# New methods for image classification, image retrieval and semantic correspondence

Rafael Sampaio de Rezende

# THÈSE DE DOCTORAT

de l'Université de recherche Paris Sciences Lettres
PSL Research University

## Préparée à l'École normale supérieure

# New Methods for Image Classification, Image Retrieval and Semantic Correspondence

Nouvelles Méthodes pour Classification d'Image, Recherche d'Image et Correspondance Sémantique.

## École doctorale 386

ÉCOLE DOCTORALE DE SCIENCES MATHÉMATIQUES DE PARIS CENTRE

**Spécialité** INFORMATIQUE

**COMPOSITION DU JURY** :

M Francis Bach
Inria Paris, Directeur de thèse

M Matthieu Cord
Université Pierre et Marie Curie, Président du jury

M Frédéric Jurie
Université de Caen Normandie, Rapporteur

M Patrick Pérez
Technicolor Rennes, Membre du jury

M Florent Perronnin
Naver Labs, Rapporteur

M Jean Ponce
ENS Paris, Directeur de thèse

**Soutenue par Rafael SAMPAIO DE REZENDE**
**le 19 Décembre 2017**

Dirigée par **Jean PONCE et Francis BACH**

ENS ÉCOLE NORMALE SUPÉRIEURE | PSL RESEARCH UNIVERSITY PARIS

To the memory of my grandfathers, Joaquim and Gege.
À memoria dos meus avôs, Joaquim e Gege.

# Abstract

The problem of image representation is at the heart of computer vision. The choice of feature extracted of an image changes according to the task we want to study. Large image retrieval databases demand a compressed global vector representing each image, whereas a semantic segmentation problem requires a clustering map of its pixels. The techniques of machine learning are the main tool used for the construction of these representations. In this manuscript, we address the learning of visual features for three distinct problems: Image retrieval, semantic correspondence and image classification.

First, we study the dependency of a Fisher vector representation on the Gaussian mixture model used as its codewords. We introduce the use of multiple Gaussian mixture models for different backgrounds, *e.g.*, different scene categories, and analyze the performance of these representations for object classification and the impact of scene category as a latent variable.

Our second approach proposes an extension to the exemplar SVM feature encoding pipeline. We first show that, by replacing the hinge loss by the square loss in the ESVM cost function, similar results in image retrieval can be obtained at a fraction of the computational cost. We call this model square-loss exemplar machine, or SLEM. Secondly, we introduce a kernelized SLEM variant which benefits from the same computational advantages but displays improved performance. We present experiments that establish the performance and efficiency of our methods using a large array of base feature representations and standard image retrieval datasets.

Finally, we propose a deep neural network for the problem of establishing semantic correspondence. We employ object proposal boxes as elements for matching and construct an architecture that simultaneously learns the appearance representation and geometric consistency. We propose new geometrical consistency scores tailored to the neural network's architecture. Our model is trained on image pairs obtained from keypoints of a benchmark dataset and evaluated on several standard datasets, outperforming both recent deep learning architectures and previous methods based on hand-crafted features.

We conclude the thesis by highlighting our contributions and suggesting possible future research directions.

**Keywords** : Image representation, image retrieval, kernel methods, neural networks, region matching, semantic matching, Fisher vectors.

# Résumé

Le problème de représentation d'image est au cœur du domaine de vision. Le choix de représentation d'une image change en fonction de la tâche que nous voulons étudier. Un problème de recherche d'image dans des grandes bases de données exige une représentation globