. The regions are either selected geometrically to cover the whole image at different scales, as in R-MAC [TSJ16], or by considering the content by object or region proposal methods [Sal+16; Son+17; Gor+16a].

Another direction of suppressing irrelevant content is saliency detection [KMO15; Noh+16]. For each image, a saliency map, that captures more general region shapes compared to (a small set of) rectangles, is first estimated. The contribution of each pixel (or region) is then proportional to the saliency of that location.

In this work we introduce a very simple pooling scheme that inherits the properties of both saliency detection and region based pooling and that, like all previous approaches, is applied to each image in the database *independently*. In addition, we investigate the use of the resulting regional representation for automatic, offline object discovery and suppression of background clutter, which considers the image collection *as a whole*. Unlike previous approaches, we do this in an unsupervised way. As a consequence, our representation takes two saliency detection steps into account. One that acts per image and depends solely on its content and another that considers the image collection as a whole and captures frequently appearing objects.

In both cases, we derive a *global* representation that outperforms comparable state-of-the-art methods in retrieving small objects on standard benchmarks, while the memory footprint and online cost is only a fraction of more powerful *regional* representations [Raz+16; Isc+17]. Moreover, we show that our representation benefits significantly from *query expansion* methods.

We make the following contributions:

1. We show that it is possible to select a set of candidate image regions based on CNN activations of off-the-shelf networks trained on retrieval tasks without bounding box annotations.

2. We obtain a global dataset-dependent "significance" of such regions via a neighborhood graph.

3. We represent the dataset by sampling and pooling CNN activations according to a dense saliency map of automatically discovered objects.

4. We thereby improve the state of the art on particular object retrieval, especially in a large scale dataset containing small objects.

Compared to the prior version of this work [Sim+18], we make the following improvements. We apply our method to the recent GeM representation [RTC18]. We use a multi-scale representation in saliency computation. We analyze multiple graph centrality measures and validate their impact on detection quality using bounding box annotations. Finally, we evaluate using the recently revisited and challenging $\mathcal{R}$Oxford and $\mathcal{R}$Paris benchmarks [Rad+18].

5.2 Related work

Local features and geometric matching offer an attractive way for retrieval systems to handle occlusions and clutter as discussed in Chapter 4. One of their drawbacks is high query complexity and large storage cost when an image is represented by several thousands features. Many methods attempt to decrease the amount of indexed features by removing background clutter while maintaining the relevant information. The selection procedure is

either applied independently per image or considers an image collection as a whole. Common examples of the former case are bursty feature detection [SAJ15], symmetry detection [TKA12] or use of semantic segmentation [AZ14; Ome+08]. The methods of the second category, are scalable enough to jointly process the whole collection and perform feature selection by the following assumption. A feature that repeats over multiple instances of the same object in the dataset is likely to appear in novel views of the object too. Representative cases are common object discovery [TL09; TAJ16], co-occurrence detection [CM10], or methods using GPS information [Gam+09; KSP10].

The work by Turcot and Lowe [TL09] performs pairwise spatial verification on handcrafted local features across all images and only indexes verified features. With an additional off-line cost, the on-line stage is sped up and the memory footprint is reduced. However, unique views of objects are not verified and thus discarded. In this work, we address a similar selection problem based on more powerful CNN-based representation rather than local features.

Recent advances on deep learning [Azi+14; TSJ16; KMO15; Gor+16b; RTC16] dispense with the large memory footprint by using global descriptors and cast the problem of instance search as Euclidean nearest neighbor search. Nevertheless, background clutter and occlusion are better handled by regional representation. Regional descriptors significantly increase the performance when they are indexed independently [Raz+16; Isc+17] but this comes at a prohibited memory and computational cost for large scale scenarios. RPNs are applied either off-the-shelf [Sal+16] or after fine-tuning [Son+17] for instance search. The RPNs reduce the number of regions per image only to the order of tens. A new dataset including bounding box annotations on landmark images allows training a network specifically for image retrieval [Tei+19]. Our work focuses on aggregating regional representation that keeps the complexity low but we rather detect regions around salient objects and objects that frequently appear in the dataset.

Recent work uses CNN activation statistics to construct a saliency map in an unsupervised manner. These methods consider each dataset image independently. CroW [KMO15] computes spatial and channel weights based on activation maps, and weighs each feature accordingly. Similarly, Laskar and Kannala [LK17] weigh each R-MAC region. Traditional methods are applied on top of CNN activations for the same purpose. Jeong *et al*. [Jeo+17] use the Hessian-affine detector on activation maps to obtain repeatable regions. Pang *et al*. [Pan+18] use heat diffusion within an image to eliminate bursty features and keep the discriminative ones.

Another line of research trains a network to estimate the saliency map or find regions of interest. Zhu *et al*. [Zhu+18b] train an attention layer applied over multiple scales and use it for visual place recognition. This way, the network learns to discard background clutter. Similarly, Kim and Yoon [KY18] train a regional attention network to focus on important parts of an image. Jimenez *et al*. [JAG17] construct saliency maps and perform region detection to construct global image vectors, which is similar to our goal. However, they employ object detectors trained on ImageNet classes, which does not apply to networks fine-tuned for retrieval on new classes. Mohedano *et al*. [Moh+17] evaluate deep and non-deep saliency models in order to detect regions of interest from an image.

All aforementioned methods either act per image without supervision or learn from a collection with ground-truth. By contrast, our method operates on the entire dataset jointly and at the same time is fully unsupervised.

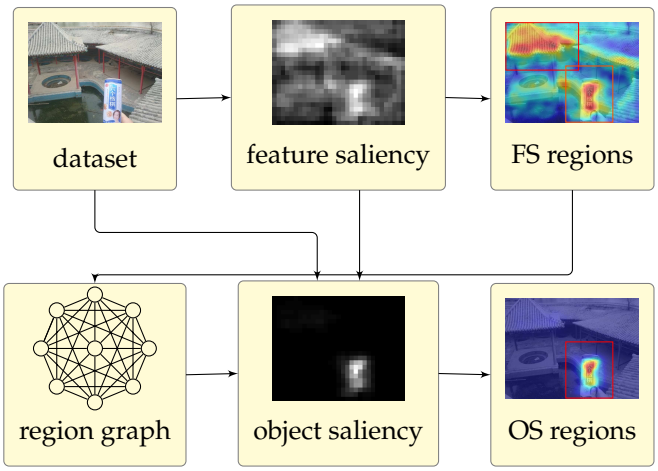


FIGURE 5.2 – Overview of our offline unsupervised process. On the top row, CNN activations of dataset images are used to extract a *feature saliency* map, on which a set of regions is detected. On the bottom row, a *centrality* measure is obtained per region from a region *k*-NN graph. Using this measure, a dense *object saliency* map is formed from the original CNN activations and the feature saliency. This map is focusing on objects automatically discovered in the dataset, with background clutter removed. Finally, another set of regions is detected on the object saliency map to extract descriptors and represent the dataset for retrieval.

The problem that this work is dealing with has been addressed previously in the literature but on different tasks. Common objects or regions in an image collection has been addressed in several different ways. Bagon *et al*. [Bag+10] use *Local Self-Similarity Descriptors*, while Rubinstein *et al*. [Rub+13] use SIFT flow. [Siv+05; Rus+06] apply statistical topic discovery models. More recently, Cho *et al*. [Cho+15] start from region proposals and perform matching to discover dominant objects, while Kwak *et al*. [Kwa+15] extend such a discovery process to videos. A relevant work is Kim *et al*. [KT09], who start from candidate regions found by segmentation and select regions of interest using PageRank [Pag+99]. We apply a similar idea to instance-level discovery based on CNN feature maps; we also use apply discovered objects to improve image representation for retrieval.

5.3 Method

Like [TL09], our objective is to remove transient and non-distinctive objects as in Figure 5.1 and rather focus on objects appearing frequently in a dataset. Beginning with the activation map of a convolutional layer in a CNN, one would need access to a local representation to automatically discover such objects. On the other hand, knowing what these objects are would help forming a local representation by selecting regions depicting them, which appears to be a chicken-and-egg problem. Without an initial region selection, we risk "discovering" uninformative but frequently appearing "stuff"-like patches, for instance sky.

5.3.1 Overview

Fortunately, it is possible to make an initial selection based on CNN activations alone, without any training and without bounding box annotations. As described in Subsection 5.3.3, the mechanism is inspired by CroW [KMO16] and Grad-CAM [Sel+17] and generates a *feature saliency* map. This initiates our offline analysis illustrated in Figure 5.2. A small set of rectangular regions is detected per image from this map as discussed in Subsection 5.3.4.

This first round of detection is applied independently per image and depends only on its content.

Each region in the dataset is associated to a feature saliency score and a visual descriptor, pooled from the activation map of the corresponding image, as discussed in Subsection 5.3.5. It is now possible to compute a *centrality* score per region, representing the "significance" of each region in the dataset. This is based on a region k-NN graph and is discussed in Subsection 5.3.6 and Subsection 5.3.7.

Now, given a new image, we can infer the "significance" of every region from its nearest neighbors in the graph, yielding a dense *object saliency* map as discussed in Subsection 5.3.8. This is a regression problem and we suggest a non-parametric k-NN solution. Finally, we detect a small set of rectangular regions on this saliency map and extract a global descriptor to represent dataset images for retrieval, as discussed in Subsection 5.3.9. This second detection procedure takes into account all salient and repeating objects appearing in the dataset.

The entire process is fully unsupervised and only assumes on the-shelf networks trained on a classification or retrieval task without bounding box annotations.

5.3.2 Notation

Following notations given in Table 1, we represent the activation map of a convolutional layer as a non-negative 3d tensor $A \in \mathbb{R}^{h \times w \times k}$ where h, w are the spatial resolution (height, width) and c is the number of feature channels. The set of valid spatial positions is $P := [h] \times [w]$[1] and the set of all rectangles with vertices in P is denoted by $\mathcal{R}$. By A^{pj} we represent an element of A at position $p \in P$ and channel $j \in [k]$. By $A^{\bullet j} \in \mathbb{R}^{h \times w}$ we denote the 2D feature map of A corresponding to channel $j \in [k]$. By $A^{p\bullet} \in \mathbb{R}^k$ we denote the vector containing all feature channels at position $p \in P$. By $\nu(\mathbf{x})$ we denote the ℓ^2-normalized vector $\mathbf{x} / \|\mathbf{x}\|_2$.

5.3.3 Feature saliency

Inspired by CroW [KMO15] and CAM [Zho+16], we construct a 2D saliency map of an image based on a convolution activation of that image alone. Following CroW, we compute an idf-like weight per channel $\mathbf{b} \in \mathbb{R}^k$ with elements

$$b_j = \log\left(\frac{(\mathbf{a} + \epsilon)^\top \mathbf{1}}{a_j + \epsilon}\right) \tag{5.1}$$

for $j \in [k]$, where $\mathbf{a} := \frac{1}{wh} \sum_{p \in P} \mathbb{1}[A^{p\bullet}] \in \mathbb{R}^k$ is the average number of nonzero elements per channel. We then compute a weighted sum over channels

$$F := \sum_{j \in [k]} b_j A^{\bullet j} \tag{5.2}$$

Finally, we obtain the 2D *feature saliency* (Feature Saliency (FS)) map $\hat{F} \in \mathbb{R}^{h \times w}$ by normalizing F according to [KMO15]. Examples of feature saliency maps are presented in Section 5.4. Despite its simplicity, this kind of saliency can focus on objects of interest when the background is simple enough. It fails however in the presence of clutter. Contrary to CroW, we

[1] Here, $[i]$ is the set $\{1, \ldots, i\}$ for $i \in \mathbb{N}$.

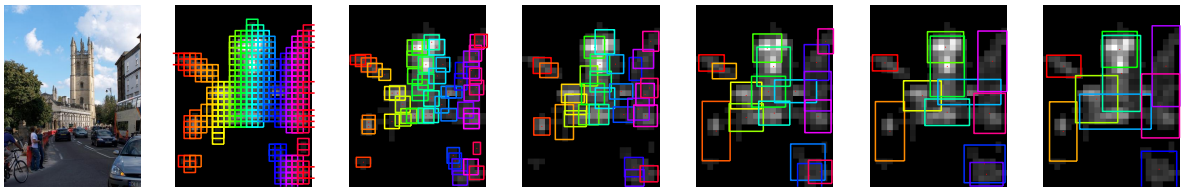


image $i = 0$ $m = 272$ $i = 2$ $m = 29$ $i = 3$ $m = 22$ $i = 5$ $m = 17$ $i = 7$ $m = 11$ $i = 14$ $m = 9$

FIGURE 5.3 – Evolution of regions during EGM iterations on the feature saliency map of an image of *Magdalen tower* from Oxford buildings dataset, shown on the left. Below each image we display the iteration i and the number of regions m.

use the feature channel weights when computing the 2D spatial weights, amplifying channels with sparse activation. This order of summation is the same as in CAM. However, we are working with channel weights obtained by a sparsity property on any convolutional layer, without any assumption on the network topology. CAM on the other hand, assumes global average pooling followed by a fully connected layer mapping channels to classes and uses the parameters of this layer to obtain a saliency map per class.

5.3.4 Region detection

We are given a 2D saliency map S, which can be either the feature saliency described in Subsection 5.3.3 or the object saliency described in Subsection 5.3.8. We use an Expanding Gaussian Mixture (EGM) model [AK12] to detect a number of salient rectangular regions. This is a variant of Expectation-Maximization (EM) that iteratively performs local averaging (E- and M-steps) interleaved with a selection process (P-step) similar to NMS. In doing so, it dynamically estimates the number of regions.

The original algorithm applies to point sets and isotropic Gaussian components. Here we extend it to functions, considering that a saliency map is a function $S : P \to \mathbb{R}$. We use it to fit a number of components, each modeling a rectangular region in 2D coordinate space. We also extend it to a diagonal covariance model, so that a rectangle is modeled by an axis-aligned ellipse.

In particular, given 2D saliency map $S \in \mathbb{R}^{h \times w}$, we represent it as a set of Gaussian functions $s_i : \mathbb{R}^2 \to \mathbb{R}$ with

$$s_i(\mathbf{x}) := S_{p_i} \mathcal{N}(\mathbf{x} | p_i, \sigma I_2) \tag{5.3}$$

for $i \in [\ell]$, $\mathbf{x} \in \mathbb{R}^2$ where $\mathcal{N}$ is the normal density, $\ell = |P|$ is the number of positions and we represent P as $\{p_1, \ldots, p_\ell\}$. Here, σ is a *scale* parameter that determines how coarse or fine the region representation will be for the given saliency map. Similarly, we represent components as Gaussian functions $q_\kappa : \mathbb{R}^2 \to \mathbb{R}$ with

$$q_\kappa(\mathbf{x}) := \pi_\kappa \mathcal{N}(\mathbf{x} | \mu_\kappa, \Sigma_\kappa) \tag{5.4}$$

for $\kappa \in [m]$, $\mathbf{x} \in \mathbb{R}^2$, where m is the number of components and $\pi_\kappa \in \mathbb{R}$, $\mu_\kappa \in \mathbb{R}^2$ and $\Sigma_\kappa \in \mathbb{R}^{2 \times 2}$ are the mixing coefficient, mean and diagonal covariance matrix respectively of component κ. Means represent region centers, while the (inverse) eigenvalues of covariance matrices represent heights and widths. We initialize components as $q_\kappa \leftarrow s_\kappa$ for $\kappa \in [m]$,

with $m \leftarrow \ell$. In the *expectation* (E)-step, we compute the *responsibility*

$$\gamma_{i\kappa} \leftarrow \frac{\langle s_i \rangle q_\kappa}{\sum_{j \in [m]} \langle s_i \rangle q_j} \tag{5.5}$$

of component $\kappa \in [m]$ for sample $i \in [\ell]$, where $\langle f \rangle g$ is the L^2 inner product of square-integrable functions $f, g : \mathbb{R}^d \to \mathbb{R}$, computed in closed form for Gaussian functions [AK12]. In the *maximization* (M)-step, we update parameters as

$$\pi_\kappa \leftarrow \frac{\ell_\kappa}{\ell} \tag{5.6}$$

$$\mu_\kappa \leftarrow \frac{1}{\ell_\kappa} \sum_{i=1}^{n} \gamma_{i\kappa} p_i \tag{5.7}$$

$$\Sigma_\kappa \leftarrow \frac{1}{\ell_\kappa} \sum_{i=1}^{n} \gamma_{i\kappa} \operatorname{diag}(p_i - \mu_\kappa)^{\circ 2} \tag{5.8}$$

where $\ell_\kappa := \sum_{i=1}^{n} \gamma_{i\kappa}$ is the effective number of points assigned to component κ and $X^{\circ 2} := X \circ X$ is the Hadamard product power for a vector or matrix X.

Finally, in the *purge* (P)-step, similarly to NMS, we process components in descending order of mixing coefficient and we decide whether to keep a component or not depending on its overlap with the collection of previously kept components. Overlap is measured by a generalized responsibility function similar to (5.5), and again inner products are given in closed form [AK12]. This means that the number of components m is potentially reducing at each iteration.

Figure 5.3 shows how regions are formed during EGM iterations, starting from one small region centered on each spatial position. We get 4 clean regions on the ground truth building, as well as 6 regions on background objects, which, although less salient, cannot be removed based on the feature saliency alone.

5.3.5 Region pooling and whitening

Given a rectangular region $R \in \mathcal{R}$ of an image with feature saliency map $\hat{F} \in \mathbb{R}^{h \times w}$, we associate to it *feature saliency* $f := \mu_{\hat{F}}(R) \in \mathbb{R}$, where

$$\mu_{\hat{F}}(R) := \frac{1}{|R|} \sum_{p \in R} \hat{F}_p \tag{5.9}$$

is the average of 2D map $\hat{F}$ over R.

In addition, given the activation map $A \in \mathbb{R}^{h \times w \times k}$ of the same image, it is standard practice that a descriptor is obtained by pooling over R. As detailed in Section 2.5, it is possible to use sum [BL15] (2.11), weighted sum [KMO15], max [Azi+14; TSJ16] (2.12), or GeM [RTC18] (2.13) pooling. We adopt the latter choice to extract descriptor $\mathbf{z} := m_A(R) \in \mathbb{R}^k$, where

$$m_A(R) := \left(\frac{1}{|R|} \sum_{q \in R} (A^{q \bullet})^\omega \right)^{\frac{1}{\omega}} \tag{5.10}$$

is the generalized-mean of 3d tensor A over R along the spatial dimensions, and ω is a pooling parameter that is learned. This has been the basis of fine-tuning in [RTC18] and produces a global description, referred to as GeM, in the special case where there is a single region $R = P$. In contrast, we detect a set of regions based on saliency maps in this work.

It is also standard practice to perform a sequence of post-processing steps including normalization, PCA and whitening (Section 2.5) . We follow [RTC18] in performing supervised whitening by simultaneous diagonalization [MM07]. In particular, given a descriptor $\mathbf{z} \in \mathbb{R}^k$, we ℓ^2-normalize, center, whiten, PCA-project and renormalize by function $w : \mathbb{R}^k \to \mathbb{R}^d$ to generate a d-dimensional descriptor:

$$w(\mathbf{z}) := \nu(U_{\mathcal{D}}^\top T_{\mathcal{D}}(\nu(\mathbf{z}) - \mu_{\mathcal{D}})). \tag{5.11}$$

Parameters $T_{\mathcal{D}}$, $U_{\mathcal{D}}$, $\mu_{\mathcal{D}}$ are trained on an independent labeled dataset $\mathcal{D} = \{Z, y\}$ where $Z = \{\mathbf{z}_1, \ldots, \mathbf{z}_n\} \subset \mathbb{R}^k$ and $y : [n]^2 \to \{0, 1\}$ maps a pair of descriptors to 1 if "positive" (similar) or 0 if "negative" (dissimilar). In particular, given set $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, define covariance matrix

$$C_l(X) := \frac{1}{|Y_l|} \sum_{(i,j) \in Y_l} (\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^\top \tag{5.12}$$

where $Y_l := \{(i, j) \in [n]^2 : y(i, j) = l\}$. Then, the $k \times k$ *whitening matrix* $T_{\mathcal{D}} := C_1(\hat{Z})^{-\frac{1}{2}}$ is the inverse square root of the intra-class covariance matrix of $\hat{Z} = \nu(Z)$. The *PCA matrix* $U_{\mathcal{D}}$ is the $c \times d$ matrix having as columns the top d eigenvectors of $C_0(T_{\mathcal{D}}\hat{Z})$, that is, the inter-class covariance matrix of the whitened counterpart of $\hat{Z}$. Finally, the *mean vector* is $\mu_{\mathcal{D}} := \frac{1}{n} \sum_{i \in [n]} \nu(\mathbf{z}_i)$.

5.3.6 Graph construction

Given an image dataset, we assume here a set of regions $\{R_1, \ldots, R_n\}$ are detected from the saliency maps (Subsection 5.3.4), a *feature saliency* vector $\mathbf{f} := (f_1, \ldots, f_n) \in \mathbb{R}^n$ is computed with the corresponding average saliency per region in (Equation 5.9), and a set of descriptors $V := \{\mathbf{v}_1, \ldots, \mathbf{v}_n\} \subset \mathbb{R}^d$ are extracted from the activation maps, whitened and normalized per region (Subsection 5.3.5).

Based on the above information, we construct a k-NN graph on those regions in order to compute a global *centrality* score per region as discussed in Subsection 5.3.7, which enables us to form an *object saliency* map on a new image, described in Subsection 5.3.8.

We construct a weighted undirected graph having the set of descriptors V as vertices following Section 2.4. It is described by the sparse symmetric nonnegative adjacency matrix $W \in \mathbb{R}^{n \times n}$, and its normalized form $\mathcal{W} := D^{-1/2} W D^{-1/2} \in \mathbb{R}^{n \times n}$ (2.3) where $D \in \mathbb{R}^{n \times n}$ is the degree matrix. We also construct the $n \times n$ regularized Laplacian $L_\alpha := D - \alpha W$ (2.6) and its normalized form $\mathcal{L}_\alpha := I_n - \alpha \mathcal{W}$ (2.7) following [Isc+17; Isc+18a].

5.3.7 Graph centrality

With the above definitions in place, the objective is to compute a vector $\mathbf{u} \in \mathbb{R}^n$ where each element u_i represents the significance of vertex $\mathbf{v}_i$ in the graph, for $i \in [n]$. We choose *graph centrality* [New10] to estimate $\mathbf{u}$. Centrality is a global measure of significance of vertices in a graph. Different definitions exist, each one reflecting a different kind of vertex

importance. Herein, we consider a number of centrality measures on adjacency matrix W, which we then evaluate in the experimental section.

Degree centrality is the simplest one to define and to compute. It is defined as the degree of each vertex, *i.e.* the diagonal of degree matrix D. Large value means that the vertex is connected to many other vertices with edges of large weight (similarity).

Eigenvector centrality, also known as eigencentrality [Bon87], corresponds to the dominant eigenvector of W. Centrality value of vertex $\mathbf{v}_i$ is given by the i-th element of this eigenvector. Large value means that the vertex is connected to many vertices which themselves have high centrality.

PageRank centrality is maybe the most well-known [Pag+99] centrality measure, and is a variant of the eigencentrality. It is given by the dominant left eigenvector of the transition matrix $\alpha D^{-1}W + (1-\alpha)J/n$, where J is an $n \times n$ matrix of ones. It is efficiently computed with power iteration. It models a random walk on the graph and its score represents the probability of visiting a vertex.

Betweenness centrality reflects the number of times a vertex is part of the shortest path between any two other vertices of the graph. In contrast to all previously mentioned measures that are computed based on edge importance (similarity w_{ij}), it is computed based on an edge cost. In this case, we are using Euclidean distance $||\mathbf{v}_i - \mathbf{v}_j||$.

Closeness centrality is inversely proportional to the average length of the shortest path to all other nodes. Closeness centrality for vertex $\mathbf{v}_i$ is equal to $\sum_{\mathbf{v}_j \neq \mathbf{v}_i} \frac{n-1}{|\mathbf{v}_i \to \mathbf{v}_j|}$, where $|\mathbf{v}_i \to \mathbf{v}_j|$ is the length of the shortest path between $\mathbf{v}_i$ and $\mathbf{v}_j$. Similarly to betweenness, closeness applies to edge cost, and we use the Euclidean distance.

Katz centrality [Kat53] is given by the solution $\mathbf{u}^* \in \mathbb{R}^n$ of the linear system

$$\mathcal{L}_\alpha \mathbf{u} = (1-\alpha)\mathbf{1}. \tag{5.13}$$

As in [Isc+17], we solve this system by the *conjugate gradient method* (CG) [NW06]. Any method would be equally appropriate because this is computed just once offline.

It is interesting to observe that in [Zho+03b], given a vertex $\mathbf{v}_i \in V$ as a *query*, the linear system $\mathcal{L}_\alpha \mathbf{x} = (1-\alpha)\mathbf{e}_i$ is considered instead, where $\mathbf{e}_i$ is the i-th canonical basis vector. A random walk iteration is applied, which (slowly) converges to $\mathbf{x}^*$, where each element x^*_j represents the "similarity" of vertex $\mathbf{v}_j$ to the query $\mathbf{v}_i$. In [Isc+17], the same linear system is rather solved directly and more efficiently with CG. One may then interpret (5.13) as follows. If we denote by $\mathbf{x}^*_i$ the solution to $\mathcal{L}_\alpha \mathbf{x}_i = (1-\alpha)\mathbf{e}_i$ for $i \in [n]$, then the solution of (5.13) is $\mathbf{u}^* = \sum_{i \in [n]} \mathbf{x}^*_i$. It follows that each element u^*_j measures the "expected similarity" of $\mathbf{v}_j$ to a query vertex of the graph for $j \in [n]$, averaged over all query vertices.

In fact, Katz centrality was introduced as such a global measure before being adapted by a *boundary condition* to measure relevance to individual vertices by Hubbell [Hub65]. This work has a long history before being rediscovered *e.g.* by [Pag+99; Zho+03b], as summarized in the study of *spectral ranking* [Vig09].

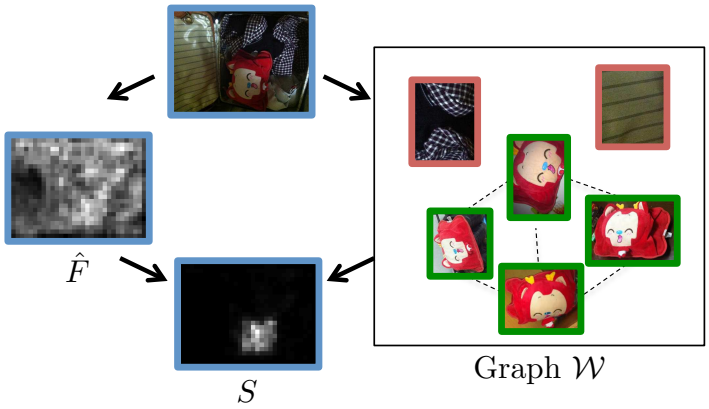


FIGURE 5.4 – Computing the *object saliency* map S of an image from Instre dataset (top), as defined in (5.14). For each patch, its neighbors in the graph (right) are found. Common patterns with high centrality in green outline, outliers with low centrality in red. S (bottom) then focuses on patches similar to common patterns and combines with feature saliency $\hat{F}$ (left).

5.3.8 Saliency map construction

Given the region descriptor set V, the region saliency vector $\mathbf{f}$ and the associated centrality vector $\mathbf{u}^*$ of an entire dataset, the problem is to construct a new saliency map $S \in \mathbb{R}^{h \times w}$ for an image in the dataset. The image is represented by its activation map $A \in \mathbb{R}^{h \times w \times k}$. Since this saliency is based on regions or patterns appearing frequently in the dataset, which are commonly associated to repeating objects, we call it *object saliency* (Object Saliency (OS)).

We compute S by a sliding window iteration over each position $p \in P$. The saliency value S_p at p is found as a linear combination of the centrality values of the nearest neighbors in V of a patch centered at p. In particular, we consider a square patch R_p of side a centered at p. We compute the vector $\mathbf{z}_p := w(m_A(R_p)) \in \mathbb{R}^d$ by max-pooling over R_p, whitening and normalizing as discussed in Subsection 5.3.5. If N_p is the set of indices of the k-NN of $\mathbf{z}_p$ in V, we compute S_p as

$$S_p := \hat{F}_p^\theta \sum_{i \in N_p} s(\mathbf{v}_i, \mathbf{z}_p) f_i^\theta u_i^*. \tag{5.14}$$

That is, each neighboring region descriptor $\mathbf{v}_i$ is weighted by its similarity to patch descriptor $\mathbf{z}_p$, its feature saliency f_i and its centrality u_i^*, while the entire sum is scaled by the feature saliency $\hat{F}_p$ at the current position p of the image being considered. Exponent θ controls the relative importance of feature saliency of the current image and neighbors compared to centrality. The object saliency computation is illustrated in Figure 5.4. Looking at the input image and is feature saliency map $\hat{F}$ alone, it is not evident which is the object of interest and which is clutter. This is only found by discovering other instances of the same object in the dataset, as represented by the graph.

5.3.9 Saliency-based representation

The object saliency map S highlights patterns that appear frequently in the dataset, with the background clutter removed. It is only natural then to apply the same method described in Subsection 5.3.4 to this map in order to detect a small number of regions per image. Unlike the regions detected from the feature saliency map $\hat{F}$, these new regions are more likely to appear in a new image. For the purpose of evaluation, we investigate both saliency maps.

For each region R detected from a saliency map ($\hat{F}$ or S) in a dataset image with activation map A, we apply GeM pooling and ℓ^2-normalization. All descriptors are then summed and the resulting descriptor is whitened with $w : \mathbb{R}^k \to \mathbb{R}^d$ as described in Subsection 5.3.5. We apply whitening on the aggregated vector and not separately per region. This follows the spirit of R-MAC [TSJ16], but performs GeM pooling instead of max, and the regions are detected in the saliency map rather than being uniformly distributed. This yields a global image representation in $\mathbb{R}^d$.

Pooling based on saliency is in fact the idea explored in CroW [KMO15], but here we follow the nonlinear two-level pooling of R-MAC (first within a region, then over regions) rather than the one-level sum of CroW. This is more powerful and has also been the basis of fine-tuning in [Gor+16b; RTC18].

5.3.10 Multi-scale representation

Multi-scale descriptor extraction with CNNs is becoming standard practice [Gor+16b; RTC18]. It consists of the following steps. Re-sample the image to multiple scales, feed each scale separately to the network, obtain an ℓ_2-normalized vector per scale, sum-pool over scales, re-normalize, whiten, and re-normalize. We follow the same principle, but use the representation of Subsection 5.3.9 per scale. Saliency maps are simply constructed independently per scale. Finally, pooling over scales is done with the generalized-mean as in [RTC18].

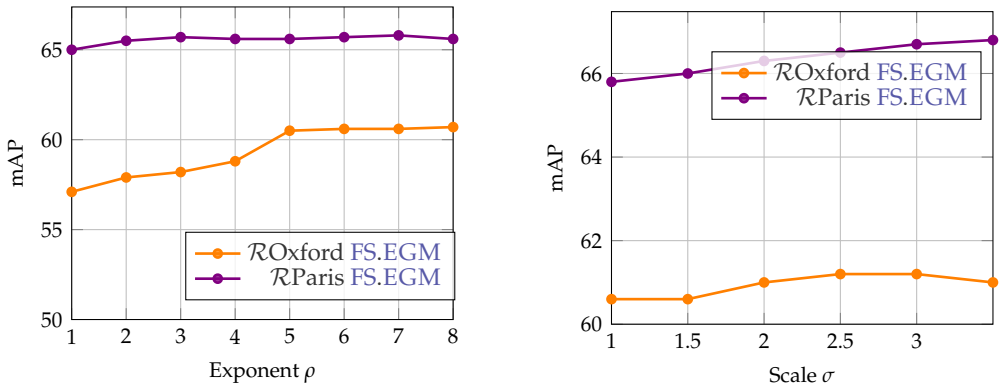
We additionally adopt the multi-scale concept for graph construction. Regions are detected independently per scale and the corresponding descriptors form new vertices.

5.4 Experiments

We apply the proposed representation on image retrieval. In particular, we have two variants of our method that both use the region detection described in Subsection 5.3.4. The saliency map which the detection is performed on is different in each case. FS.EGM uses the feature saliency map described in Subsection 5.3.3, and OS.EGM uses the object saliency map described in Subsection 5.3.4. The former is image specific, while the latter is both image and database specific.

5.4.1 Experimental setup

Test sets. We use three image retrieval benchmarks for our experiments. We evaluate on the revisited benchmark [Rad+18] of the Oxford Buildings [Phi+07] and Paris [Phi+08] datasets. We refer to the revisited datasets as $\mathcal{R}$Oxford and $\mathcal{R}$Paris respectively. We also use the more recently introduced Instre [WJ15] dataset. Instre contains around 27k images of small objects in cluttered scenes while objects appear with different variations, such as rotation, occlusion and scale changes, making it a challenging case. We use the evaluation protocol introduced in [Isc+17] for Instre. Search performance in all datasets is measured with mean average precision (mAP).



(a) Saliency exponent ρ (b) EGM scale parameter σ

FIGURE 5.5 – mAP on $\mathcal{R}$Oxford and $\mathcal{R}$Paris versus parameters in FS.EGM.

Image Representation. We use the global image representation as described in Subsection 5.3.9. This reduces image similarity to cosine, which is common practice [TSJ16]. Feature extraction is performed with the VGG network [SZ14] with GeM pooling that is fine-tuned for image retrieval [RTC18]. We use supervised whitening [RTC16; RTC18] as described in Subsection 5.3.5. Compared to global GeM pooling, our variants are different in that regions are detected from salient and repeating objects, while aggregation and whitening is identical. Detection is applied to dataset images only, while we use the provided bounding boxes on the query side.

Implementation Details. To simplify region detection, each saliency map is masked above threshold τ and element-wise raised to exponent ρ before detection, which removes the weakest regions and increases the contrast between foreground and background objects. We fix threshold $\tau = 0.01$ in order to remove noise from saliency maps. We set exponent $\rho = 1$ and scale parameter $\sigma = 1$ before any parameter tuning is performed. We determine OS parameter θ in (5.14) by visual inspection of OS and set $\theta = 3$ throughout our experiments. We perform our experiments on a 16-core Intel Xeon 2.00GHz CPU. It takes 36s to create the graph on Instre, while centrality computation takes negligible amount of time. Saliency computation and detection per image takes 0.02s for FS and 0.23s for OS.

5.4.2 Parameter tuning

In this section, we show the impact of FS.EGM and OS.EGM detection parameters on the retrieval performance. We tune the parameters on $\mathcal{R}$Oxford and $\mathcal{R}$Paris, while showing that their impact is similar on both datasets.

Feature saliency detection is evaluated first by FS.EGM, while we do not compute object saliency and OS.EGM yet. Figure 5.5a shows the effect of ρ on FS, which controls the contrast of the the saliency map. We observe that large ρ is needed to remove as much clutter as possible from the noisy FS activations. We set $\rho = 7$ for the rest of our experiments. Scale σ is used during EGM sampling as explained in Subsection 5.3.4. Its impact in performance on FS is shown in Figure 5.5b. Setting $\sigma = 2.5$ results in good performance and regions that are large enough for FS.EGM.

Centrality measure We use the different centrality measurements presented in Subsection 5.3.7, compute the OS map, and evaluate the quality of saliency maps. We use Instre's bounding

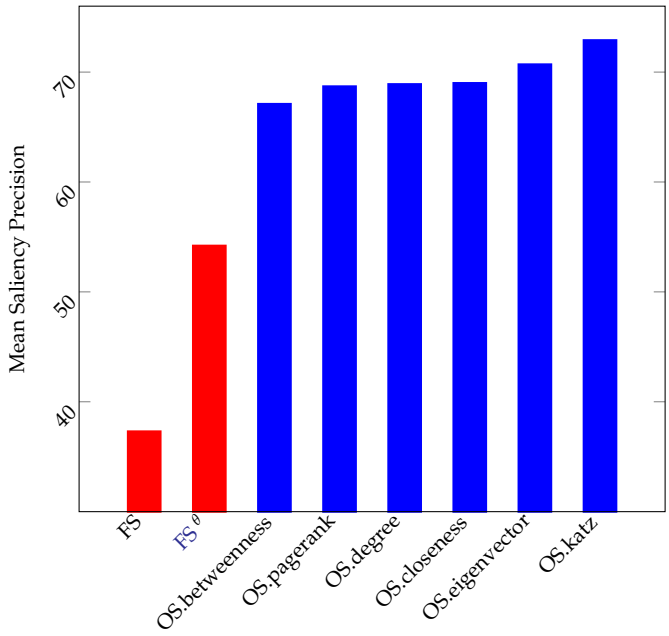


FIGURE 5.6 – Saliency precision evaluation of FS and OS created with several centralities maps measured on all images of Instre.

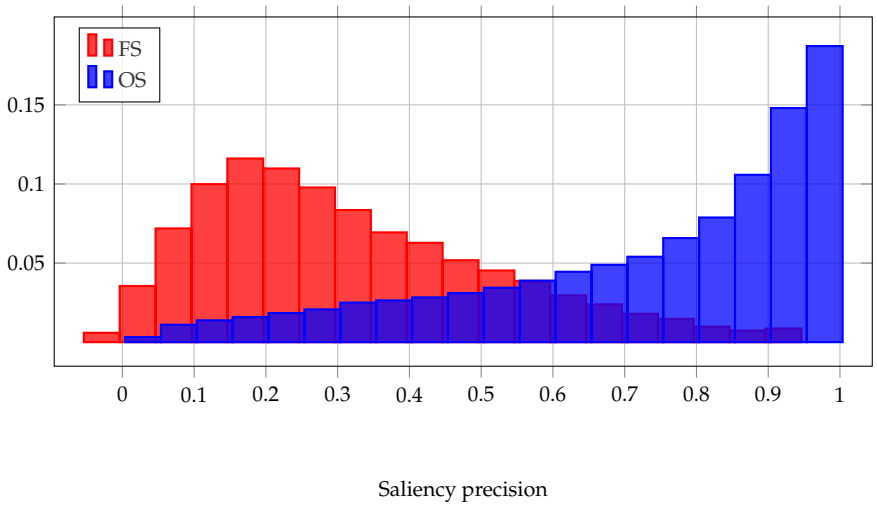
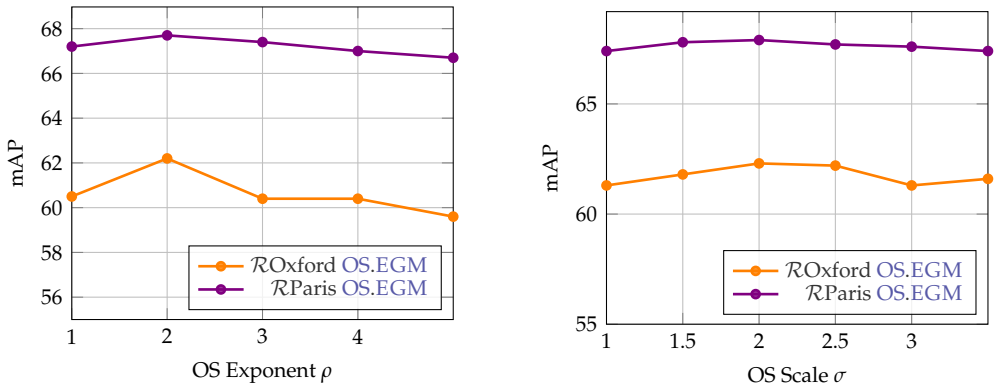


FIGURE 5.7 – Histogram of saliency precision for FS and OS maps measured on all images of Instre.



(a) Saliency exponent ρ (b) EGM scale parameter σ

FIGURE 5.8 – mAP on $\mathcal{R}$Oxford and $\mathcal{R}$Paris versus parameters in OS.EGM.

box annotation as ground truth and measure *saliency precision* as the sum of saliency map values inside the bounding box normalized by the total sum over the entire image. High precision means that a saliency map captures the repeating object and discards the background.

We present the average saliency precision over the entire Instre dataset in Figure 5.6. The comparison includes saliency maps created with FS and FS $^{\theta}$, since the latter is used in (5.14). Improvements brought by OS compared to FS are significant regardless of the centrality measurement, while Katz centrality gives the best precision. Figure 5.7 shows the distribution of precision over all Instre images for FS and OS. We use Katz centrality measurement for the rest of our experiments and simply refer to it as OS.

Object saliency detection is then evaluated for retrieval. Now, the feature saliency parameters are fixed, Katz centrality is selected and EGM detection is applied on the new saliency map. We observe that OS behaves quite differently compared to FS, because foreground objects are much cleaner. The impact of parameters σ and ρ is shown in Figure 5.8a and Figure 5.8b respectively. It is remarkable that a much lower exponent is needed in this case. We choose $\rho = 2$ and $\sigma = 2.5$.

Visual examples for saliency maps and detections for FS.EGM and OS.EGM are shown in Figure 5.9. In all cases, OS is cleaner and focuses on objects that FS cannot discriminate.

5.4.3 Comparison to other methods

We compare our method against GeM descriptor [RTC18]. We additionally evaluate the multi-scale variants for GeM, FS.EGM and OS.EGM, as described in Subsection 5.3.10. All methods are tested with k-NN search and global diffusion [Isc+17], which is a method for QE and is known to significantly improve performance. Results are given in Table 5.1. We report results for both Medium and Hard settings on $\mathcal{R}$Oxford and $\mathcal{R}$Paris.

FS.EGM does not always improve the performance, since objects captured are not necessarily relevant for the particular dataset. This is what OS.EGM captures and boosts the search performance, especially on Instre. We outperform GeM pooling in all datasets and scenarios, except for $\mathcal{R}$Paris- Hard with query expansion. Note that we use the default diffusion parameters which are tuned on GeM-like descriptors. OS.EGM provides larger

FIGURE 5.9 – Examples of images from $\mathcal{R}$Oxford (first 2 rows) and Instre (last 3 rows) datasets, along with smoothed FS and OS maps superimposed on the images and regions detected by EGM, in red.

		Medium		Hard	
Method	Instre	$\mathcal{R}$Oxford	$\mathcal{R}$Paris	$\mathcal{R}$Oxford	$\mathcal{R}$Paris
Single Scale					
GeM [RTC18]	54.2	60.8	67.0	33.3	42.1
FS.EGM	54.6	61.2	66.5	33.4	41.7
OS.EGM	58.3	62.2	67.7	34.9	43.4
Single Scale + QE					
GeM [RTC18]	73.3	69.3	83.9	42.3	71.8
FS.EGM	72.8	71.0	84.1	42.3	71.4
OS.EGM	76.5	**72.2**	83.8	**46.5**	69.9
Multi Scale					
GeM [RTC18]	57.0	62.0	69.3	33.7	44.3
FS.EGM	57.7	63.0	68.7	34.5	43.9
OS.EGM	61.3	64.2	69.9	35.9	46.1
Multi Scale + QE					
GeM [RTC18]	75.0	69.3	83.9	41.1	**73.9**
FS.EGM	74.6	71.0	84.1	40.6	72.5
OS.EGM	**77.4**	69.0	**85.4**	41.9	72.3

TABLE 5.1 – mAP comparison of our method against baselines on all tested datasets. QE refers to query expansion by diffusion [Isc+17].

improvements on Instre, which is more challenging due to small objects and severe background clutter. This is exactly where our detection is essential.

There are several other approaches that deal with region detection or saliency masks. Their results are not directly comparable, because they are not evaluated on $\mathcal{R}$Oxford, but Oxford5k. Therefore, they are not included in Table Table 5.1. Nevertheless, we outperform their reported results. We achieve 86.6 on Oxford5k with OS.EGM and no query expansion. Salvador *et al*. [Sal+16] use the off-the-shelf VGG and fine-tune RPN on the test set. Without using query expansion, they obtain 71.0 on Oxford5k. Similarly, Jimenez *et al*. [JAG17] learn class weights and apply them on the activation maps of off-the-shelf VGG and achieve 73.6 on Oxford5k. Song *et al*. [Son+17] train on different datasets, and achieve 78.3 on Oxford5k. The results obtained by learning a saliency mask in [Noh+16] are not comparable since spatial verification with local features is always applied in the end. Zheng *et al*. [Zhe+16] achieve 83.4 with regional representation on Oxford5k. They employ both CNN and local features, while we only rely on CNN and much more compact representation. Finally, according to our knowledge, [Moh+17] is the only work that evaluates on Instre, which is rather challenging due to small objects. However, their results are not directly comparable as they use a different CNN model (ResNet101).

Region cross-matching methods [Raz+16] represent an image with multiple vectors, sacrificing memory footprint and complexity for accuracy. In particular, the memory is linear in the number of regions, while the complexity is quadratic. We compare our global representation with region cross-matching (R-Match) and regional diffusion [Isc+17] in Figure 5.10. We sample regions uniformly at 3 scales as in R-MAC [TSJ16] and apply GeM pooling separately for each region. Different numbers of regions are obtained by GMM reduction, exactly as in [Isc+17].

Compared to regional descriptors, we require about 4 times less memory to achieve the same performance. The runtime complexity gain is in the order of 4^2, which holds for the case of R-Match and also for the first part of diffusion where Euclidean nearest neighbors

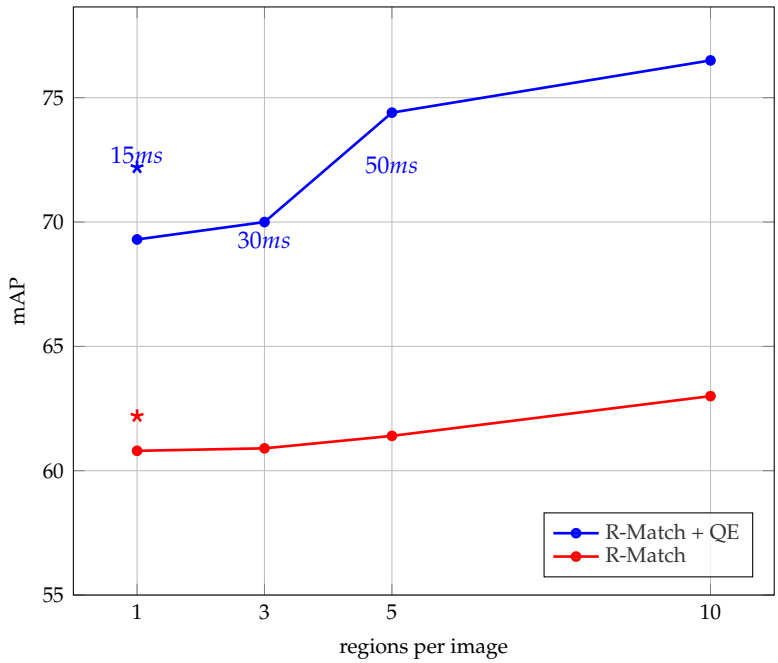


FIGURE 5.10 – mAP comparison of our global OS.EGM ($\star$) to R-Match with uniformly sampled regional descriptors, with and without diffusion on $\mathcal{R}$Oxford (Medium setup). Text labels refer to query time.

are found. The diffusion complexity is O(m), where m is the number of non-zero entries of the graph. We found that m is 3.7 times smaller in our case and our measurements of actual query timings agree with this ratio.

5.5 Discussion

We propose a region detection approach that is dataset specific but requires no supervision. It captures not only salient objects by considering each image individually but also frequently appearing ones by considering the dataset as a whole. As a result, we avoid separate indexing of regional descriptors and construct a global descriptor by pooling over data-dependent regions, which performs well under background clutter and severe occlusions. We demonstrate that this approach is effective in particular object retrieval where background clutter is a common problem.

Chapter 6

Conclusions

Contents

In this chapter we discuss the work presented in this thesis. We first start by drawing global conclusions and summarize the knowledge that can be extracted from the different presented efforts. In particular, we discuss how CNNs pre-trained on classification or manifold learning allow the performance of different tasks. Additionally we propose perspectives for those works, from fine-grained improvements to general ideas. This conclusion is also a chance to discuss unpublished works.

Section 6.1 describes lessons learned from Chapter 3, 4 and 5. We present incomplete works in Section 6.2. And finally, we propose perspectives in Section 6.3.

6.1 Discussion

Works presented in this document focus on different applications, and therefore methods are specialized to task requirements. However, knowledge accumulated from those efforts is more general than any particular application. Indeed, it is possible to induce global ideas that can be generally used in computer vision. This thesis has been focusing on two aspects: *extracting as much information as possible* and *using the minimum of supervision*.

We have seen that CNNs designed for classification, or manifold learning, learn robust representation that can be exploited in diverse ways. Indeed, feature maps can be used to extract both classification and localization information even when the network was trained without spatial cues.

We also discussed the difficulty to provide enough supervision to efficiently train deep networks and we proposed a solution to minimize supervision. On the one hand, it is possible to directly reduce the amount of labeled data and achieve comparable results as with

a full supervision scenario. On the other hand, it is possible to focus on available learned models pre-trained on a task before transferring knowledge to another dissociated task.

6.1.1 Exploiting data

We have seen that state-of-the-art CNN methods, commonly trained in a supervised manner, need large amounts of labeled data. Unlabeled data can be added directly to the training of CNNs. Unsupervised training allows models to be trained on images not labeled using for instance clustering methods [CN12]. Performing transfer learning with unsupervised learned weights proved to outperform random initialization [Car+18a]. This pre-training can be relevant both on large datasets and on smaller ones as we show in Chapter 3. Unlabeled data can also be integrated in fully-supervised training by predicting pseudo-labels and using adapted losses [Isc+19b].

We have shown that active learning can benefit largely from information extractable from unlabeled images (Chapter 3). Adding unsupervised and semi-supervised methods improved results significantly compared to results obtained with acquisition functions alone. When constructing a model for a particular task, we believe that unlabeled data should be integrated as much as possible. We showed they contain much information that can be used to construct better representations or to perform training. Although we applied those methods to active learning, impressive gains due to the using of unlabeled data must be applicable to other tasks.

Unlabeled data can also be exploited to extract statistics from a dataset. We have shown that graph-based methods are very relevant and in particular in the context of retrieval. Indeed when the same instances of an object are searched throughout a dataset, it is possible to use the repetition information using a centrality measure and to detect nodes that are popular in a graph (Chapter 5). We were able to produce instance mining having only a pre-trained model on manifold learning. The dataset is then considered as a whole, which is particularly interesting when it is not too large. We also have seen that graphs allow to perform more unsupervised operation as similarity propagation [Isc+18b] (Chapter 4, 5) or labels propagation [Isc+19b] (Chapter 3).

We were also able to perform weakly-supervised localization by exploiting pre-trained networks on "simple" supervision tasks, such as classification or manifold learning. We additionally proposed to integrate spatial information in training using only positive and negative pair labels. We showed that with or without any additional training, it was possible to detect zones of interest in feature maps, either objects (Chapter 5) or local features (Chapter 4). This is discussed further in the following section.

6.1.2 Exploiting the feature maps of CNNs

Throughout this thesis, we have been using CNNs although performing different tasks. We showed that a backbone network can be used in many different ways, leading to diverse robust representation. This makes CNNs a powerful model. The set of feature maps–outputs of the backbone–can be fed to fully-connected layers (Chapter 3) that generate classification predictions, or can be pooled globally (Chapter 4, 5) or locally (Chapter 4) as shown in Figure 1.3.

We discussed convolutional representations and showed that their properties allow the extraction of accurate localization information. Deep feature maps are sparse [KMO16]; the

sparsity can be used to weight feature maps or select features maps non-sparse to extract local features (Chapter 4). Feature maps are also semantic [Zho+16]. We were able to aggregate the activations of feature maps into a feature saliency map that focuses on objects without additional information. We also showed that it was possible to integrate additional information in the construction of saliency maps focusing on objects of interest in Chapter 5.

Alternatively we have shown that feature maps contain local information that can be used to establish repeatable correspondences from one image to another (Chapter 4). We were able to extract robust local features by using a detector. We used MSER [Mat+02] but another detector could be used (feature maps are more simple than images). We showed that the invariance of feature maps to variations such as light changes, some rotation and scale variations make them very relevant in the context of instance-level tasks. In particular, local features detected within the feature maps were robust enough to perform spatial matching without descriptors. Interestingly, we also showed that local features could be detected from feature maps learned both on manifold learning–trained on datasets close to the target datasets–but also on the classification dataset Imagenet [Don+09]. In both cases, spatial information is repeatable and can be exploited to perform spatial matching.

6.2 Unpublished efforts

With more than 3000 submissions to every important conference, computer vision is currently a tense field. Ideas must not only be supported by many experiments and significant improvement, they must also be developed fast. While ideas spread, speed is a key factor. This section introduces ideas we worked on but were not completed or published.

6.2.1 Deformable representation

We have seen that properties of feature maps can be used to solve weakly-supervised localization problems. The extracted spatial information relies on the quality of feature maps and one may wonder how they can be improved. We proposed in Chapter 4 to force the network to match parts of images, but only achieved mixed results and at a high computational cost.

The first project we worked on was based on the Region-based Fully Convolutional Network (R-FCN) [Dai+16] object detection method. This fully convolutional network learns position-sensitive score maps. Object bounding boxes are divided into parts represented by cells of a grid and each feature map is trained to specifically recognize a certain cell (object part) of a certain class. However, we know that objects are deformable and that parts cannot be expected to always be at the same position. We tried to improve the training by allowing object parts to move by some learned offsets from their original positions. Our idea was that if this deformation was beneficial for an object detector, it could also be beneficial for convolutional kernels.

In particular, convolutional kernels are rigid grids, and their receptive fields are fixed. Intuitively, the relevant information to a kernel may be slightly out of the kernel scope. Kernels should also be allowed some flexibility and we investigated this idea. Dai *et al.* [Dai+17] concurrently proposed exactly to add flexibility to the R-FCN and convolutional kernels and they published their results while we were experimenting. Although doing something very similar, they had the additional idea of predicting offsets rather than having them as learned parameters. They showed that the receptive field of a pixel in the last

features maps were more localized around the objects of interest than with the standard convolution. This work was further improved by Zhu *et al*. [Zhu+18a].

6.2.2 Parametric convolutions

The performance of deformable convolution, as discussed above, opened the question of how convolutions are performed. This simple operation could be modified or revisited. We attempted to propose a new convolutional structure and in particular to use GMM. The idea was to fit a set of components to feature maps using EGM [AK12]. This method has shown to be very relevant for detecting candidate objects on feature maps, automatically finding the right number of components (Chapter 5). Starting from this idea, we have imagined integrating the detector as a layer in the neural network, rather than an offline method. Both feature maps and convolution kernels are represented parametrically as sets of Gaussian components. An intermediate convolutional layer inputs and outputs components and computes convolutions directly in closed form as a sum. Additionally, in the last layer, the detection of Gaussian components yields an object detection which integrates component selection–avoiding using non-differentiable NMS. We have implemented this model and performed several experiments, but we have encountered computation issues, both in terms of speed and memory. The number of components to treat using EGM was too high.

6.2.3 Spatial dropout

Another way to improve feature maps quality could be to use a regularization scheme; we tried to improve them in the context of weakly-supervised object detection. We first observed that CAMs [Zho+16] were focusing only on the most discriminative part of an object: heads for human and all animals, wheels for cars, etc. Even if this is good enough for classification, if we want to use the feature maps for weakly supervised detection, then all parts of the objects should be found in the CAMs. The regularization abilities of CNNs can be improved by using ideas such as dropout [Sri+14]. Dropout had been successfully applied on fully-connected layers. However, convolutional dropout is not straightforward. On the one hand, dropping a single pixel has very little effect as neighboring pixels still participate to the convolution operation; on the other hand dropping an entire feature map [Tom+15] would discard too much information. We experimented with an intermediate solution: we randomly removed information from the most activated *regions* in the feature maps. With this simple idea, we could see on a small dataset that the activated regions were indeed evolving during the training. Some initial experiments have also shown improvement on classification. Based on a similar idea, Ghiasi *et al*. [GLL18] concurrently proposed to drop convolutional units of contiguous regions in feature maps and showed both improvements in terms of accuracy and stability to hyperparameter choices.

6.3 Perspectives

6.3.1 Exploration

We have discussed several ways to extract as much information as possible with as little supervision as possible. Our different works provided some initial solutions but could be further explored and used in other contexts.

Active learning or not? Active learning is a task constructed to help creating a new dataset at low labeling costs. We have discussed in Chapter 5 that the performance of active learning

in the context of deep learning are disputable. In particular, we have shown that the labeling budget has a great impact on the utility of active learning: random sampling is better with a low budget (10 images per class). Some remaining questions are: given a new unlabeled dataset, how to define the best label budget? When does active learning actually help? We have shown that it is beneficial to use an acquisition function (over random sampling) when the image representation is good enough. Therefore, one solution may be to start by randomly selecting a large enough budget of images, train the model and then use an acquisition function on smaller budgets in the following cycles. Indeed, an active learning scenario could include different acquisition functions, with different budgets, at different cycles. In general an active learning scenario, which is currently fixed to a constant label budget per cycle, could be given more freedom in order to discover when active learning can actually help. Predicting the required budget would be an interesting direction in this respect.

Active metric learning. Active learning has been investigated mostly in the context of class-level tasks. We have discussed throughout this thesis the possibility to improve different tasks using similar processes. One line of work could be to perform active metric learning. In particular, it could be applied to select image pairs–or positive and negative images for a given anchor–to label for fine-grained classification [Ber+14] which is often performed following metric learning methods [Wan+14b; Cui+15].

Locally verified classification. We have seen that local features can be extracted from networks pre-trained for classification on Imagenet (Chapter 4). Local features detected using DSM are semantically meaningful and could maybe be used in the context of fine-grained image classification. This task can be performed by using a pipeline close to the one applied in image retrieval [Gor+16a; RTC18] and spatial matching again would be a refinement step. We have shown that local features detected in Chapter 4 focus on different parts of an objects, which would suit fine-grained classification.

Object discovery spatially verified. We have shown in Chapter 5 that it is possible to discover objects in images by using an unlabeled dataset. We applied object discovery to image retrieval which aims at finding images depicting the same instance of an object. One direction could be to improve GOD in order to perform unsupervised instance-level *object detection* and *segmentation*, maybe by integrating label information in the graph. Moreover, local features detected on the same feature maps as GOD, using DSM, could help improve object detection by performing spatial matching (Chapter 4). Objects of the same instance can be aligned, as performed in fine-grained classification and discussed above.

Class-level object discovery. Our instance-level methods rely on the robustness of the representation to image variations. If such representations were robust to intra-class variation, *class-level object* discovery could work in a similar manner. We showed positive results of DSM on feature maps trained on Imagenet [Don+09]. Application to GOD should also be possible. We would be able to first construct a coarse saliency map, which would discard background elements; feature maps trained for classification are activated mostly at the object position, as shown by [Zho+16]. Moreover, centrality could still be applicable as it depends on the similarity between descriptors, which in this case would have been trained to minimize distances between objects of the same class. Alternatively, we could add some supervision and augment the object saliency to also predict a class; this could be performed by adding labels to the graph, maybe using a label propagation method [Isc+19b]. Our

technique would only require class-level labels for the training of global descriptors, discriminative in the context of classification and the label propagation in the graph.

Learning to mine. The final object saliency maps of our GOD (Chapter 5) are localizing objects with a good accuracy (as shown in Figure 5.6). They are computed without supervision which makes them very relevant when dealing with new datasets. Such saliency maps could serve as supervision while training a CNN to *discover objects*. We are aware that the saliency maps constructed without supervision are not perfect. However, results obtained with semi-supervised training [Isc+19b] (Chapter 3) showed that training a model on noisy labels can achieve very good results. Moreover, GOD could also be improved by including labels in the graph, either on all images or some (semi-supervision). The final trained network would generate a saliency map at the small cost of a feed-forward pass; it could replace our graph-based method at test time. Alternatively, we could perform weakly-supervised object detection by training a RPN or Teichmann *et al*. [Tei+19] pipeline with the bounding boxes generated by GOD. The object saliency maps could also directly serve as supervision for segmentation, however performing instance segmentation, which require separating the objects of a same class, would necessitate more improvements.

Small objects. In our work presented in Chapter 4, we experimented on city datasets. Those datasets include images depicting buildings and statues. Such structural elements often represent a large portion of an image and are invariant in terms of low-level information, as shape or texture. They can therefore easily be described by local features that highlight detailed parts of the objects. DSM can extract up to hundreds of features if the building contains many details (see Figure 4.1). This is not the case of images depicting little objects *e.g*. Instre [WJ15]. One question remains: how to adapt the detection of local features in the case of small objects? A first obvious solution would be to improve our multi-scale approach. Alternatively, regions of interest could be detected using GOD before going back to the image space. We could increase the resolution of the selected regions before detecting features using DSM.

Spatially verified diffusion. The diffusion of similarity in a manifold gives impressive results in image retrieval. It allows the discovery of images semantically close rather than in the Euclidian space. We have shown that the quality of a diffusion process depends on the query images used to start the walk (Chapter 4). In particular, DSM helped by providing reliable images similar to the query. Additionally, spatial matching could be used while diffusing similarity and give a hard decision–are two images matching?–signaling where to stop diffusing. Following a similar idea, Chang *et al*. [Cha+19] incorporated spatial verification to k-NN by re-weighting edges of the graph. Alternatively, one could use the transformation computed between images to *follow the objects* and discover when they get out of sight. Diffusion could also be performed using local features provided by DSM rather than global descriptors, providing refined similarity diffusion. Additionally, one could investigate how the diffusion process could be integrated to manifold learning, in particular by providing an unsupervised training set.

Local feature detection. DSM can be used in image retrieval to perform spatial matching (Chapter 4) but could also be used in other tasks *e.g*. optical flow [Dos+15b] or local descriptors for patches [TFW17]. Most methods are trying to detect features from a single saliency map [Noh+17], which is less sparse than the feature maps. On the contrary, we exploit the sparsity property of feature maps to detect well defined features (Chapter 4). Dusmanu *et al*. [Dus+19] and Cieslewski *et al*. [CBS19] also proposed to detect features within feature maps;

they used detected features on patches and showed good results. However, all methods discovering features on the feature space are bound to the low resolutions of feature maps. One direction to follow could be to construct or learn a projection from feature space to image space.

Whitening. Whitening is currently added as a post-processing step on global descriptors. It proved to significantly enhance the descriptor quality [JC12] (Chapter 4). We have seen that feature maps suffer from repetition; we removed repeating activations using NMS in Chapter 4. Whitening applied locally on the feature maps could help in that regard. It should be adapted to the properties of feature maps, which are sparse and non negative as opposed to descriptors.

6.3.2 General perspectives

A decade ago, CNNs were hardly used in computer vision and most efforts were being focused on SVMs. While this thesis is centered around CNN, past directions achieving good results should not be forgotten. I started working on computer vision problems in 2016, exactly when CNNs were becoming state-of-the-art in all tasks. I could have easily focused on creating new layers, new architectures, to the exclusion of all else. I consider myself fortunate to have had the chance to mix within this PhD CNNs with pre-existing ideas. I used graphs, performed diverse operations on them, detected local features, performed spatial matching. All those techniques have been deeply studied and well defined by experts. One perspective would be to push the integration of methods developed before and after CNNs achieved state-of-the-art results. In particular, neural networks are an instance of differential programming. They are defined as a set of differentiable operations and learned to approximate desired functions. The design of neural networks can be performed by *brute force*, assembling all operations in a diverse way and selecting the best combinations. However, one direction could be to get inspiration from previous work and in particular to encode past expertise in current models, with differential programming.

Moreover, to design a network by brute-force also has a high cost in terms of energy consumption. Once a model is carefully designed, meta-parameters must be determined. Again, an expensive solution to develop a model is often to perform brute-force search. Strubell *et al*. [SGM19] discussed the cost of developing a neural model in the context of Natural Language Processing (NLP) both in terms of finances and energy. In particular, their first table discuss CO_2 consumption. They estimated that developing a neural network for NLP–including the tuning of meta-parameters–costs over 626,155 lbs in terms of CO_2 emissions *vs*. 36,156 lbs for one year of consumption of an average American. The authors encouraged the community to take these considerations into account while doing research. In order to avoid such expensive efforts, a solution could be to use AutoML to predict the design of architectures and their best parameters. AutoML is currently an active field [Guy+15; He+18; Won+18; Feu+18]. It promises to generate small models that are less expensive to train and use at test time. AutoML is still expensive in terms of computation; however, different solutions have been proposed to reduce the cost [Won+18] and hopefully it will become more efficient. We should consider our responsibility towards society and we cannot always ignore the energy we use in research, even if we are lucky enough to be provided with abundant computational resources.

Appendices

Résumé Étendu

La vision par ordinateur est un domaine de recherche qui propose des solutions permettant à un ordinateur d'interpréter des images. Bien qu'étudié depuis plus de cinquante ans, le domaine a récemment gagné en popularité grâce à d'excellents résultats qui ont permis la création de nouvelles technologies, notamment les voitures intelligentes.

Le contexte

En mai 2017, le journal "The Economist" publiait un article intitulé "La ressource la plus précieuse au monde n'est plus le pétrole, mais la donnée" [17]. L'article décrit la transition effectuée par l'industrie depuis le début des années 2000. Les entreprises font actuellement de très larges profits en exploitant les données des utilisateurs, données qui sont souvent fournies gratuitement. En tant qu'utilisateurs, nous profitons aussi des données amoncelées dans les serveurs. Nous sommes encerclés par des technologies contenant des appareils photo tels que les téléphones, voitures ou drones *etc*. Ces machines capturent des images et vidéos de notre monde et peuvent nous aider à améliorer notre vie quotidienne à condition qu'elles comprennent ces images. Qu'est ce qu'une image représente ? Quels sont les objets d'intérêt ? Que se passe-t-il dans la scène ? Les réponses à ces questions doivent être extraites directement à partir des pixels d'une image sans aucune autre information.

Le domaine de la *vision par ordinateur* étudie une large variété de problèmes. Il comprend des tâches qui se concentrent soit sur des images 2D ou 3D soit sur des vidéos. La vision par ordinateur propose des solutions pour effectuer de la reconnaissance, de l'apprentissage, de la compression, de la génération ou de l'exploitation d'image. Les images peuvent aussi être associées à d'autres modalités, telles que du texte, du son ou des méta-informations. Ce domaine a été étudié depuis des décennies et atteint aujourd'hui des résultats qui touchent un large public. Il a ainsi récemment gagné en popularité et attire à la fois des chercheurs et des industriels. La conférence CVPR de 2019 a regroupé plus de 9000 visiteurs, 4000 de plus qu'en 2017.

Cette thèse se concentre sur la compréhension des images. Comme souvent en apprentissage statistique, les premiers travaux se sont inspirés de la biologie et en particulier des neurosciences. ROSENBLATT [Ros58] se demandait comment les systèmes biologiques reçoivent, encodent et enregistrent l'information à partir du monde physique. Il a proposé un "système nerveux hypothétique", appelé le *perceptron*, qui a été présenté comme le début de l'intelligence artificielle. Relayé par le New York Times dans un article publié en 1958 et intitulé "Un équipement de la marine apprend en agissant" [58], la machine a reçu beaucoup d'attention.

Cependant, les promesses d'une machine qui serait "capable de lire et écrire [...] devant être fini en un an et à un coût de $100,000" ne furent pas tenues. Quelques années plus tard, le travail fût critiqué par des collègues [MP88]. En conséquence, les réseaux de neurones convolutifs ont perdu de l'intérêt, précédant un premier "hiver" de l'intelligence artificielle.

Il aura fallu plus de cinquante ans de recherche pour que les modèles neuronaux atteignent des résultats intéressants. Ces efforts comprennent le développement de la "back-propagation" [RHW86], des réseaux de neurones convolutifs (CNNs) [KSH12 ; SZ14 ; He+16], les progrès du matériel informatique (*e.g.* GPU) et la collection de très grandes bases de données [Kri09 ; Don+09]. Un succès notable fut celui du modèle proposé par KRIZHEVSKY et al. [KSH12] en 2012, qui a prouvé que les CNNs permettent d'atteindre de meilleurs résultats que ceux obtenus en combinant des descripteurs locaux et des classifieurs linéaires. Enthousiasmée par ces résultats, la communauté de vision par ordinateur s'est largement orientée vers l'apprentissage profond, ce qui a conduit au développement de modèles profonds très performants [He+16 ; Hua+17] qui contiennent des millions de paramètres à apprendre.

Les problèmes du monde réel peuvent être traduits par des tâches en vision par ordinateur. Elles sont définies par un jeu de données d'entrainement et un jeu de test, des données de vérité terrain et des fonctions de coût. Le domaine s'intéresse à de très nombreux problèmes. En particulier, les tâches de compréhension d'image sont au centre de beaucoup de recherches. Elles se concentrent sur des informations de classe ou d'instance des objets dans une image. La *classification d'image* consiste à découvrir quelles sont les classes des objets présents dans une image. La *détection d'objets* nécessite de localiser les objets classés dans des boîtes englobantes. La *segmentation d'image* consiste à classer chaque pixel d'une image. La *recherche d'images* suppose de trouver toutes les images contenues dans une base de données qui contiennent un objet en particulier, dépeint dans une image exemple donnée en requête.

Les méthodes basées sur des CNNs sont à l'état de l'art de nombreuses tâches de la vision par ordinateur, comprenant toutes celles listées ci-dessus, mais aussi des tâches de traitement de texte, de question/réponses et d'audio. Les entrées et sorties d'un modèle varient selon la tâche. Cependant, les solutions basées sur des CNNs partagent souvent une partie de la structure du réseau. En effet, les réseaux profonds contiennent un premier empilement de couches de convolution (le réseau "*backbone*") suivis de couches spécialisées. Les réseaux sont généralement entraînés de façon supervisée, et les labels et fonctions de coût sont construits ensemble de façon à former un réseau qui apprend les bonnes informations.

Apprendre à voir

Les qualités de la représentation d'une image nécessaires au traitement dépendent directement de la tâche de vision par ordinateur. En effet, d'une tâche à l'autre, les propriétés les plus importantes d'une image varient. La conception de cette représentation, soit construite, soit apprise, est un problème central dans le domaine de la vision par ordinateur.

De plus, cette représentation doit être robuste aux variations dans une image. Tout d'abord, les objets dépeints par plusieurs images peuvent varier. Ces variations peuvent être dûes à un changement d'angle de la caméra, à des effets de zoom, des changements de lumière, des variations jour/nuit. Les objets peuvent aussi être partiellement représentés dans l'image à cause d'obstructions faites par d'autres objets ou le cadrage de la caméra. D'autre part, les objets d'une même classe peuvent connaitre de très grandes variations "intra-classe". Ils

peuvent être différents en couleur, forme, texture jusqu'à avoir une apparence entièrement différente. Selon le problème, une représentation ne doit appréhender que certaine types de variations et déformations : les plus importantes pour la tâche.

Les premières techniques mises en place étaient basées sur des descripteurs locaux [Low99 ; MS04] calculés en utilisant des opérations mathématiques directement sur l'image. Ces descripteurs ne dépendent pas des tâches pour lesquels ils sont utilisés. Ils se sont avérés très efficaces pour effectuer des opérations au niveau de l'instance des objets, car ils permettent d'acquérir des informations locales et fines. Ces descripteurs locaux peuvent aussi être agrégés en une représentation globale *e.g.* BoW [SZ03]. Ils peuvent être utilisés par des classifieurs, entrainables ou non, tels que les algorithmes des k plus proches voisins [CH67] ou les SVMs [BGV92]. Cependant, l'utilisation de ces descripteurs est limitée à cause de leur définition fixe.

Les CNNs offrent la possibilité de spécialiser la représentation d'une image à un certain problème en l'apprenant directement pour la tâche désirée. En effet, différentes architectures de CNN et fonctions de coût, telles que les fonctions de régression et de classification, permettent d'extraire différentes propriétés à partir d'une image. Pour chaque tâche, le type de supervision est défini, de même que les labels, les fonctions d'apprentissage et d'optimisation qui permettent de maximiser la performance d'un modèle pour une tâche.

Représentation. Une image peut être décrite par un ensemble de *caractéristiques locales* ou une *représentation globale*, et, l'un et l'autre doivent décrire l'information insensible aux variations précédemment décrites. Les caractéristiques locales, *e.g.* "SIFT" [Low99], détaillent les spécificités d'un objet. Elles peuvent aider dans le cas où des parties d'un objet manquent dans une image, à cause d'une obstruction du champ ou de la présence de multiples objets. Elles peuvent en effet faciliter une classification *partielle* ou la mise en correspondance spatiale. Cependant, stocker tous ces descripteurs a un coût important, et peut être irréalisable quand des milliards d'images doivent être traitées. Les représentations globales sont plus grossières et contiennent moins de détail qu'un jeu de caractéristiques locales. Elles sont aussi plus difficiles à construire : une bonne représentation globale nécessite souvent un apprentissage. Cependant, cette représentation a l'avantage d'être facile à stocker.

Les CNNs sont invariants en translation et, selon les données utilisées lors de l'apprentissage, peuvent être invariants à de petites rotations de l'image ainsi qu'à des changements de lumière. Ils peuvent aussi être explicitement conçus pour être invariants à plus de transformations, comme la rotation [Jad+15 ; Est+18], la réflexion [CW16] ou le changement d'échelle [SM13 ; KSJ14] en utilisant la technique d'augmentation des données. Les représentations globales obtenues avec un CNN, sont compactes et discriminantes. Elles permettent le stockage de larges bases de données dans un espace mémoire limité tout en permettant le calcul d'opérations sur le jeu de données entier.

Labels et fonctions de coût. La plupart des méthodes d'apprentissage statistique appliquées à la vision par ordinateur sont entrainées de façon supervisée. Les réseaux de neurones convolutifs nécessitent de très grandes quantités de données annotées. La définition d'un label dépend de la tâche, par exemple : un scalaire représente la classe d'un objet pour la classification, alors qu'un couple boîte englobante / classe est utilisé pour la détection d'objet, un label par pixel pour la segmentation d'image et des pairs positive/négative d'images pour la recherche d'image. La communauté de vision par ordinateur a souvent

fait appel au service d'AMT pour annoter des images et produire de nouveaux jeux de données [Das+17 ; Agr+19]. Ce service est coûteux en termes d'effort humain, et il est compliqué de s'assurer que tous les employés annotent les images de la même façon. OQUAB et al. [Oqu+15] se demandaient :

> *"Par exemple, devrions nous annoter la tête du chien ou le corps entier ? Que fait-on quand la tête du chien est cachée par un autre objet ? "*

Étant donné que la capacité d'un CNN à généraliser dépend largement de la quantité et de la variété des images vues à l'entrainement, il est important d'améliorer la construction des jeux de données.

Supervision. La très grande base de données Imagenet [Don+09] a permis l'entraînement de nombreux réseaux, et les résultats de classification ont ainsi été beaucoup améliorés. Cependant, pour éviter d'avoir besoin d'acquérir des millions d'images annotées par des hommes pour chacune des tâches de vision par ordinateur, la communauté travaille à minimiser les besoins en supervision. En particulier, des images non annotées sont accessibles en grandes quantités sur internet et divers types d'informations peuvent en être extraits. Des efforts ont été fournis pour intégrer ces images aux entrainements des modèles.

La classification est la tâche qui nécessite le moins de labels et les labels les plus simples. Ainsi, minimiser la supervision revient à utiliser des données non annotées, soit en faisant de l'apprentissage avec seulement les images non annotées (apprentissage *non supervisé* [JMF99 ; HS06 ; Goo+14]), peu de données labellisées (apprentissage *few-shot* [Lak+11 ; Koc15]) ou un mixte des deux (apprentissage *semi-supervisé* [Zhu+06 ; Lee13 ; Shi+18]). L'apprentissage *faiblement supervisé* [Li+17] peut aussi être implémenté en ajoutant des labels plus ou moins bien annotés. Alternativement, il est possible de sélectionner les images les plus pertinentes à l'entrainement d'un modèle, et de les faire annoter par des humains, ce qui est l'idée principale de l'*apprentissage actif* [Yan+15b ; SS18]. Différents niveaux de supervision peuvent aussi être combinés de façon à tirer profit de tous les efforts et types de données accessibles [DP18].

Les tâches nécessitant des annotations plus détaillées, comme la détection d'objet ou la segmentation d'images, peuvent aussi bénéficier de labels moins précis. Les méthodes d'apprentissage faiblement supervisé permettent d'effectuer la tâche de détection d'objet tout en profitant d'entrainements effectués pour la tâche de classification [Oqu+15] et la tâche de segmentation d'images peut profiter des supervisions pour la classification et la détection d'objets [He+17]. D'autres formes de supervision plus faibles que des boîtes englobantes peuvent aussi être définis *e.g.* des points [Bea+15] ou des estimations de taille [SF16b].

Sans jouer sur le nombre de données labellisées utilisées, il est aussi possible de transférer directement l'apprentissage d'un modèle à un autre en utilisant la technique du *transfert d'apprentissage*. En particulier, les modèles entrainés sur Imagenet [Don+09] peuvent être utilisés de façon efficace en tant qu'initialisation pour de nouvelles tâches de classification ou même d'autres tâches. La base de donnée Imagenet contient 1000 classes, ce qui est bien plus que la plupart des autres jeux de données d'apprentissage. Effectuer le *pré-entrainement* d'un modèle est devenu une pratique ordinaire. De nombreuses expériences ont prouvé qu'initialiser un modèle avec des poids pré-entrainés améliore grandement la qualité de l'apprentissage, à la fois en termes de vitesse d'apprentissage et de résultats, dans toutes

les tâches incluant la classification [Lec+98], la détection d'objets [Lin+14], la segmentation d'images [He+17] ou la recherche d'images [Rad+18].

Comprendre et chercher des images

Comme nous l'avons vu, le domaine de la vision par ordinateur couvre de nombreux problèmes et challenges. En particulier, nous nous concentrons dans cette thèse sur deux applications : les tâches de classification d'image et de recherche d'image. Ces deux tâches nécessitent l'extraction de différents concepts dans une image. Un modèle construit pour la classification d'image doit être insensible aux variations intra-classe alors que la tâche de recherche d'image nécessite l'insensibilité aux variations des images, *e.g.* zoom, obscurité, représentant un objet en particulier. Ces deux tâches différent aussi de par le type de supervision et d'architecture des modèles utilisés pour les résoudre.

La classification d'images

Parmi toutes les tâches de reconnaissance d'objets, la classification d'images est l'une des plus simples en termes de supervision. En considérant un nombre limité de classes, la tâche consiste à prédire la classe des objets présents dans une image. Comme tout problème se concentrant sur le concept de classe, les modèles utilisés pour la classification d'images doivent appréhender de grandes variations intra-classes. Les méthodes basées sur des descripteurs locaux échouent à extraire des informations sémantiques qui pourraient permettre à un classifieur d'effectuer la bonne prédiction. Cependant, un modèle CNNs peut être conçu pour extraire des informations conceptuelles en utilisant une fonction de perte appropriée (*e.g.* la fonction d'erreur d'entropie croisée) et un jeu de données annotées par un label de classe. La capacité d'un CNN à généraliser à différents objets de la même classe dépend de la quantité des images vues pendant l'entrainement.

La recherche d'images

Dans la deuxième partie de cette thèse, nous nous concentrons sur la tâche de recherche d'images. La tâche nécessite de sélectionner et d'ordonner par pertinence toutes les images, contenues dans une base de données, représentant le même objet que celui dépeint dans l'image requête. Il est possible d'effectuer une recherche rapide en utilisant des représentations *globales*. La précision du tri dépend alors directement de la qualité de la représentation globale. Cette représentation peut soit être une agrégation de descripteurs locaux [PD07 ; Jég+10], soit un descripteur global extrait directement d'un modèle CNN [Gor+17 ; RTC18]. Cependant, un descripteur global manque souvent d'information détaillée. Il contient en effet des informations sur l'image entière mais rate des détails et ne se concentre pas sur les objets d'intérêt.

La recherche de correspondances régionales [TSJ16] améliore grandement les performances. Cette technique consiste à comparer de façon exhaustive différentes parties des images, chaque partie représentée par plusieurs descripteurs. Faire une recherche régionale est bien plus coûteux qu'effectuer une recherche globale, à la fois en terme de coût de calcul et de stockage. Un tel coût ne passe pas à l'échelle quand de très larges bases de données sont utilisées.

Si un premier tri a été effectué en utilisant des représentations globales, il est possible de réordonner les premières images en utilisant des techniques plus coûteuses. La *mise en correspondance spatiale* [FB81 ; CMK03] est effectuée sur une paire d'images et consiste à trouver la transformation permettant de passer des descripteurs locaux de la première image de la paire à ceux de la deuxième, et inversement. Cette mise en correspondance spatiale écarte les images qui ne contiennent pas les mêmes détails. Cependant, mettre en place une telle technique nécessite de stocker les descripteurs locaux. A la place ou en addition, il est possible d'utiliser des méthodes d'*expansion des requêtes* [Chu+07] qui permettent de découvrir des images similaires à une version étendue de l'image requête. Même si cette méthode ne nécessite pas plus d'information que les descripteurs globaux, effectuer une expansion de requête ajoute le coût d'une seconde recherche globale.

Nos contributions

Dans cette thèse, nous examinons plusieurs techniques permettant *d'extraire le plus de connaissance possible avec un minimum de supervision*. En particulier, nous nous concentrons sur les opportunités fournies par un modèle CNN. Comme étudié précédemment, un même réseau "backbone" peut être utilisé pour effectuer différentes tâches. Nous montrons que la sortie convolutive du réseau "backbone", un ensemble de *cartes de caractéristique*, peut être exploitée de différentes façons comme présenté dans la Figure 1.

Celle-ci résume les différentes contributions présentées dans cette thèse. Une fois qu'une image est représentée par des cartes de caractéristiques, il est possible de prédire les labels de classe en utilisant une couche entièrement connectée (FC) comme présenté par la Figure 1 (a). Un descripteur global peut être construit par "pooling" (Figure 1 (b)) à partir des cartes de caractéristiques et utilisé dans le contexte de la recherche d'image. La qualité de la recherche dépend largement de la représentation des images. En particulier, une image contient souvent plus d'objets que juste celui d'intérêt, ce qui peut altérer la qualité du descripteur global. Nous proposons une méthode pour identifier des objets d'intérêt et focaliser la représentation sur eux (Figure 1 (c)). Finalement, nous montrons que des caractéristiques locales peuvent être extraites directement dans les cartes de caractéristiques, permettant d'effectuer une mise en correspondance spatiale précise (Figure 1 (d)).

Revisiter l'apprentissage actif

Nous nous concentrons d'abord sur la tâche de classification et nous essayons de minimiser les besoins en supervision. Dans le scénario de l'apprentissage actif [Set09], un certain budget d'images à annoter est alloué, et ces images doivent être intelligemment sélectionnées de façon a maximiser la qualité de l'apprentissage d'un modèle effectué avec elles. Cette tâche réduit les besoins en images annotées. L'apprentissage actif est très pertinent dans le contexte des caractéristiques locales. Les images sont sélectionnées une à la fois, annotées, et ajoutées au jeu d'entrainement. Quand les réseaux neuronaux ont atteint les meilleurs résultats de la tâche de classification d'images, le processus de sélection a été adapté aux besoins des CNNs. En particulier, annoter une image à la fois ne serait pas pertinent étant donné que les réseaux sont entrainés en utilisant des groupes d'images. C'est aujourd'hui une pratique courante que d'annoter un certain *budget* d'images [GE17] à chaque cycle d'apprentissage, le budget allant généralement de centaines et des milliers d'images.

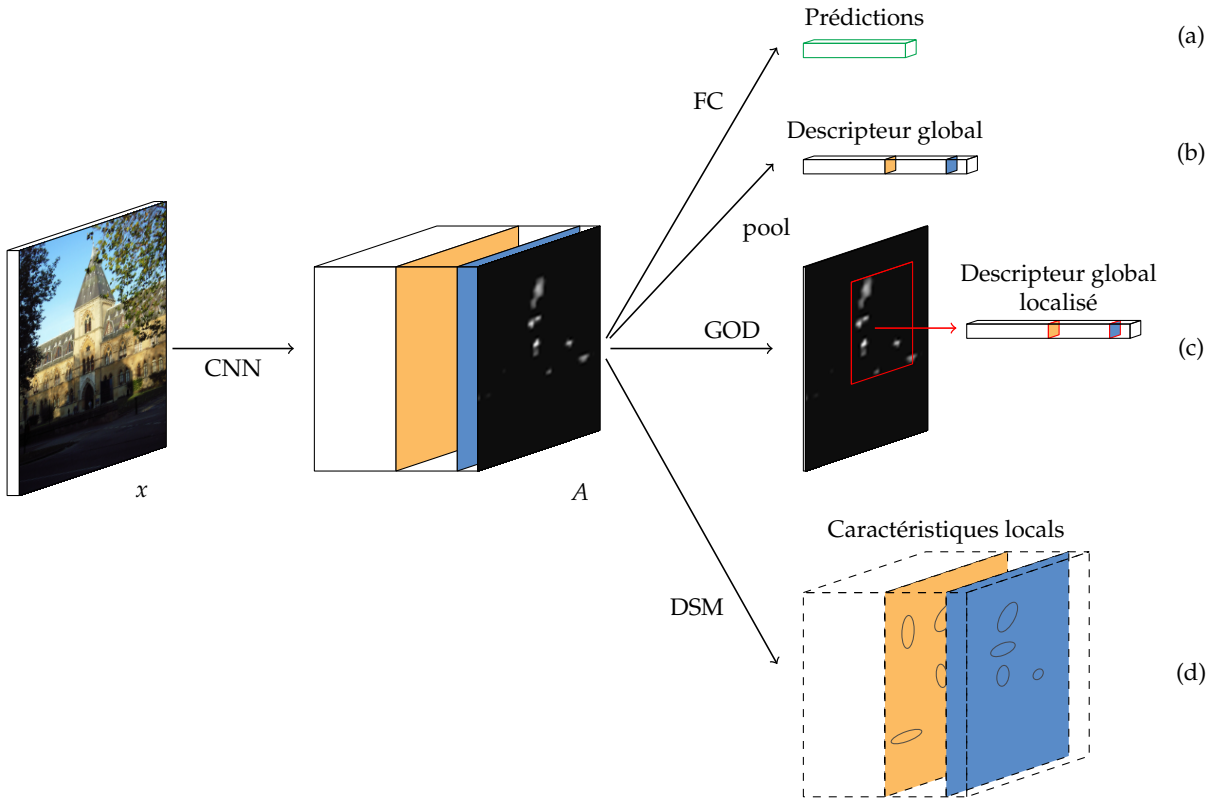


FIGURE 1 – Résumé de la thèse. Nous considérons un image d'entrée x transformée par un réseau "backbone" entièrement composé de couches convolutives. Nous montrons différentes façons d'exploiter les cartes de caractéristique A, produites par le réseau, pour générer une représentation d'images robuste. Pour effectuer une classification d'images, il est possible de calculer un vecteur de prédictions (a) en utilisant une couche entièrement connectée (FC) (utilisé dans le Chapter 3). Une représentation globale peut-être extraite par "pooling" (b) à partir des cartes de caractéristiques (vu dans les Chapter 4 et 5) Il est aussi possible de construire une représentation globale localisée (c) en utilisant notre méthode GOD (Chapter 5). Finalement, (d) présente la sortie de notre extracteur DSM de caractéristiques locales présenté (Chapter 4).

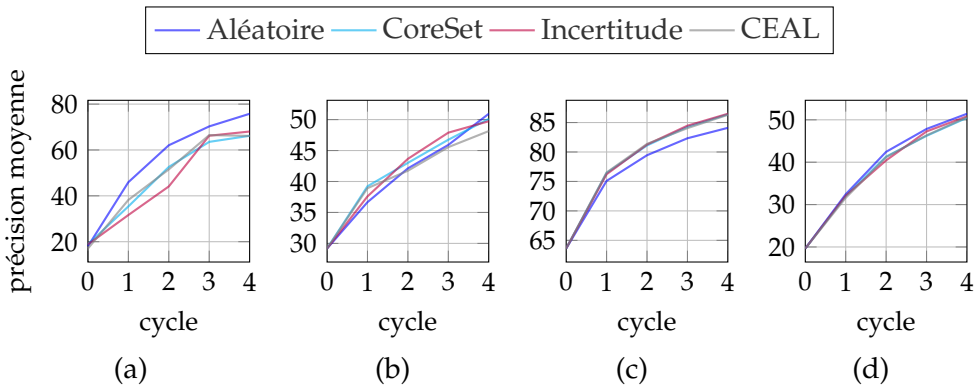


FIGURE 2 – Précision moyenne *vs*. cycle pour différentes configurations et fonctions d'acquisition sur les jeux de données SVHN avec un budget ($b = 100$)(a), CIFAR-10 avec un budget de 100 et 1000 ((b) et (c) respectivement), et CIFAR-100 avec $b = 1000$ (d).

Dans ce travail [Sim+20], nous revisitons l'apprentissage actif avec l'idée d'utiliser toutes les données, quelles soient annotées ou non, pendant l'apprentissage à chaque cycle de l'apprentissage actif. Ainsi, les images sans label contribuent maintenant directement à réduire la fonction de coût et à améliorer les paramètres du modèle utilisé, plutôt que de n'être utilisée que pour effectuer une acquisition. Nous implémentons notre idée en utilisant deux principes : *l'apprentissage non-supervisé de caractéristiques* et *l'apprentissage semi-supervisé*. Ces deux méthodes sont bien connues pour effectuer de l'apprentissage profond de façon générale, et nous soutenons que leur valeur a été sous-exploitée dans le contexte de l'apprentissage actif profond.

Plusieurs fonctions d'acquisition ont été proposées pour sélectionner des images, soit basées sur la géométrie [GE17 ; SS18] soit sur la certitude du modèle utilisé. Cependant, des travaux récents rapportent que les différences de performance entre les fonctions ne sont pas importantes [GS18 ; CAL19 ; Bel+18], et ne permettent pas de sélectionner la "meilleure" fonction d'acquisition. Nous faisons le même constat comme présenté par la Figure 2. Nous comparons sur différents jeux de données plusieurs fonctions d'acquisition : une acquisition *aléatoirement* uniforme, correspondant à une apprentissage non-actif, la méthode d'*incertitude* basée sur l'entropie, *CEAL* [Wan+17], et *CorSet* [GE17 ; SS18]. Nous constatons que la différence de résultat entre les différentes fonctions d'acquisition n'est pas significative, seule l'acquisition aléatoire se distingue.

Les CNNs apparaissent seulement partiellement dépendant des images utilisées pendant l'apprentissage. En particulier, les méthodes d'acquisition dépendent de la qualité des modèles utilisés, et, avec un petit budget nous montrons que les images sélectionnées sont moins pertinentes qu'une sélection aléatoire. Il apparait alors qu'effectuer de l'apprentissage actif dans le cadre de l'apprentissage profond nécessite plus de réflexion.

Nous proposons d'inclure des connaissances extraites à partir d'images non labellisées, en ajoutant des méthodes non-supervisée et semi-supervisée dans le processus de l'apprentissage actif. En particulier, nous suivons une configuration classique d'apprentissage actif, mais proposons d'effectuer un pré-apprentissage non-supervisé du modèle [Car+18b]. L'apprentissage non-supervisé de ces poids se fait en alternant entre une étape de partitionnement des données et une étape d'apprentissage. Nous utilisons les poids finaux pour initialiser nos modèles d'apprentissage actif. D'autre part, nous intégrons aussi la méthode de propagation de label [Isc+19a] qui s'appuie sur une diffusion des labels connus aux images

non annotées et propose d'entrainer le modèle sur toutes les images en utilisant les labels connus et prédits.

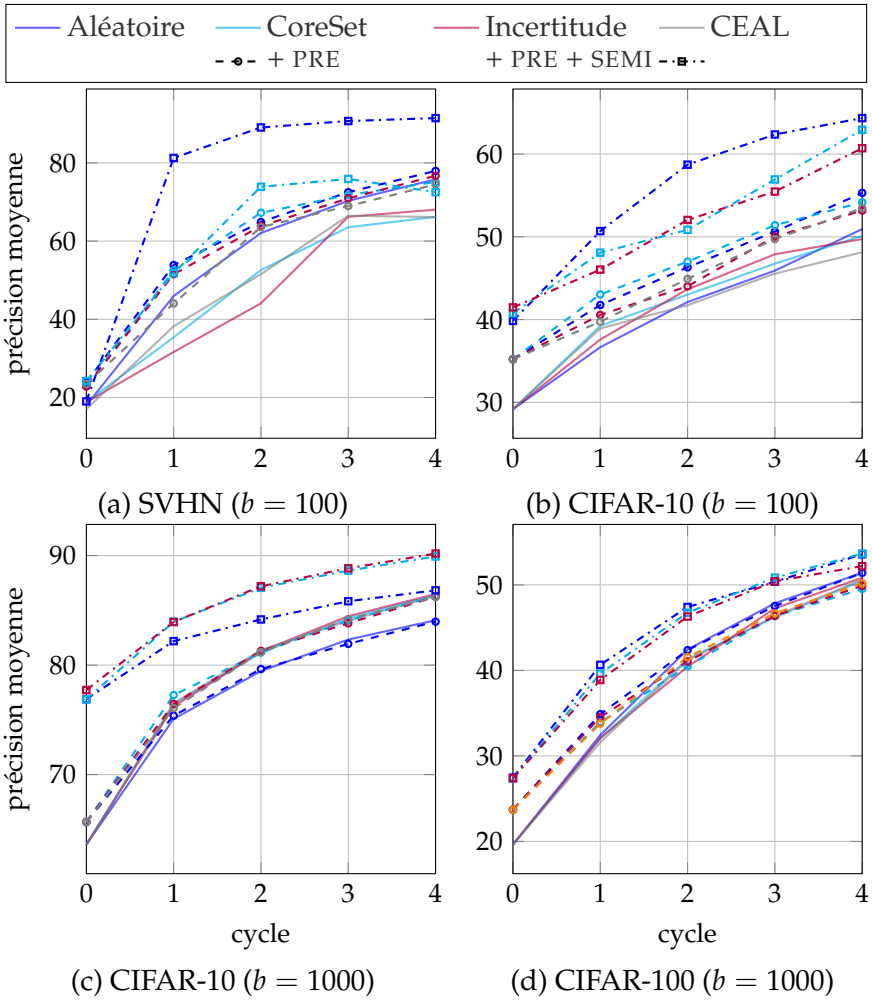


FIGURE 3 – Précision moyenne *vs.* cycle pour différentes configurations et fonctions d'acquisition : référence, PRE et PRE +SEMI. Les scénarios PRE et PRE + SEMI sont représentés par différentes lignes pointillées comme décrit dans la légende. Pour référence, la précision de l'entrainement complet est de 96.97% pour SVHN, 94.84 % pour CIFAR-10 et 76.43 % pour CIFAR-100.

Les lignes pointillées avec cercle dans la Figure 3 présentent les résultats obtenus avec le pré-apprentissage (PRE). Nous observons une amélioration de la précision allant jusqu'à 6% de gain. Cette amélioration est indépendante de l'utilisation du semi-apprentissage. Mais le gain le plus impressionnant est obtenu en ajoutant l'apprentissage semi-supervisé (PRE + SEMI). Dans la plupart des cas, le gain est de 10%, ce qui est de loin un bénéfice plus important que celui obtenu en utilisant une fonction d'acquisition ou une autre. A noter aussi que dans le cas de SVHN et d'un budget de 100, après 2 cycles d'apprentissage actif, la précision est presque aussi bonne qu'avec un apprentissage complet supervisé.

En conclusion, nous avons montré le grand intérêt à intégrer des images non annotées dans le processus de l'apprentissage actif. Nous proposons d'intégrer de façon systématique ces données à l'apprentissage actif et de façon générale de repenser l'apprentissage actif. Nous proposons aussi de toujours comparer les fonctions d'acquisition à une sélection

aléatoirement uniforme, qui est équivalente à ne pas effectuer d'apprentissage actif du tout, de façon à faire une comparaison juste.

Mises en correspondance spatiale

Des descripteurs globaux sont construits en extrayant par "pooling" des informations des cartes de caractéristiques. De façon traditionnelle, les détails spatiaux contenus dans une carte de caractéristiques sont agrégés dans une unique valeur, perdant ainsi toute information structurelle. Cependant, il est possible de faire plus qu'un simple "pooling" et garder ainsi les informations spatiales. Il a été démontré que les cartes de caractéristiques entrainées pour la tâche de classification sont actives sur différentes parties d'une image [Zho+16], qui correspondent à certaines zones caractéristiques. Ainsi, en utilisant des réseaux pré-entrainés sur des tâches de d'apprentissage de variété, il est possible d'exploiter les cartes de caractéristiques CNN et de construire des cartes de saillance qui peuvent aider à localiser les objets intéressants. Cette discussion, plutôt nouvelle, ouvre les porte à beaucoup de possibilités pour effectuer de l'apprentissage faiblement supervisé en utilisant seulement des labels de classe.

Dans ce travail [SAC19], nous étudions comment de telles informations de localisation peuvent être utilisées dans le contexte de la *mise en correspondance spatiale*. Cette tâche nécessite que les images soient représentées par un jeu de caractéristiques locales qui peuvent être mises en correspondance d'une image à l'autre. Les caractéristiques sont généralement traduites par une position, une région et un descripteur qui doit être insensible aux variations de l'instance de l'objet. Nous proposons une nouvelle technique pour extraire les descripteurs locaux directement dans les cartes de caractéristiques, de façon presque "gratuite".

Des expériences dans [TSJ16] ont montré que la méthode de "pooling" MAC était supérieure à d'autres techniques de "pooling", au moins dans le cas de la recherche d'images. Cette observation peut être rapprochée des travaux montrant que les cartes de caractéristiques sont clairsemées, et que cette information de quantité d'activation peut être utilisée pour construire des descripteurs [KMO15 ; KMO16]. De plus, en regardant la position des éléments maximums des canaux de cartes de caractéristiques contribuant le plus lors de la recherche par similarité, il es possible d'établir rapidement des correspondances entre les images [TSJ16]. Plus tard, la méthode de "pooling" GeM [RTC18] s'est avérée meilleure que MAC. Ceci peut être attribué au fait que cette technique utilisent l'information de plusieurs localisations dans les cartes de caractéristiques.

En suivant ces observations, nous étudions les réponses de la dernière couche convolutive du réseau VGG sur plusieurs images du jeu de données $\mathcal{R}$Oxford. Nous découvrons que, comme le montre la Figure 4, dans la plupart des cas les réponses sont clairsemées et cohérentes d'une image représentant un objet à une autre représentant ce même objet. Les réponses sont invariantes à la translation d'un objet et aussi à une certaine modification du zoom.

Ces idées ont initié notre projet DSM illustré dans la Figure 5. Nous considérons un réseau entièrement fait de couches convolutives. Nous détectons des régions *affines* dans chaque canal des cartes de caractéristiques. Nous trions ensuite les images prometteuses obtenues par recherche globale en les mettant en correspondance en utilisant la technique FSM [Phi+07]. En particulier, nous forçons les caractéristiques associées par cette méthode à provenir des mêmes canaux, ce qui impose que l'information *sémantique* contenue par les

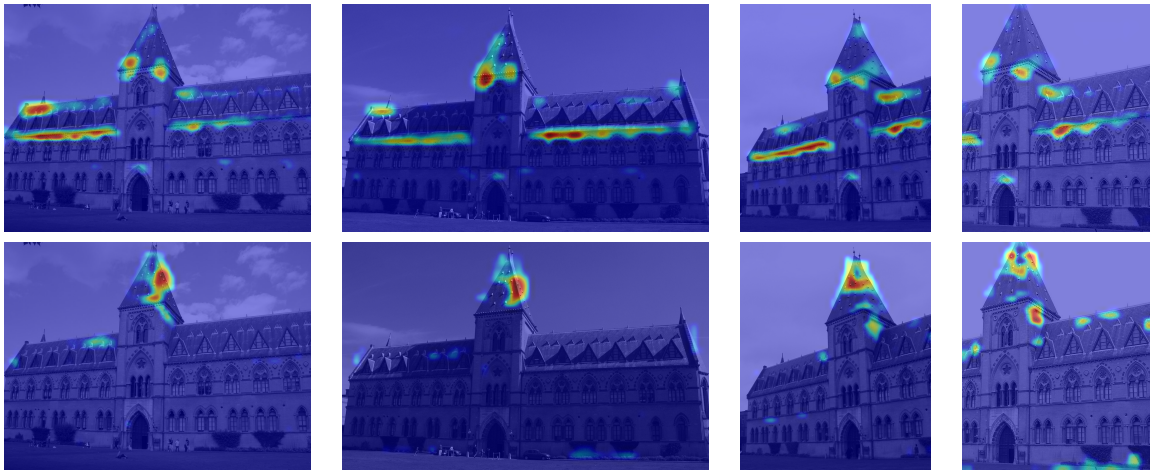


FIGURE 4 – Quatre vues (en colonne) du musée d'histoire naturelle dans le jeu de données $\mathcal{R}$Oxford, sur lesquelles sont superposées deux cartes de caractéristiques différentes (lignes) de la dernière couche convolutive du réseau VGG16 [SZ14]. Le noyau de chaque canal répond à des structures similaires dans les images. Toutes les activations sont naturellement sporadique et les réponses non nulles coïncident à la fois en termes de location et en termes de forme locale entre toutes les images.

cartes de caractéristiques soit maintenue pendant la mise en correspondance spatiale. Cette mise en correspondance sémantique diminue le besoin d'avoir des descripteurs.

Notre méthode obtient de bonnes performances de mise en correspondance comme il est possible de le voir dans la Figure 6. Les caractéristiques obtenues et mises en relations sont la plupart du temps cohérentes. Notre technique a l'avantage de pouvoir être appliquée à n'importe quel réseau. Et, de façon intéressante les résultats obtenus sur des réseaux seulement pré-entrainés sur Imagenet [Don+09] sont bons, comme montré dans la Table 1. Nous utilisons la technique de "pooling" GeM, qui est meilleure que MAC sur de tels réseaux [Rad+18]. La Table 1 montre l'effet de DSM sur $\mathcal{R}$Oxford et $\mathcal{R}$Paris dans les scénarios moyens et durs. Nous améliorons les résultats avec et sans diffusion [Isc+17]. Le gain est significatif sur $\mathcal{R}$Oxford, jusqu'à 13% meilleur sur VGG-GeM avec diffusion, dans le scénario moyen. L'amélioration est moindre sur $\mathcal{R}$Paris, où les performances sont déjà de 20 à 40 points de mAP plus hautes que $\mathcal{R}$Oxford.

En conclusion, nos expériences valident que la représentation DSM proposée pour la mise en correspondance spatiale atteint des performances qui sont à l'état de l'art pour différents jeux de données, réseaux et mécanismes de "pooling". Cette représentation émane naturellement dans les cartes de caractéristiques existantes provenant des réseaux pré-entrainés sur Imagenet ou entrainés pour la recherche d'images, sans aucun effort particulier pour détecter des caractéristiques locales ou extraire des descripteurs locaux sur les patchs d'image. Cette méthode ne nécessite aucune modification des réseaux ou ré-entrainements. C'est un pas important vers la réduction du fossé entre les descripteurs globaux, qui sont efficaces pour faire un classement initial en utilisant une recherche des plus proches voisins, et les représentations locales, qui sont compatibles avec la mise en correspondance spatiale.

Exploration d'une base de données sans supervision

Nous proposons une nouvelle méthode de "pooling", qui tire profit de l'information contenue dans les cartes de caractéristique. Nous proposons de considérer le jeu de données

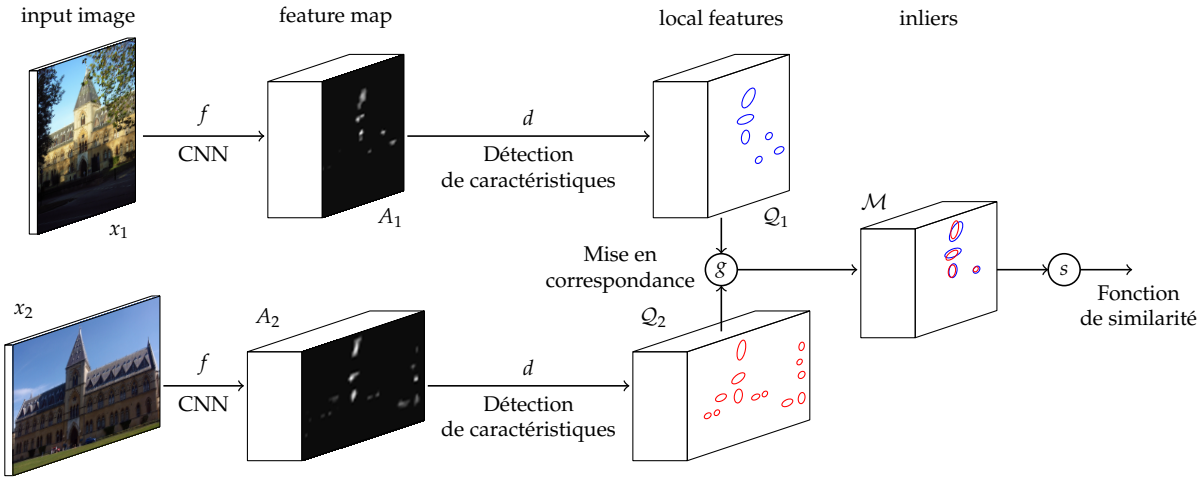


FIGURE 5 – Architecture du réseau DSM. Deux images d'entrée x_1, x_2 sont transformées par le réseau f donnant respectivement les tenseurs de caractéristiques A_1, A_2. Les *caractéristiques locales* $\mathcal{Q}_1, \mathcal{Q}_2$ sont extraites par un *détecteur* d, sont traitées par une méthode de *mise en correspondance spatiale* g, résultant en une collection d'"inliers" $\mathcal{M}$. Une fonction de similarité s est appliquée à la collection. Les caractéristiques locales sont détectées et mises en correspondance indépendamment par canal, et les canaux jouent le rôle de *mots visuels*. Cette méthode est réalisée sans nécessiter d'entrainement additionnel et sans avoir à adapter le réseau "backbone". En recherche d'images, seuls les descripteurs locaux $\mathcal{Q}_1, \mathcal{Q}_2$ sont stockés et g s'applique directement au second classement.

Method	Moyen				Dur			
	$\mathcal{R}$Oxford		$\mathcal{R}$Paris		$\mathcal{R}$Oxford		$\mathcal{R}$Paris	
	mAP	mP@10	mAP	mP@10	mAP	mP@10	mAP	mP@10
V	44.8	63.3	65.7	95.0	18.4	31.2	41.0	79.1
V+DSM	51.1	77.3	66.2	96.9	25.3	40.3	41.0	81.7
R↑	44.4	64.2	69.0	96.4	17.7	31.2	46.5	85.3
R↑+DSM	49.6	74.0	69.7	98.4	21.7	37.6	46.7	87.0
V+D	48.4	65.2	81.4	95.6	24.8	37.1	67.1	93.0
V+DSM+D	61.6	81.0	82.8	97.6	35.5	48.1	68.7	95.9
R↑+D	53.8	69.0	85.6	96.3	29.8	38.1	72.1	94.1
R↑+DSM+D	60.2	78.9	86.3	96.9	33.1	42.0	72.8	95.0

TABLEAU 1 – Impact de la méthode proposée (DSM) sur mAP et mP@10 sur les jeux de données $\mathcal{R}$Oxford et $\mathcal{R}$Paris [Rad+18] avec les réseaux VGG (V) et ResNet (R) pré-entrainés sur Imagenet [Don+09] . ↑ : sur-échantillonnage ; D : diffusion [Isc+17]. DSM : ce travail. Tous les résultats sont avec le "pooling" GeM et transformation de blanchiment ("whitening") supervisé.

FIGURE 6 – Exemples de notre méthode DSM entre des images des jeux de données $\mathcal{R}$Oxford et $\mathcal{R}$Paris. Les "inliers" (ellipses) et correspondances (lignes) sont montrées de différentes couleurs.

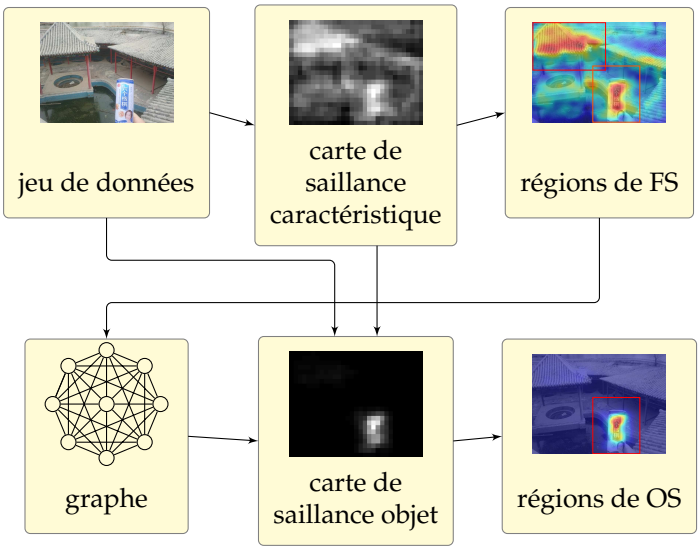


FIGURE 7 – Aperçu de notre méthode non supervisée. Sur la première ligne, les cartes de caractéristiques des images du jeu de données sont utilisées pour extraire une première *carte de saillance caractéristique*, notée FS, sur lesquelles des régions d'intérêt sont détectées. Sur la ligne du bas, une mesure de *centralité* est obtenue par région, à partir d'un graphe k-NN. En utilisant cette valeur, nous pouvons construire une *carte de saillance objet* notée OS. Cette carte se concentre sur les objets découverts directement à partir du jeu de données. Finalement, nous détectons un nouveau jeu de régions d'intérêt qui est utilisé pour extraire un descripteur représentant l'image.

dans son entièreté, le structurant en graphe, et de faire de l'exploration de données pour découvrir les objets d'intérêt, et ce en utilisant le simple concept de répétition.

En particulier, notre méthode GOD [Sim+19] découvre les objets dans des images en combinant les informations contenues dans les cartes de caractéristiques et le graphe du jeu de données. Nous sommes capables de construire des *cartes de saillance* qui mettent en évidence avec une grande précision les objets désirés. Une fois qu'ils sont découverts, nous pouvons construire une représentation globale localisée, améliorant ainsi grandement la recherche globale.

La Figure 7 présente un aperçu de notre technique. Pour toutes les images d'un jeu de données, nous commençons par extraire des régions d'intérêt. Pour se faire, nous commençons par construire une *carte de salliance caractéristique* à partir des cartes de caractéristiques CNN. Nous nous inspirons des travaux CroW [KMO16] and Grad-CAM [Sel+17] qui agrègent les information contenues par les cartes de caractéristiques. Une fois ces cartes de sailliance obtenues, nous détectons des régions rectangulaires sur les cartes en utilisant EGM [AK12]. Ce premier tour de détection est effectué indépendamment pour chaque image et dépend seulement de son contenu.

Chaque région du jeu de données est ensuite associée à un score de saillance et un descripteur visuel extrait des cartes de caractéristiques correspondant à l'image. Il est alors possible de calculer un score de *centralitée par région*, représentant l'"importance" de chaque région dans le jeu de données. Cette mesure est basée sur un graphe k-NN.

A présent, il est possible d'évaluer l'"importance" des régions d'une *nouvelle* image, en utilisant les scores des régions les plus proches. Cette étape permet de construire une *carte de saillance objet* qui fait ressortir les objets d'intérêt d'une image. Finalement, nous détectons

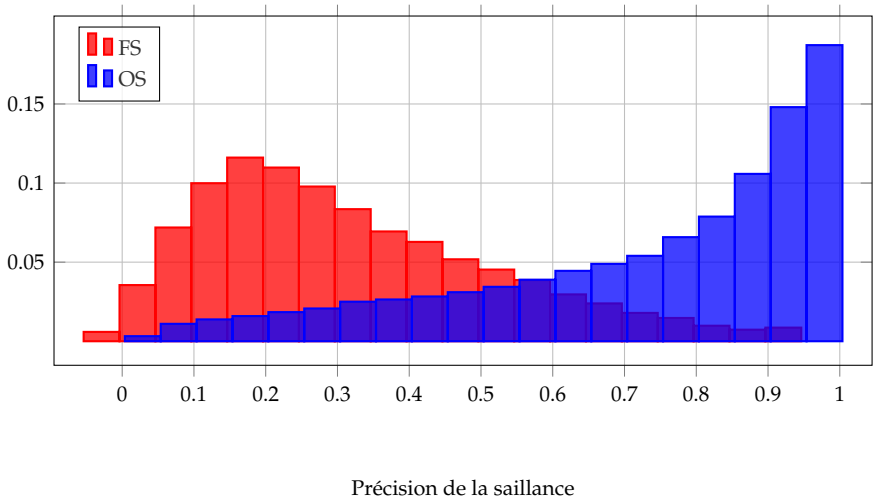


FIGURE 8 – Histogramme de la précision des cartes de saillance pour les cartes FS et OS mesurées sur toutes les images du jeu de données Instre.

un petit jeu de régions rectangulaires, en utilisant une nouvelle fois EGM, sur cette carte de saillante et nous extrayons un descripteur global qui représente l'image. Ce descripteur est ainsi concentré sur les objets importants. Cette seconde procédure de détection prend en compte tous les objets saillants et se répétant dans un jeu de données.

Cette méthode est entièrement non-supervisée et se base seulement sur des réseaux déjà entrainés pour la tache de classification ou de recherche d'images sans annotations de boîtes.

Nous évaluons la qualité des cartes de saillance FS et OS, visibles dans la Figure 10 sur le jeu de données Instre. Les résultats sont présentés dans la Figure 8 . La précision de la saillance est une mesure évaluant si la saillance est bien concentrée sur les objets d'intérêt. Elle est calculée en utilisant les annotations de boîtes connues pour Instre. Nous observons que OS est bien plus précise et concentrée sur les objet recherchés.

D'autre part, nos résultats montrent que nos descripteurs globaux concentrés sur les objets importants sont de bonne qualité et permettent d'effectuer une recherche globale efficace. La Figure 9 permet d'établir que notre recherche globale nécessite 4 fois moins de mémoire pour reproduire les résultats obtenus en effectuant une recherche régionale. Le gain en complexité est de l'ordre de 4^2.

En conclusion, notre approche ne nécessite aucune supervision et est spécifique à un jeu de données. Elle capture non seulement des objets importants en considérant chaque image individuellement mais aussi ceux qui apparaissent fréquemment dans le jeu de données. Nous n'avons donc pas besoin de descripteurs régionaux, et nous construisons un descripteur global en extrayant les informations contenus dans les régions. Nos résultats montrent que cette technique est efficace pour la recherche d'images.

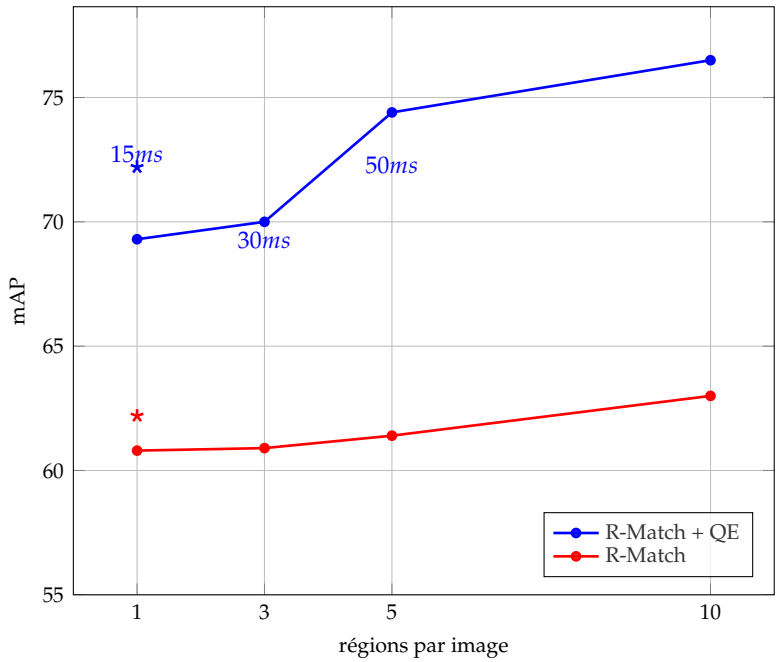


FIGURE 9 – Comparaison des scores mAP de notre représentation globale OS.EGM ($\star$) aux résultats obtenus avec R-Match et des descripteurs régionaux obtenus de façon uniforme aléatoire, ce avec et sans diffusion sur le jeu de données $\mathcal{R}$Oxford (évaluation des cas moyens). Les labels textuels correspondent au temps de réponse à la requête.

FIGURE 10 – Exemples d'images provenant des jeux de données $\mathcal{R}$Oxford (deux premières lignes) et Instre (dernières 3 lignes), à coté des cartes FS and OS superposées sur les images avec les régions détectées par EGM [AK12], en rouge.

Bibliography

References

[17] "The world's most valuable resource is no longer oil, but data", *in*: *The Economist* (2017).

[58] "New navy device learns by doing", *in*: *New York Times* (1958).

[Agr+19] Harsh Agrawal* *et al.*, "nocaps: novel object captioning at scale", *in*: *ICCV*, 2019.

[AK12] Yannis Avrithis and Yannis Kalantidis, "Approximate Gaussian Mixtures for Large Scale Vocabularies", *in*: *ECCV*, Springer, 2012, pp. 15–28.

[Alm+18] Jon Almazan, Bojana Gajic, Naila Murray, and Diane Larlus, "Re-ID done right: towards good practices for person re-identification", *in*: *arXiv preprint arXiv:1801.05339* (2018).

[Alv+19] Aitor Alvarez-Gila, Adrian Galdran, Estibaliz Garrote, and Joost Weijer, "Self-supervised blur detection from synthetically blurred scenes", *in*: *Image and Vision Computing* (Aug. 2019).

[Ara+16] Relja Arandjelović, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic, "NetVLAD: CNN architecture for weakly supervised place recognition", *in*: *CVPR*, 2016.

[AV07] D. Arthur and S. Vassilvitskii, "K-Means++: the Advantages of Careful Seeding", *in*: *SODA*, Society for Industrial and Applied Mathematics, 2007.

[AZ12] Relja Arandjelovic and Andrew Zisserman, "Three things everyone should know to improve object retrieval", *in*: *CVPR*, June 2012.

[AZ14] Relja Arandjelović and Andrew Zisserman, "Visual vocabulary with a semantic twist", *in*: 2014.

[Azi+14] Hossein Azizpour, Ali Sharif Razavian, Josephine Sullivan, Atsuto Maki, and Stefan Carlsson, "From generic to specific deep representations for visual recognition", *in*: *arXiv* (2014).

[Bab+14] Artem Babenko, Anton Slesarev, Alexandr Chigorin, and Victor Lempitsky, "Neural Codes for Image Retrieval", *in*: *ECCV*, 2014.

[Bag+10] Shai Bagon, Ori Brostovski, Meirav Galun, and Michal Irani, "Detecting and sketching the common", *in*: *CVPR*, 2010.

[Bal81] Dana H. Ballard, "Generalizing the Hough transform to detect arbitrary shapes", *in*: *Pattern Recognition* 13 (1981), pp. 111–122.

[Bau+16] Miguel Ángel Bautista, Artsiom Sanakoyeu, Ekaterina Sutter, and Björn Ommer, "CliqueCNN: Deep Unsupervised Exemplar Learning", *in*: *NIPS*, 2016.

[Bay+08] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool, "Speeded-up robust features (SURF)", *in*: *CVIU* 110.3 (2008), pp. 346–359.

[Baz+14] Loris Bazzani, Alessandro Bergamo, Dragomir Anguelov, and Lorenzo Torresani, "Self-taught Object Localization with Deep Networks", *in*: *arXiv* (2014).

[Bea+15] Amy L. Bearman, Olga Russakovsky, Vittorio Ferrari, and Fei-Fei Li, "What's the Point: Semantic Segmentation with Point Supervision", *in*: *ECCV*, 2015.

[Bel+18] William H Beluch, Tim Genewein, Andreas Nürnberger, and Jan M Köhler, "The power of ensembles for active learning in image classification", *in*: *CVPR*, 2018.

[Ber+14] Thomas Berg, Jiongxin Liu, Seung Woo Lee, Michelle L. Alexander, David W. Jacobs, and Peter N. Belhumeur, "Birdsnap: Large-Scale Fine-Grained Visual Categorization of Birds", *in*: *CVPR* (2014), pp. 2019–2026.

[Ber+16] Luca Bertinetto, João F. Henriques, Jack Valmadre, Philip H. S. Torr, and Andrea Vedaldi, "Learning Feed-forward One-shot Learners", *in*: *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, Barcelona, Spain: Curran Associates Inc., 2016, pp. 523–531.

[Ber+19] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin Raffel, "Mixmatch: A holistic approach to semi-supervised learning", *in*: *arXiv preprint arXiv:1905.02249* (2019).

[BGV92] Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik, "A Training Algorithm for Optimal Margin Classifiers", *in*: *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, Pittsburgh, Pennsylvania, USA: ACM, 1992, pp. 144–152.

[BJ17] Piotr Bojanowski and Armand Joulin, "Unsupervised Learning by Predicting Noise", *in*: *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML, Sydney, NSW, Australia: JMLR.org, 2017, pp. 517–526.

[BL15] Artem Babenko and Victor Lempitsky, "Aggregating Deep Convolutional Features for Image Retrieval", *in*: *ICCV*, 2015.

[Boj+17] Piotr Bojanowski, Armand Joulin, David Lopez-Paz, and Arthur Szlam, "Optimizing the Latent Space of Generative Networks", *in*: *arXiv* abs/1707.05776 (2017).

[Bon87] Phillip Bonacich, "Power and Centrality: A Family of Measures", *in*: *American Journal of Sociology* 92.5 (1987), pp. 1170–1182.

[BR19] Eric Brachmann and Carsten Rother, "Neural-Guided RANSAC: Learning Where to Sample Model Hypotheses", *in*: *ICCV*, 2019.

[Bra+17] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumhold, and Carsten Rother, "DSAC-differentiable RANSAC for camera localization", *in*: *CVPR*, vol. 3, 2017.

[C C+11] Dan C. Cireşan, Ueli Meier, Jonathan Masci, Luca M. Gambardella, and Jürgen Schmidhuber, "High-Performance Neural Networks for Visual Object Classification", *in*: *Computing Research Repository - CORR* (Feb. 2011).

[Cal+10] M. Calonder, Vincent Lepetit, C. Strecha, and Pascal Fua, "BRIEF: Binary Robust Independent Elementary Features", *in*: *ECCV*, Oct. 2010.

[CAL19] Kashyap Chitta, Jose M Alvarez, and Adam Lesnikowski, "Large-Scale Visual Active Learning with Deep Probabilistic Ensembles", *in*: *arXiv preprint arXiv:1811.03575* (2019).

[Car+18a] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze, "Deep Clustering for Unsupervised Learning of Visual Features", *in*: *ECCV*, ed. by Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, Cham: Springer International Publishing, 2018, pp. 139–156.

[Car+18b] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze, "Deep Clustering for Unsupervised Learning of Visual Features", *in*: *arXiv preprint arXiv:1807.05520* (2018).

[CBS19] Titus Cieslewski, Michael Bloesch, and Davide Scaramuzza, "Matching features without descriptors: implicitly matched interest points", *in*: *BMVC*, 2019.

[CFS09] Jie Chen, Haw-ren Fang, and Yousef Saad, "Fast Approximate k NN Graph Construction for High Dimensional Data via Recursive Lanczos Bisection", *in*: *The Journal of Machine Learning Research* 10 (Sept. 2009), pp. 1989–2012.

[CH67] T. Cover and P. Hart, "Nearest neighbor pattern classification", *in*: *IEEE Transactions on Information Theory* 13.1 (Jan. 1967), pp. 21–27.

[Cha+09] Vijay Chandrasekhar, Gabriel Takacs, David M. Chen, Sam S. Tsai, Radek Grzeszczuk, and Bernd Girod, "CHoG: Compressed histogram of gradients A low bit-rate feature descriptor", *in*: *2009 IEEE Conference on Computer Vision and Pattern Recognition* (2009), pp. 2504–2511.

[Cha+19] Cheng Chang, Guangwei Yu, Chundi Liu, and Maksims Volkovs, "Explore-Exploit Graph Traversal for Image Retrieval", *in*: *CVPR*, June 2019.

[Che+18] Bowen Cheng, Yunchao Wei, Honghui Shi, Shiyu Chang, Jinjun Xiong, and Thomas S. Huang, "Revisiting Pre-training: An Efficient Training Method for Image Classification", *in*: *arXiv preprint arXiv:1811.09347* (2018).

[CHL05] Sumit Chopra, Raia Hadsell, and Yann LeCun, "Learning a Similarity Metric Discriminatively, with Application to Face Verification", *in*: *CVPR*, June 2005, pp. 539–546.

[Cho+15] Minsu Cho, Suha Kwak, Cordelia Schmid, and Jean Ponce, "Unsupervised object discovery and localization in the wild: Part-based matching with bottom-up region proposals", *in*: *CVPR*, 2015.

[Cho+16] Christopher B Choy, JunYoung Gwak, Silvio Savarese, and Manmohan Chandraker, "Universal correspondence network", *in*: *NIPS*, 2016, pp. 2414–2422.

[Chr15] Sergey Ioffe andpro Christian Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift", *in*: *ICML*, 2015.

[Chu+07] Ondřej Chum, James Philbin, Josef Sivic, Michael Isard, and Andrew Zisserman, "Total Recall: Automatic Query Expansion with a Generative Feature Model for Object Retrieval", *in*: *ICCV*, Oct. 2007.

[Chu+11] O. Chum, A. Mikulik, M. Perdoch, and J. Matas, "Total Recall II: Query Expansion Revisited", *in*: *CVPR*, June 2011.

[Chu97] Fan RK Chung, *Spectral graph theory*, vol. 92, American Mathematical Soc., 1997.

[CK10] Michael Connor and Piyush Kumar, "Fast Construction of k-Nearest Neighbor Graphs for Point Clouds", *in*: *IEEE transactions on visualization and computer graphics* 16 (July 2010), pp. 599–608.

[CM10] Ondřej Chum and Jiri Matas, "Unsupervised Discovery of Co-occurrence in Sparse High Dimensional Data", *in*: *CVPR*, June 2010.

[CMK03] Ondřej Chum, Jiri Matas, and Josef Kittler, "Locally Optimized RANSAC", *in*: *DAGM Symposium on Pattern Recognition*, Springer Verlag, 2003, p. 236.

[CMV15] Mircea Cimpoi, Subhransu Maji, and Andrea Vedaldi, "Deep filter banks for texture recognition and segmentation", *in*: June 2015, pp. 3828–3836.

[CN12] Adam Coates and Andrew Y. Ng, "Learning Feature Representations with K-Means", *in*: *Neural Networks: Tricks of the Trade: Second Edition*, ed. by Grégoire Montavon, Geneviève B. Orr, and Klaus-Robert Müller, Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 561–580.

[CPS06] Kumar Chellapilla, Sidd Puri, and Patrice Simard, "High Performance Convolutional Neural Networks for Document Processing", *in*: *Tenth International Workshop on Frontiers in Handwriting Recognition*, ed. by Guy Lorette, http://www.suvisoft.com, Université de Rennes 1, La Baule (France): Suvisoft, Oct. 2006.

[Csu+04] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual Categorization with Bags of Keypoints", *in*: *ECCVW*, 2004.

[CSZ06] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien, *Semi-Supervised Learning*, MIT Press, 2006.

[Cui+15] Yin Cui, Feng Zhou, Yuanqing Lin, and Serge Belongie, "Fine-grained Categorization and Dataset Bootstrapping using Deep Metric Learning with Humans in the Loop", *in*: *CVPR* (2015).

[CV95] Corinna Cortes and Vladimir Vapnik, "Support-Vector Networks", *in*: *Mach. Learn.* 20.3 (Sept. 1995), pp. 273–297.

[CW16] Taco Cohen and Max Welling, "Group Equivariant Convolutional Networks", *in*: *ICML*, 2016, pp. 2990–2999.

[CZG18] Yanbei Chen, Xiatian Zhu, and Shaogang Gong, "Semi-Supervised Deep Learning with Memory", *in*: (2018).

[Dai+16] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun, "R-FCN: Object Detection via Region-based Fully Convolutional Networks", *in*: *NIPS*, 2016.

[Dai+17] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei, "Deformable Convolutional Networks", *in*: *ICCV* (2017), pp. 764–773.

[Das+17] Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, José M.F. Moura, Devi Parikh, and Dhruv Batra, "Visual Dialog", *in*: *CVPR*, 2017.

[DB13] Michael Donoser and Horst Bischof, "Diffusion processes for retrieval revisited", *in*: *CVPR*, 2013.

[DCL11] Wei Dong, M. Charikar, and K. Li, "Efficient K-Nearest Neighbor Graph Construction for Generic Similarity Measures", *in*: *WWW*, Mar. 2011.

[DGE15] Carl Doersch, Abhinav Gupta, and Alexei A. Efros, "Unsupervised Visual Representation Learning By Context Prediction", *in*: *ICCV*, 2015.

[Dix+17] Mandar Dixit, Roland Kwitt, Marc Niethammer, and Nuno Vasconcelos, "AGA : Attribute-Guided Augmentation", *in*: (2017).

[DKD16] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell, "Adversarial Feature Learning", *in*: *arXiv* abs/1605.09782 (2016).

[DLR77] Arthur P Dempster, Nan M Laird, and Donald B Rubin, "Maximum likelihood from incomplete data via the EM algorithm", *in*: *Journal of the royal statistical society. Series B (methodological)* (1977), pp. 1–38.

[Don+09] Wei Dong, Richard Socher, Li Li-Jia, Kai Li, and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database", *in*: *CVPR*, June 2009.

[Dos+14] Alexey Dosovitskiy, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox, "Discriminative Unsupervised Feature Learning with Convolutional Neural Networks", *in*: *NIPS*, 2014.

[Dos+15a] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox, "Flownet: Learning optical flow with convolutional networks", *in*: *CVPR*, 2015, pp. 2758–2766.

[Dos+15b] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox, "Flownet: Learning optical flow with convolutional networks", *in*: *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2758–2766.

[Dou+18] Matthijs Douze, Arthur Szlam, Bharath Hariharan, and Hervé Jégou, "Low-shot learning with large-scale diffusion", *in*: *CVPR* (2018).

[DP18] Melanie Ducoffe and Frederic Precioso, "Adversarial Active Learning for Deep Networks: a Margin Based Approach", *in*: *arXiv preprint arXiv:1802.09841* (2018).

[DT05] Navneet Dalal and Bill Triggs, "Histograms of Oriented Gradients for Human Detection", *in*: *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, CVPR '05, Washington, DC, USA: IEEE Computer Society, 2005, pp. 886–893.

[Dum+16] Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Alex Lamb, Martín Arjovsky, Olivier Mastropietro, and Aaron C. Courville, "Adversarially Learned Inference", *in*: *arXiv* abs/1606.00704 (2016).

[Dus+19] M. Dusmanu, I. Rocco, T. Pajdla, M. Pollefeys, J. Sivic, A. Torri, and T. Sattler, "D2-Net: A Trainable CNN for Joint Description and Detection of Local Features", *in*: *CVPR*, Long Beach, CA, US, June 2019.

[Els+07] Jeremy Elson, John (JD) Douceur, Jon Howell, and Jared Saul, "Asirra: A CAPTCHA that Exploits Interest-Aligned Manual Image Categorization", *in*: *Proceedings of 14th ACM Conference on Computer and Communications Security (CCS)*, Oct. 2007.

[Est+18] Carlos Esteves, Christine Allen-Blanchette, Xiaowei Zhou, and Kostas Daniilidis, "Polar Transformer Networks", *in*: *ICLR*, 2018.

[FAL17] Chelsea Finn, Pieter Abbeel, and Sergey Levine, "Model-agnostic meta-learning for fast adaptation of deep networks", *in*: *ICML*, 2017.

[FB81] Martin A. Fischler and Robert C. Bolles, "Random sample consensus", *in*: *Communications of ACM* 6.24 (1981), pp. 381–395.

[Feu+18] Matthias Feurer, Katharina Eggensperger, Stefan Falkner, Marius Lindauer, and Frank Hutter, "Practical Automated Machine Learning for the AutoML Challenge 2018", *in*: 2018.

[FFP06] Li Fei-Fei, Rob Fergus, and Pietro Perona, "One-shot learning of object categories", *in*: *IEEE transactions on pattern analysis and machine intelligence* 28.4 (2006), pp. 594–611.

[FRS01] Karl Pearson F.R.S., "LIII. On lines and planes of closest fit to systems of points in space", *in*: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 2.11 (1901), pp. 559–572.

[Fuk75] Kunihiko Fukushima, "Cognitron: A self-organizing multilayered neural network", *in*: *Biological Cybernetics* 20.3 (Sept. 1975), pp. 121–136.

[Fuk80] Kunihiko Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position", *in*: *Biological Cybernetics* 36.4 (Apr. 1980), pp. 193–202.

[FXL18] Chelsea Finn, Kelvin Xu, and Sergey Levine, "Probabilistic model-agnostic meta-learning", *in*: *NIPS* (2018).

[FXT19] Zeyu Feng, Chang Xu, and Dacheng Tao, "Self-Supervised Representation Learning by Rotation Feature Decoupling", *in*: *CVPR*, June 2019.

[Gam+09] Stephan Gammeter, Lukas Bossard, Till Quack, and Luc V. Gool, "I Know What You Did Last Summer: Object-Level Auto-Annotation of Holiday Snaps", *in*: *ICCV*, 2009.

[GB10] Xavier Glorot and Yoshua Bengio, "Understanding the difficulty of training deep feedforward neural networks", *in*: *In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS'10). Society for Artificial Intelligence and Statistics*, 2010.

[GE17] Yonatan Geifman and Ran El-Yaniv, "Deep Active Learning over the Long Tail", *in*: *arXiv preprint arXiv:1711.00941* (2017).

[Geo+17] Dileep George *et al.*, "A generative vision model that trains with high data efficiency and breaks text-based CAPTCHAs", *in*: *Science* 358 (Oct. 2017), eaag2612.

[GIG17] Yarin Gal, Riashat Islam, and Zoubin Ghahramani, "Deep bayesian active learning with image data", *in*: *arXiv preprint arXiv:1703.02910* (2017).

[Gir+14] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", *in*: *CVPR*, CVPR '14, Washington, DC, USA: IEEE Computer Society, 2014, pp. 580–587.

[GK78] K. Chidananda Gowda and G. Krishna, "Agglomerative clustering using the concept of mutual nearest neighbourhood", *in*: *Pattern Recognition* 10 (1978), pp. 105–112.

[GLL18] Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V Le, "DropBlock: A regularization method for convolutional networks", *in*: *Advances in Neural Information Processing Systems 31*, ed. by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Curran Associates, Inc., 2018, pp. 10727–10737.

[Gon+14] Yunchao Gong, Liwei Wang, Ruiqi Guo, and Svetlana Lazebnik, "Multi-scale orderless pooling of deep convolutional activation features", *in*: *ECCV*, 2014.

[Goo+14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, "Generative Adversarial Nets", *in*: *NIPS*, ed. by Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Curran Associates, Inc., 2014, pp. 2672–2680.

[Gor+16a] Albert Gordo, Jon Almazan, Jerome Revaud, and Diane Larlus, "Deep Image Retrieval: Learning global representations for image search", *in*: *ECCV* (2016).

[Gor+16b] Albert Gordo, Jon Almazan, Jerome Revaud, and Diane Larlus, "End-to-end Learning of Deep Visual Representations for Image Retrieval", *in*: *arXiv preprint arXiv:1610.07940* (2016).

[Gor+17] Albert Gordo, Jon Almazan, Jerome Revaud, and Diane Larlus, "End-to-End Learning of Deep Visual Representations for Image Retrieval", *in*: *IJCV* 124.2 (Sept. 2017), pp. 237–254.

[Gra06] Leo Grady, "Random Walks for Image Segmentation", *in*: *IEEE Trans. PAMI* 28.11 (2006), pp. 1768–1783.

[GS18] Daniel Gissin and Shai Shalev-Shwartz, "Discriminative Active Learning", *in*: (2018).

[GSK18] Spyros Gidaris, Praveer Singh, and Nikos Komodakis, "Unsupervised Representation Learning by Predicting Image Rotations", *in*: *ICLR*, 2018.

[Guy+15] I. Guyon *et al.*, "Design of the 2015 ChaLearn AutoML challenge", *in*: *2015 International Joint Conference on Neural Networks (IJCNN)*, July 2015, pp. 1–8.

[GWW01] Y. Gdalyahu, D. Weinshall, and M. Werman, "Self-organization in vision: stochastic clustering for image segmentation, perceptual grouping, and image database organization", *in*: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 23.10 (Oct. 2001), pp. 1053–1074.

[HA14] Elad Hoffer and Nir Ailon, "Deep Metric Learning Using Triplet Network", *in*: Dec. 2014.

[HCL06] Raia Hadsell, Sumit Chopra, and Yann Lecun, "Dimensionality Reduction By Learning an Invariant Mapping", *in*: *Computer Vision and Pattern Recognition*, vol. 2, IEEE, 2006, pp. 1735–1742.

[He+15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification", *in*: *ICCV*, Washington, DC, USA: IEEE Computer Society, 2015, pp. 1026–1034.

[He+16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition", *in*: *CVPR*, 2016.

[He+17] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN", *in*: *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct. 2017, pp. 2980–2988.

[He+18] Yihui He, Ji Lin, Zhijian Liu, Hanrui Wang, Li-Jia Li, and Song Han, "AMC: AutoML for Model Compression and Acceleration on Mobile Devices", *in*: *The European Conference on Computer Vision (ECCV)*, Sept. 2018.

[HG17] Bharath Hariharan and Ross Girshick, "Low-Shot Visual Recognition by Shrinking and Hallucinating Features", *in*: *ICCV*, 2017.

[HGD18] Kaiming He, Ross B. Girshick, and Piotr Dollár, "Rethinking ImageNet Pre-training", *in*: *ArXiv* abs/1811.08883 (2018).

[Hin+12] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors", *in*: *ArXiv* abs/1207.0580 (2012).

[HOT06] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh, "A Fast Learning Algorithm for Deep Belief Nets", *in*: *Neural Comput.* 18.7 (July 2006), pp. 1527–1554.

[HS06] Geoffrey E. Hinton and R R. Salakhutdinov, "Reducing the dimensionality of data with neural networks", *in*: *Science* 313.5786 (2006), pp. 504–507.

[HS88] Chris Harris and Mike Stephens, "A combined corner and edge detector.", *in*: *Alvey vision conference*, vol. 15, 1988, p. 50.

[Hua+17] G. Huang, Z. Liu, L. v. d. Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks", *in*: *CVPR*, July 2017, pp. 2261–2269.

[Hub65] Charles H Hubbell, "An input-output approach to clique identification", *in*: *Sociometry* (1965).

[HW59] David H. Hubel and Torsten N. Wiesel, "Receptive Fields of Single Neurons in the Cat's Striate Cortex", *in*: *Journal of Physiology* 148 (1959), pp. 574–591.

[HW62] D. Hubel and T. Wiesel, "Receptive fields, binocular interaction, and functional architecture in the cat's visual cortex", *in*: *Journal of Physiology* 160 (1962), pp. 106–154.

[HW68] David H. Hubel and Torsten Nils Wiesel, "Receptive fields and functional architecture of monkey striate cortex.", *in*: *The Journal of physiology* 195 1 (1968), pp. 215–43.

[HZ03] Richard Hartley and Andrew Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed., New York, NY, USA: Cambridge University Press, 2003.

[HZ04] Richard Hartley and Andrew Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, 2004.

[Isc+17] Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, Teddy Furon, and Ondřej Chum, "Efficient Diffusion on Region Manifolds: Recovering Small Objects with Compact CNN Representations", *in*: *CVPR*, 2017.

[Isc+18a] Ahmet Iscen, Yannis Avrithis, Giorgos Tolias, Teddy Furon, and Ondřej Chum, "Fast Spectral Ranking for Similarity Search", *in*: *CVPR* (2018).

[Isc+18b] Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondřej Chum, "Mining on Manifolds: Metric Learning without Labels", *in*: *CVPR* (2018).

[Isc+19a] A. Iscen, G. Tolias, Y. Avrithis, and O. Chum, "Label propagation for Deep Semi-supervised Learning", *in*: 2019.

[Isc+19b] Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondřej Chum, "Label propagation for deep semi-supervised learning", *in*: *CVPR*, 2019.

[Jad+15] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and koray kavukcuoglu koray, "Spatial Transformer Networks", *in*: *Advances in Neural Information Processing Systems 28*, ed. by C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Curran Associates, Inc., 2015, pp. 2017–2025.

[JAG17] Albert Jimenez, Jose M Alvarez, and Xavier Giro-i-Nieto, "Class-Weighted Convolutional Features for Visual Instance Search", *in*: *BMVC* (2017).

[JB09] Alexis Joly and Olivier Buisson, "Logo Retrieval with A Contrario Visual Query Expansion", *in*: *ACM Multimedia*, Oct. 2009.

[JC12] Hervé Jégou and Ondřej Chum, "Negative evidences and co-occurrences in image retrieval: The benefit of PCA and whitening", *in*: *ECCV*, Oct. 2012.

[JDS08] Hervé Jégou, Matthijs Douze, and Cordelia Schmid, "Hamming embedding and weak geometric consistency for large scale image search", *in*: *ECCV*, Oct. 2008.

[JDS10] Hervé Jégou, Matthijs Douze, and Cordelia Schmid, "Improving bag-of-features for large scale image search", *in*: *IJCV* 87.3 (Feb. 2010), pp. 316–336.

[Jég+10] Hervé Jégou, Matthijs Douze, Cordelia Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation", *in*: *CVPR*, June 2010.

[Jeo+17] Dong-Ju Jeong, Sungkwon Choo, Wonkyo Seo, and Nam Ik Cho, "Regional deep feature aggregation for image retrieval", *in*: *ICASSP*, 2017.

[JMF99] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data Clustering: A Review", *in*: *ACM Comput. Surv.* 31.3 (Sept. 1999), pp. 264–323.

[JPP09] A. J. Joshi, F. Porikli, and N. Papanikolopoulos, "Multi-class active learning for image classification", *in*: *CVPR*, June 2009, pp. 2372–2379.

[Kat53] Leo Katz, "A New Status Index Derived From Sociometric Analysis", *in*: *Psychometrika* 18.1 (1953), pp. 39–43.

[KDB09] Peter Kontschieder, Michael Donoser, and Horst Bischof, "Beyond pairwise shape similarity analysis", *in*: *ACCV*, 2009.

[KF17] Angelos Katharopoulos and François Fleuret, "Biased Importance Sampling for Deep Neural Network Training", *in*: *arXiv preprint arXiv:1706.00043* (2017).

[Kim+17] Seungryong Kim, Dongbo Min, Bumsub Ham, Sangryul Jeon, Stephen Lin, and Kwanghoon Sohn, "Fcss: Fully convolutional self-similarity for dense semantic correspondence", *in*: *CVPR*, 2017.

[KLL08] Tae Hoon Kim, Kyoung Mu Lee, and Sang Uk Lee, "Generative Image Segmentation Using Random Walks with Restart", *in*: *ECCV*, Springer, 2008, pp. 264–275.

[KMO15] Yannis Kalantidis, Clayton Mellina, and Simon Osindero, "Cross-dimensional weighting for aggregated deep convolutional features", *in*: *arXiv*, 2015.

[KMO16] Yannis Kalantidis, Clayton Mellina, and Simon Osindero, "Cross-Dimensional Weighting for Aggregated Deep Convolutional Features", *in*: *ECCVW*, ed. by Gang Hua and Hervé Jégou, Cham: Springer International Publishing, 2016, pp. 685–701.

[Koc15] Gregory R. Koch, "Siamese Neural Networks for One-Shot Image Recognition", *in*: 2015.

[Kri09] Alex Krizhevsky, "Learning Multiple Layers of Features from Tiny Images", *in*: 2009.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton, "Imagenet Classification with Deep Convolutional Neural Networks", *in*: *NIPS*, 2012.

[KSJ14] Angjoo Kanazawa, Abhishek Sharma, and David W. Jacobs, *Locally Scale-Invariant Convolutional Neural Networks*, 2014.

[KSP10] Jan Knopp, Josef Sivic, and Tomas Pajdla, "Avoiding Confusing Features in Place Recognition", *in*: *ECCV*, 2010.

[KT09] Gunhee Kim and Antonio Torralba, "Unsupervised detection of regions of interest using iterative link analysis", *in*: *NIPS*, 2009.

[Kur91] Takio Kurita, "An efficient agglomerative clustering algorithm using a heap", *in*: *Pattern Recognition* (Dec. 1991), pp. 205–209.

[KW52] Jürgen Kiefer and Jacob Wolfowitz, "Stochastic Estimation of the Maximum of a Regression Function", *in*: 1952.

[Kwa+15] Suha Kwak, Minsu Cho, Ivan Laptev, Jean Ponce, and Cordelia Schmid, "Unsupervised object discovery and tracking in video collections", *in*: *CVPR*, 2015.

[KY18] Jaeyoon Kim and Sung-Eui Yoon, "Regional Attention Based Deep Feature for Image Retrieval", *in*: *BMVC* (2018).

[LA16a] Samuli Laine and Timo Aila, "Temporal ensembling for semi-supervised learning", *in*: *arXiv preprint arXiv:1610.02242* (2016).

[LA16b] Samuli Laine and Timo Aila, "Temporal ensembling for semi-supervised learning", *in*: *arXiv preprint arXiv:1610.02242* (2016).

[LA17] Samuli Laine and Timo Aila, "Temporal ensembling for semi-supervised learning", *in*: *ICLR*, 2017.

[Lak+11] Brenden M. Lake, Ruslan Salakhutdinov, Jason Gross, and Joshua B. Tenenbaum, "One shot learning of simple visual concepts", *in*: *CogSci*, 2011.

[LeC+89] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation Applied to Handwritten Zip Code Recognition", *in*: *Neural Comput.* 1.4 (Dec. 1989), pp. 541–551.

[LeC+98] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, *et al.*, "Gradient-based learning applied to document recognition", *in*: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[Lec+98] Yann Lecun, Leon Bottou, Y Bengio, and Patrick Haffner, "Gradient-Based Learning Applied to Document Recognition", *in*: *Proceedings of the IEEE* 86 (Dec. 1998), pp. 2278–2324.

[Lee13] Dong-Hyun Lee, "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks.", *in*: *ICMLW*, 2013.

[LFP06] Li Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories", *in*: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.4 (Apr. 2006), pp. 594–611.

[LG13] X. Li and Y. Guo, "Adaptive Active Learning for Image Classification", *in*: *CVPR*, June 2013, pp. 859–866.

[LG97] Tony Lindeberg and Jonas Gårding, "Shape-adapted smoothing in estimation of 3-D shape cues from affine deformations of local 2-D brightness structure. Image Vis. Comput. 15(6), 415-434", *in*: *Image and Vision Computing* 15 (Apr. 1997), pp. 415–434.

[LH17] Ilya Loshchilov and Frank Hutter, "SGDR: Stochastic Gradient Descent with Warm Restarts", *in*: *ICLR*, 2017.

[LHB04] Yann LeCun, Fu Jie Huang, and Léon Bottou, "Learning methods for generic object recognition with invariance to pose and lighting", English (US), *in*: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, 2004.

[Li+17] Yuncheng Li, Jianchao Yang, Yale Song, Liangliang Cao, Jiebo Luo, and Li-Jia Li, "Learning from Noisy Labels with Distillation", *in*: *ICCV* (2017), pp. 1928–1936.

[Li+19] Yanchao Li, Yong li Wang, Dong-Jun Yu, Ye Ning, Peng Hu, and Ruxin Zhao, "ASCENT: Active Supervision for Semi-supervised Learning", *in*: *IEEE Transactions on Knowledge and Data Engineering* (2019).

[Lia+16] Renjie Liao, Alex Schwing, Richard Zemel, and Raquel Urtasun, "Learning Deep Parsimonious Representations", *in*: *NIPS*, ed. by D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Curran Associates, Inc., 2016, pp. 5076–5084.

[Lin+14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick, "Microsoft coco: Common objects in context", *in*: *ECCV*, 2014.

[Lin98] Tony Lindeberg, "Feature detection with automatic scale selection", *in*: *IJCV* 30.2 (1998), pp. 77–116.

[LK17] Zakaria Laskar and Juho Kannala, "Context Aware Query Image Representation for Particular Object Retrieval", *in*: *Scandinavian Conference on Image Analysis*, 2017.

[LLH15] Xinchao Li, Martha Larson, and Alan Hanjalic, "Pairwise geometric matching for large-scale object retrieval", *in*: *CVPR* (2015), pp. 5153–5161.

[Llo82] S. Lloyd, "Least squares quantization in PCM", *in*: *IEEE Transactions on Information Theory* 28.2 (Mar. 1982), pp. 129–137.

[LNH09] CH. Lampert, H. Nickisch, and S. Harmeling, "Learning To Detect Unseen Object Classes by Between-Class Attribute Transfer", *in*: *CVPR 2009*, Max-Planck-Gesellschaft, Piscataway, NJ, USA: IEEE Service Center, June 2009, pp. 951–958.

[Lon+08] Jun Long, Jianping Yin, Wentao Zhao, and En Zhu, "Graph-Based Active Learning Based on Label Propagation", *in*: *International Conference on Modeling Decisions for Artificial Intelligence*, Springer, 2008, pp. 179–190.

[Low99] David G Lowe, "Object recognition from local scale-invariant features", *in*: *ICCV*, 1999, pp. 1150–1157.

[LS18] Zhengqi Li and Noah Snavely, "MegaDepth: Learning single-view depth prediction from internet photos", *in*: *CVPR* (2018).

[LST13] Brenden M Lake, Ruslan R Salakhutdinov, and Josh Tenenbaum, "One-shot learning by inverting a compositional causal process", *in*: *Advances in Neural Information Processing Systems 26*, ed. by C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Curran Associates, Inc., 2013, pp. 2526–2534.

[LST15] Brenden M. Lake, Ruslan Salakhutdinov, and Joshua B. Tenenbaum, "Human-level concept learning through probabilistic program induction", *in*: *Science* 350.6266 (2015), pp. 1332–1338.

[LZD14] Jonathan L Long, Ning Zhang, and Trevor Darrell, "Do Convnets Learn Correspondence?", *in*: *NIPS*, 2014.

[Mah+18] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten, "Exploring the Limits of Weakly Supervised Pretraining", *in*: Cham: Springer International Publishing, 2018, pp. 185–201.

[Mat+02] Jiri Matas, Ondřej Chum, U. Martin, and T. Pajdla, "Robust wide baseline stereo from maximally stable extremal regions", *in*: *BMVC*, Sept. 2002, pp. 384–393.

[Men+12] Thomas Mensink, Jakob Verbeek, Florent Perronnin, and Gabriela Csurka, "Metric Learning for Large Scale Image Classification: Generalizing to New Classes at Near-Zero Cost", *in*: *ECCV 2012 - 12th European Conference on Computer Vision*, ed. by Andrew Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, vol. 7573, Lecture Notes in Computer Science, Florence, Italy: Springer, Oct. 2012, pp. 488–501.

[Miy+16] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, Ken Nakae, and Shin Ishii, "Distributional Smoothing with Virtual Adversarial Training", *in*: *ICLR*, 2016.

[Miy+18] Takeru Miyato, Shin-ichi Maeda, Shin Ishii, and Masanori Koyama, "Virtual adversarial training: a regularization method for supervised and semi-supervised learning.", *in*: (2018).

[MM07] Krystian Mikolajczyk and Jiri Matas, "Improving Descriptors for Fast Tree Matching By Optimal Linear Projection", *in*: *CVPR*, 2007.

[MMK02] Ion Muslea, Steven Minton, and Craig A Knoblock, "Active+ semi-supervised learning = robust multi-view learning", *in*: *ICML*, 2002.

[MMV00] E. G. Miller, N. E. Matsakis, and P. A. Viola, "Learning from one example through shared densities on transforms", *in*: *CVPR*, vol. 1, June 2000, 464–471 vol.1.

[MN98] M McCallum and K Nigam, "Employing EM in pool-based active learning for text classification, 1998", *in*: *ICML*, 1998.

[Moh+16] Eva Mohedano, Amaia Salvador, Kevin McGuinness, X. Giró-i-Nieto, N. O'Connor, and F. Marqués, "Bags of Local Convolutional Features for Scalable Instance Search", *in*: *ACM International Conference on Multimedia Retrieval (ICMR)*, ACM, New York City, NY; USA: ACM, June 2016.

[Moh+17] Eva Mohedano, Kevin McGuinness, Xavier Giro-i-Nieto, and Noel E O'Connor, "Saliency Weighted Convolutional Features for Instance Search", *in*: *arXiv* (2017).

[Mor80] Hans Peter Moravec, "Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover", AAI8024717, PhD thesis, Stanford, CA, USA, 1980.

[MP88] Marvin L. Minsky and Seymour A. Papert, *Perceptrons: Expanded Edition*, Cambridge, MA, USA: MIT Press, 1988.

[MRS08] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze, *Introduction to Information Retrieval*, Cambridge University Press, 2008.

[MS01] Krystian Mikolajczyk and Cordelia Schmid, "Indexing based on scale invariant interest points", *in*: *ICCV*, vol. 1, July 2001, pp. 525–531.

[MS02] K. Mikolajczyk and C. Schmid, "An Affine Invariant Interest Point Detector", *in*: *Proceedings of the 7th European Conference on Computer Vision-Part I*, ECCV '02, London, UK, UK: Springer-Verlag, 2002, pp. 128–142.

[MS04] Krystian Mikolajczyk and Cordelia Schmid, "Scale and affine invariant interest point detectors", *in*: *IJCV* 60.1 (2004), pp. 63–86.

[MT18] Christoph Mayer and Radu Timofte, "Adversarial Sampling for Active Learning", *in*: *arXiv preprint arXiv:1808.06671* (2018).

[MV15] Aravindh Mahendran and Andrea Vedaldi, "Understanding deep image representations by inverting them", *in*: *CVPR* (2015), pp. 5188–5196.

[Mya+02] Darren R. Myatt, Philip H. S. Torr, Slawomir J. Nasuto, J. Mark Bishop, and Rachel Craddock, "NAPSAC: High Noise, High Dimensional Robust Estimation - it's in the Bag", *in*: *BMVC*, 2002.

[Nad+05] Boaz Nadler, Stephane Lafon, Ronald R Coifman, and Ioannis G Kevrekidis, "Diffusion Maps, Spectral Clustering and Eigenfunctions of Fokker-Planck Operators", *in*: *NIPS* (2005).

[Net+11] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Ng, "Reading Digits in Natural Images with Unsupervised Feature Learning", *in*: *NIPS Workshop on Deep Learning and Unsupervised Feature Learning* (Jan. 2011).

[New10] M.E.J Newman, *Networks: an introduction*, Oxford University Press, Oxford, 2010.

[NF16] Mehdi Noroozi and Paolo Favaro, "Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles", *in*: *ECCV* (Mar. 2016).

[NH10] Vinod Nair and Geoffrey E. Hinton, "Rectified Linear Units Improve Restricted Boltzmann Machines", *in*: *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, Haifa, Israel: Omnipress, 2010, pp. 807–814.

[NJW+02] Andrew Y Ng, Michael I Jordan, Yair Weiss, *et al.*, "On spectral clustering: Analysis and an algorithm", *in*: *NIPS*, 2002.

[Noh+16] Hyeonwoo Noh, Andre Araujo, Jack Sim, Tobias Weyand, and Bohyung Han, "Large-Scale Image Retrieval with Attentive Deep Local Features", *in*: *arXiv*, 2016.

[Noh+17] Hyeonwoo Noh, Andre Araujo, Jack Sim, Tobias Weyand, and Bohyung Han, "Largescale image retrieval with attentive deep local features", *in*: *ICCV*, 2017.

[NS06] David Nistér and Henrik Stewénius, "Scalable Recognition with a Vocabulary Tree", *in*: *CVPR*, June 2006, pp. 2161–2168.

[NW06] Jorge Nocedal and Stephen Wright, *Numerical optimization*, Springer, 2006.

[Ome+08] Dušan Omercevic, Roland Perko, Alireza Tavakoli Targhi, Jan-Olof Eklundh, and Aleš Leonardis, "Vegetation segmentation for boosting performance of MSER feature detector", *in*: *Computer Vision Winter Workshop*, 2008.

[Oqu+15] Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic, "Is Object Localization for Free?–Weakly-Supervised Learning with Convolutional Neural Networks", *in*: *CVPR*, 2015.

[OT01] Aude Oliva and Antonio Torralba, "Modeling the shape of the scene: a holistic representation of the spatial envelope", *in*: *IJCV* 42.3 (2001), pp. 145–175.

[OT06] Aude Oliva and Antonio Torralba, "Building the gist of a scene: The role of global image features in recognition", *in*: *Progress in brain research* 155 (2006), pp. 23–36.

[Pag+99] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd, "The PageRank citation ranking: bringing order to the web.", *in*: (1999).

[Pan+18] Shanmin Pang, Jin Ma, Jianru Xue, Jihua Zhu, and Vicente Ordonez, "Image Retrieval using Heat Diffusion for Deep Feature Aggregation", *in*: *arXiv preprint arXiv:1805.08587* (2018).

[Par+12] Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar, "Cats and Dogs", *in*: *CVPR*, 2012.

[Pat+16] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros, "Context Encoders: Feature Learning by Inpainting", *in*: *CVPR*, 2016.

[PCM09] Michal Perdoch, Ondřej Chum, and Jiri Matas, "Efficient Representation of Local Geometry for Large Scale Object Retrieval", *in*: *CVPR*, June 2009.

[PD07] Florent Perronnin and Christopher R. Dance, "Fisher Kernels on Visual Vocabularies for Image Categorization", *in*: *CVPR*, June 2007.

[Phi+07] James Philbin, Ondřej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman, "Object retrieval with large vocabularies and fast spatial matching", *in*: *CVPR*, June 2007.

[Phi+08] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Lost in Quantization: Improving Particular Object Retrieval in Large Scale Image Databases", *in*: *CVPR*, June 2008.

[Qia+18] Siyuan Qiao, Wei Shen, Zhishuai Zhang, Bo Wang, and Alan Yuille, "Deep co-training for semi-supervised image recognition", *in*: (2018).

[Qin+11] Danfeng Qin, Stephan Gammeter, Lukas Bossard, Till Quack, and Luc Van Gool, "Hello Neighbor: Accurate Object Retrieval with k-reciprocal nearest neighbors", *in*: *CVPR*, 2011.

[Rad+18] Filip Radenović, Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondřej Chum, "Revisiting Oxford and Paris: Large-Scale Image Retrieval Benchmarking", *in*: *CVPR*, 2018.

[Ran+07] M. Ranzato, F. J. Huang, Y. Boureau, and Y. LeCun, "Unsupervised Learning of Invariant Feature Hierarchies with Applications to Object Recognition", *in*: *2007 IEEE Conference on Computer Vision and Pattern Recognition*, June 2007, pp. 1–8.

[Ras+15] Antti Rasmus, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko, "Semi-supervised learning with ladder networks", *in*: *NIPS*, 2015.

[RAS17] Ignacio Rocco, Relja Arandjelović, and Josef Sivic, "Convolutional neural network architecture for geometric matching", *in*: *CVPR*, 2017.

[RAS18] Ignacio Rocco, Relja Arandjelovic, and Josef Sivic, "End-to-end weakly-supervised semantic alignment", *in*: *CVPR*, 2018.

[Raz+16] Ali S Razavian, Josephine Sullivan, Stefan Carlsson, and Atsuto Maki, "Visual instance retrieval with deep convolutional networks", *in*: *ITE Transactions on Media Technology and Applications* 4 (2016), pp. 251–258.

[RD01] Matthew Richardson and Pedro M Domingos, "The Intelligent surfer: Probabilistic Combination of Link and Content Information in PageRank.", *in*: *NIPS*, 2001.

[Reb+19] Sylvestre-Alvise Rebuffi, Sebastien Ehrhardt, Kai Han, Andrea Vedaldi, and Andrew Zisserman, "Semi-Supervised Learning with Scarce Annotations", *in*: *arXiv preprint arXiv:1905.08845* (2019).

[Ren+15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks", *in*: *NIPS*, 2015, pp. 91–99.

[RHW86] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams, "Learning Internal Representations by Error Propagation", *in*: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*, ed. by David E. Rumelhart and James L. Mcclelland, Cambridge, MA: MIT Press, 1986, pp. 318–362.

[RL17] Sachin Ravi and Hugo Larochelle, "Optimization as a model for few-shot learning", *in*: *ICLR*, 2017.

[Roc+18] Ignacio Rocco, Mircea Cimpoi, Relja Arandjelović, Akihiko Torii, Tomas Pajdla, and Josef Sivic, "Neighbourhood Consensus Networks", *in*: *NIPS*, Montréal, Canada, Dec. 2018.

[Ros58] F. Rosenblatt, "The Perceptron: A Probabilistic Model for Information Storage and Organization in The Brain", *in*: *Psychological Review* (1958), pp. 65–386.

[RTC16] Filip Radenović, Giorgos Tolias, and Ondřej Chum, "CNN Image Retrieval Learns from BoW: Unsupervised Fine-Tuning with Hard Examples", *in*: *ECCV*, 2016.

[RTC18] Filip Radenović, Giorgos Tolias, and Ondřej Chum, "Fine-tuning CNN Image Retrieval with No Human Annotation", *in*: *IEEE Trans. PAMI* (2018).

[Rub+13] Michael Rubinstein, Armand Joulin, Johannes Kopf, and Ce Liu, "Unsupervised joint object discovery and segmentation in internet images", *in*: *CVPR*, 2013.

[Rus+06] Bryan C. Russell, William T. Freeman, Alexei A. Efros, Josef Sivic, and Andrew Zisserman, "Discovering object categories in image collections", *in*: *CVPR*, 2006.

[Rus+15] Olga Russakovsky *et al.*, "ImageNet Large Scale Visual Recognition Challenge", *in*: *IJCV* 115.3 (2015), pp. 211–252.

[SAC19] Oriane Siméoni, Yannis Avrithis, and Ondřej Chum, "Local Features and Visual Words Emerge in Activations", *in*: *CVPR*, Long Beach, CA, US, June 2019.

[SAJ15] Miaojing Shi, Yannis Avrithis, and Herve Jegou, "Early Burst Detection for Memory-Efficient Image Retrieval", *in*: *CVPR*, 2015.

[Sal+16] Amaia Salvador, Xavier Giró-i-Nieto, Ferran Marqués, and Shin'ichi Satoh, "Faster r-cnn features for instance search", *in*: *CVPRW*, 2016.

[SC04] John Shawe-Taylor and Nello Cristianini, *Kernel Methods for Pattern Analysis*, New York, NY, USA: Cambridge University Press, 2004.

[Sch+15] Johannes L. Schönberger, Filip Radenović, Ondřej Chum, and Jan-Michael Frahm, "From single image query to detailed 3D reconstruction", *in*: *CVPR* (2015), pp. 5126–5134.

[SED19] Samarth Sinha, Sayna Ebrahimi, and Trevor Darrell, "Variational Adversarial Active Learning", *in*: *arXiv preprint arXiv:1904.00370* (2019).

[See01] Matthias Seeger, *Learning with labeled and unlabeled data*, tech. rep., University of Edinburgh, 2001.

[See49] John R Seeley, "The Net of Reciprocal Influence. A Problem in Treating Sociometric Data", *in*: *Canadian Journal of Experimental Psychology* 3 (1949), p. 234.

[Sel+17] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra, "Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization", *in*: *ICCV*, Oct. 2017, pp. 618–626.

[Ser+13a] Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus, and Yann Lecun, "OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks", *in*: *International Conference on Learning Representations (ICLR) (Banff)* (Dec. 2013).

[Ser+13b] Pierre Sermanet, Koray Kavukcuoglu, Soumith Chintala, and Yann Lecun, "Pedestrian Detection with Unsupervised Multi-stage Feature Learning", *in*: *Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR 13, Washington, DC, USA: IEEE Computer Society, 2013, pp. 3626–3633.

[Set09] Burr Settles, *Active learning literature survey*, tech. rep., University of Wisconsin-Madison Department of Computer Sciences, 2009.

[SF16a] Johannes Lutz Schonberger and Jan-Michael Frahm, "Structure-from-motion revisited", *in*: *CVPR*, 2016.

[SF16b] Miaojing Shi and Vittorio Ferrari, "Weakly Supervised Object Localization Using Size Estimates", *in*: *ECCV*, 2016.

[SGM19] Emma Strubell, Ananya Ganesh, and Andrew McCallum, "Energy and Policy Considerations for Deep Learning in NLP", *in*: *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28-August 2, 2019, Volume 1: Long Papers*, 2019, pp. 3645–3650.

[Shi+18] Weiwei Shi, Yihong Gong, Chris Ding, Zhiheng Ma, Xiaoyu Tao, and Nanning Zheng, "Transductive semi-supervised deep learning using min-max features", *in*: *ECCV* (2018).

[Sim+14] Edgar Simo-Serra, Eduard Trulls, Luis Fc, Iasonas Kokkinos, and Francesc Moreno-Noguer, "Fracking Deep Convolutional Image Descriptors", *in*: (Dec. 2014).

[Sim+18] Oriane Siméoni, Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondřej Chum, "Unsupervised object discovery for instance recognition", *in*: *WACV*, 2018.

[Sim+19] Oriane Siméoni, Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondřej Chum, "Graph-based Particular Object Discovery", *in*: *Machine Vision and Applications (MVA)* 30.2 (Mar. 2019), pp. 243–254.

[Sim+20] Oriane Siméoni, Mateusz Budnik, Yannis Avrithis, and Guillaume Gravier, "Rethinking deep active learning: Using unlabeled data at model training", *in*: *ICPR*, 2020.

[Siv+05] Josef Sivic, Bryan C. Russell, Alexei A. Efros, Andrew Zisserman, and William T. Freeman, "Discovering object categories in image collections", *in*: *ICCV*, 2005.

[SJT16] Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdiz, "Regularization with stochastic transformations and perturbations for deep semi-supervised learning", *in*: *NIPS*, 2016.

[SM13] L. Sifre and S. Mallat, "Rotation, Scaling and Deformation Invariant Scattering for Texture Discrimination", *in*: *CVPR*, June 2013, pp. 1233–1240.

[SM83] Gerard Salton and Michael J. McGill, *Introduction to modern information retrieval*, McGraw-Hill, 1983.

[SM97] J. Shi and Jitendra Malik, "Normalized cuts and image segmentation", *in*: *CVPR*, June 1997.

[Soc+13] Richard Socher, Milind Ganjoo, Christopher D Manning, and Andrew Ng, "Zero-Shot Learning Through Cross-Modal Transfer", *in*: *Advances in Neural Information Processing Systems 26*, ed. by C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Curran Associates, Inc., 2013, pp. 935–943.

[Son+17] Jingkuan Song, Tao He, Lianli Gao, Xing Xu, and Heng Tao Shen, "Deep Region Hashing for Efficient Large-scale Instance Search from Images", *in*: *arXiv*, 2017.

[Spr+14] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin A. Riedmiller, "Striving for Simplicity: The All Convolutional Net", *in*: *arXiv* abs/1412.6806 (2014).

[Spr16] Jost Tobias Springenberg, "Unsupervised and Semi-supervised Learning with Categorical Generative Adversarial Networks", *in*: *ICLR* (2016).

[Sri+14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting", *in*: *Journal of Machine Learning Research* 15 (2014), pp. 1929–1958.

[SS02] B. Schölkopf and A. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond*, MIT Press, Cambridge, MA, 2002.

[SS18] Ozan Sener and Silvio Savarese, "Active learning for convolutional neural networks: A core-set approach", *in*: *arXiv preprint arXiv:1708.00489* (2018).

[SSP03] Patrice Y. Simard, David Steinkraus, and John C. Platt, "Best practices for convolutional neural networks applied to visual document analysis", *in*: *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.* (2003), pp. 958–963.

[SSZ17] Jake Snell, Kevin Swersky, and Richard Zemel, "Prototypical Networks for Few-shot Learning", *in*: *Advances in Neural Information Processing Systems 30*, ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Curran Associates, Inc., 2017, pp. 4077–4087.

[Sun+17] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta, "Revisiting Unreasonable Effectiveness of Data in Deep Learning Era", *in*: *2017 IEEE International Conference on Computer Vision (ICCV)* (2017), pp. 843–852.

[SZ03] Josef Sivic and Andrew Zisserman, "Video Google: A Text Retrieval Approach to Object Matching in Videos", *in*: *ICCV*, 2003.

[SZ14] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition", *in*: *ICLR* (2014).

[Sze+14] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich, "Going deeper with convolutions", *in*: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2014), pp. 1–9.

[Sze+16] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna, "Rethinking the Inception Architecture for Computer Vision", *in*: *CVPR*, 2016.

[Sze+17] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A. Alemi, "Inception-v4, inception-ResNet and the Impact of Residual Connections on Learning", *in*: *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, San Francisco, California, USA: AAAI Press, 2017, pp. 4278–4284.

[TA11] Giorgos Tolias and Yannis Avrithis, "Speeded-Up, Relaxed Spatial Matching", *in*: *ICCV*, 2011.

[TAJ13] Giorgios Tolias, Yannis Avrithis, and Hervé Jégou, "To aggregate or not to aggregate: Selective match kernels for image search", *in*: *ICCV*, Dec. 2013.

[TAJ16] Giorgos Tolias, Yannis Avrithis, and Hervé Jégou, "Image search with selective match kernels: aggregation across single and multiple images", *in*: *IJCV* (2016).

[TC09] Tatiana Tommasi and Barbara Caputo, "The More You Know, the Less You Learn: From Knowledge Transfer to One-shot Learning of Object Categories", *in*: *BMVC*, 2009.

[Tei+19] Marvin Teichmann, André Araujo, Menglong Zhu, and Jack Sim, "Detect-to-Retrieve: Efficient Regional Aggregation for Image Search", *in*: *CVPR*, 2019.

[TFW08] A. Torralba, R. Fergus, and Y. Weiss, "Small codes and large databases for recognition", *in*: *CVPR*, June 2008.

[TFW17] Yurun Tian, Bin Fan, and Fuchao Wu, "L2-Net: Deep Learning of Discriminative Patch Descriptor in Euclidean Space", *in*: *CVPR*, July 2017, pp. 6128–6136.

[Thr95] Sebastian Thrun, "Is Learning The n-th Thing Any Easier Than Learning The First?", *in*: *NIPS*, 1995.

[TJ14] Giorgos Tolias and Herve Jegou, "Visual Query Expansion with or Without Geometry: Refining Local Descriptors By Feature Aggregation", *in*: *Pattern Recognition* (2014).

[TKA12] G. Tolias, Y. Kalantidis, and Y. Avrithis, "SymCity: Feature Selection by Symmetry for Large Scale Image Retrieval", *in*: *ACM Multimedia*, 2012.

[TL09] P. Turcot and D. G. Lowe, "Better matching with fewer features: The selection of useful features in large database recognition problems", *in*: *ICCVW*, 2009.

[Tom+15] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler, "Efficient object localization using convolutional networks", *in*: *CVPR*, 2015, pp. 648–656.

[TP98] Sebastian Thrun and Lorien Pratt, eds., *Learning to Learn*, Norwell, MA, USA: Kluwer Academic Publishers, 1998.

[TSJ16] Giorgos Tolias, Ronan Sicre, and Hervé Jégou, "Particular object retrieval with integral max-pooling of CNN activations", *in*: *ICLR* (2016).

[TV17] Antti Tarvainen and Harri Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results", *in*: *NIPS*, 2017.

[TZ00] P. H. S. Torr and A. Zisserman, "MLESAC: A New Robust Estimator with Application to Estimating Image Geometry", *in*: *Computer Vision and Image Understanding* 78 (2000), pp. 138–156.

[UB09] Rafael Uetz and Sven Behnke, "Large-scale object recognition with CUDA-accelerated hierarchical neural networks", *in*: *2009 IEEE International Conference on Intelligent Computing and Intelligent Systems* 1 (2009), pp. 536–541.

[Ver+19] Vikas Verma, Alex Lamb, Juho Kannala, Yoshua Bengio, and David Lopez-Paz, "Interpolation Consistency Training for Semi-Supervised Learning", *in*: *arXiv preprint arXiv:1903.03825* (2019).

[VF08] A. Vedaldi and B. Fulkerson, *VLFeat: An Open and Portable Library of Computer Vision Algorithms*, http://www.vlfeat.org/, 2008.

[Vig09] Sebastiano Vigna, "Spectral Ranking", *in*: *arXiv preprint arXiv:0912.0238* (2009).

[Vin+08] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol, "Extracting and composing robust features with denoising autoencoders.", *in*: *ICML*, 2008.

[Vin+16] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, koray kavukcuoglu koray, and Daan Wierstra, "Matching Networks for One Shot Learning", *in*: *Advances in Neural Information Processing Systems 29*, ed. by D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Curran Associates, Inc., 2016, pp. 3630–3638.

[Wan+14a] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jinbin Wang, James Philbin, Bo Chen, and Ying Wu, "Learning Fine-Grained Image Similarity with Deep Ranking", *in*: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (Apr. 2014).

[Wan+14b] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu, "Learning fine-grained image similarity with deep ranking", *in*: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1386–1393.

[Wan+15] Lijun Wang, Wanli Ouyang, Xiaogang Wang, and Huchuan Lu, "Visual Tracking with Fully Convolutional Networks", *in*: *ICCV*, ICCV '15, Washington, DC, USA: IEEE Computer Society, 2015, pp. 3119–3127.

[Wan+17] Keze Wang, Dongyu Zhang, Ya Li, Ruimao Zhang, and Liang Lin, "Cost-effective active learning for deep image classification", *in*: *IEEE Trans. CSVT* 27.12 (2017), pp. 2591–2600.

[Wan+18] Yu-Xiong Wang, Ross Girshick, Martial Hebert, and Bharath Hariharan, "Low-Shot learning from imaginary data", *in*: *CVPR* (2018).

[WG15] Xiaolong Wang and Abhinav Gupta, "Unsupervised Learning of Visual Representations Using Videos", *in*: *ICCV*, 2015.

[WJ15] Shuang Wang and Shuqiang Jiang, "INSTRE: a new benchmark for instance-level object retrieval and recognition", *in*: *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 11 (2015), p. 37.

[Won+18] Catherine Wong, Neil Houlsby, Yifeng Lu, and Andrea Gesmundo, "Transfer Learning with Neural AutoML", *in*: *Advances in Neural Information Processing Systems 31*, ed. by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Curran Associates, Inc., 2018, pp. 8356–8365.

[WRC08] Jason Weston, Frédéric Ratle, and Ronan Collobert, "Deep Learning via Semi-supervised Embedding", *in*: *ICML*, Helsinki, Finland: ACM, 2008, pp. 1168–1175.

[WY15] A. Wong and A. Yuille, "One Shot Learning via Compositions of Meaningful Patches", *in*: *ICCV*, Dec. 2015, pp. 1197–1205.

[XGF16] Junyuan Xie, Ross Girshick, and Ali Farhadi, "Unsupervised Deep Embedding for Clustering Analysis", *in*: *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML, New York, NY, USA: JMLR.org, 2016, pp. 478–487.

[Xin+02] Eric P. Xing, Andrew Y. Ng, Michael I. Jordan, and Stuart Russell, "Distance Metric Learning, with Application to Clustering with Side-information", *in*: *Proceedings of the 15th International Conference on Neural Information Processing Systems*, NIPS'02, Cambridge, MA, USA: MIT Press, 2002, pp. 521–528.

[Yan+15a] Weixin Yang, Lianwen Jin, Dacheng Tao, Zecheng Xie, and Ziyong Feng, "Dropsample: A New Training Method to Enhance Deep Convolutional Neural Networks for Large-Scale Unconstrained Handwritten Chinese Character Recognition", *in*: *arXiv preprint arXiv:1505.05354* (2015).

[Yan+15b] Yi Yang, Zhigang Ma, Feiping Nie, Xiaojun Chang, and Alexander G. Hauptmann, "Multi-Class Active Learning by Uncertainty Sampling with Diversity Maximization", *in*: *IJCV* 113.2 (June 2015), pp. 113–127.

[Yan+16] Jianwei Yang, Devi Parikh, Dhruv Batra, and Virginia Tech, "Joint Unsupervised Learning of Deep Representations and Image Clusters", *in*: *CVPR*, 2016.

[Yi+16] Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, and Pascal Fua, "Lift: Learned invariant feature transform", *in*: *ECCV*, 2016.

[ZCJ04] Zhi-Hua Zhou, Ke-Jia Chen, and Yuan Jiang, "Exploiting Unlabeled Data in Content-Based Image Retrieval", *in*: 2004.

[ZF14] Matthew D. Zeiler and Rob Fergus, "Visualizing and understanding convolutional networks", English (US), *in*: *Computer Vision, ECCV 2014 - 13th European Conference, Proceedings*, vol. 8689 LNCS, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) PART 1, Springer Verlag, 2014, pp. 818–833.

[ZG02] Xiaojin Zhu and Zoubin Ghahramani, *Learning From Labeled and Unlabeled Data with Label Propagation*, tech. rep., 2002.

[ZG09] Xiaojin Zhu and Andrew B. Goldberg, "Introduction to Semi-Supervised Learning", *in*: *Synthesis Lectures on Artificial Intelligence and Machine Learning* 3.1 (2009), pp. 1–130.

[ZGL03] Xiaojin Zhu, Zoubin Ghahramani, and John Lafferty, "Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions", *in*: *ICML*, 2003.

[Zha+12] Shaoting Zhang, Ming Yang, Timothee Cour, Kai Yu, and Dimitris N Metaxas, "Query specific fusion for image retrieval", *in*: *ECCV*, 2012.

[Zhe+16] Liang Zheng, Shengjin Wang, Jingdong Wang, and Qi Tian, "Accurate image search with multi-scale contextual evidences", *in*: *IJCV* 120.1 (2016), pp. 1–13.

[Zho+03a] Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston, and Bernhard Schölkopf, "Learning with Local and Global Consistency", *in*: *NIPS*, 2003.

[Zho+03b] Dengyong Zhou, Jason Weston, Arthur Gretton, Olivier Bousquet, and Bernhard SchÖlkopf, "Ranking on Data Manifolds", *in*: *NIPS*, 2003.

[Zho+15] Bolei Zhou, Aditya Khosla, Àgata Lapedriza, Aude Oliva, and Antonio Torralba, "Object Detectors Emerge in Deep Scene CNNs", *in*: *ICLR* (2015).

[Zho+16] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba, "Learning Deep Features for Discriminative Localization", *in*: *CVPR*, June 2016.

[Zhu+06] Xiaojin Zhu, Jaz Kandola, John Lafferty, and Zoubin Ghahramani, "Graph Kernels By Spectral Transforms", *in*: *Semi-Supervised Learning* (2006), pp. 277–291.

[Zhu+18a] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai, "Deformable ConvNets v2: More Deformable, Better Results", *in*: *CVPR*, 2018.

[Zhu+18b] Yingying Zhu, Jiong Wang, Lingxi Xie, and Liang Zheng, "Attention-based Pyramid Aggregation Network for Visual Place Recognition", *in*: *arXiv preprint arXiv:1808.00288* (2018).

[ZLG03a] Xiaojin Zhu, John Lafferty, and Zoubin Ghahramani, "Combining active learning and semi-supervised learning using gaussian fields and harmonic functions", *in*: *ICML 2003 workshop on the continuum from labeled to unlabeled data in machine learning and data mining*, 2003.

[ZLG03b] Xiaojin Zhu, John Lafferty, and Zoubin Ghahramani, "Combining active learning and semi-supervised learning using gaussian fields and harmonic functions", *in*: *ICML 2003 workshop on the continuum from labeled to unlabeled data in machine learning and data mining*, 2003.

[ZP04] Lihi Zelnik-Manor and Pietro Perona, "Self-Tuning Spectral Clustering", *in*: *NIPS*, 2004.

[ZTF11] Matthew D. Zeiler, Graham W. Taylor, and Rob Fergu, "Adaptive deconvolutional networks for mid and high level feature learning", *in*: *ICCV*, 2011.

UNIVERSITE BRETAGNE LOIRE / MATHSTIC

UNIVERSITÉ DE RENNES 1

Inria

Titre : Représentation robuste d'images pour les tâches de classification d'images, de recherche d'images et de découverte d'objets dans une image

Mots clés : Vision par ordinateur, représentation d'image, localisation d'objets

Résumé : Les réseaux de neurones convolutifs (CNNs) ont été exploités avec succès pour la résolution de tâches dans le domaine de la vision par ordinateur tels que la classification, la segmentation d'image, la détection d'objets dans une image ou la recherche d'images dans une base de données. Typiquement, un réseau est entrainé spécifiquement pour une tâche et l'entrainement nécessite une très grande quantité d'images annotées. Dans cette thèse, nous proposons des solutions pour extraire le maximum d'information avec un minimum de supervision. D'abord, nous nous concentrons sur la tâche de classification en examinant le processus d'apprentissage actif dans le contexte de l'apprentissage profond. Nous montrons qu'en combinant l'apprentissage actif aux techniques d'apprentissage semi-supervisé et non supervisé, il est possible d'améliorer significativement les résultats. Ensuite, nous étudions la tâche de recherche d'images dans une base de données et nous exploitons les informations de localisation spatiale disponible directement dans les cartes d'activation produites par les CNNs. En première approche, nous proposons de représenter une image par une collection de caractéristiques locales, détectées dans les cartes, qui sont peu coûteuses en terme de mémoire et assez robustes pour effectuer une mise en correspondance spatiale. Alternativement, nous découvrons dans les cartes d'activation les objets d'intérêts des images d'une base de données et nous structurons leurs représentations dans un graphe de plus proches voisins. En utilisant la mesure de centralité du graphe, nous sommes capable de construire une carte de saillance, par image, qui met en lumière les objets qui se répètent et nous permet de construire une représentation globale qui exclue les objets non pertinents et d'arrière-plan.

Title: Robust image representation for classification, retrieval and object discovery

Keywords: Computer vision, image representation, object localization

Abstract: Neural network representations proved to be relevant for many computer vision tasks such as image classification, object detection, segmentation or instance-level image retrieval. A network is trained for one particular task and requires a large number of labeled data. We propose in this thesis solutions to extract the most information with the least supervision. First focusing on the classification task, we examine the active learning process in the context of deep learning and show that combining it to semi-supervised and unsupervised techniques boost greatly results. We then investigate the image retrieval task, and in particular we exploit the spatial localization information available "for free" in CNN feature maps. We first propose to represent an image by a collection of affine local features detected within activation maps, which are memory-efficient and robust enough to perform spatial matching. Then again extracting information from feature maps, we discover objects of interest in images of a dataset and gather their representations in a nearest neighbor graph. Using the centrality measure on the graph, we are able to construct a saliency map per image which focuses on the repeating objects and allows us to compute a global representation excluding clutter and background.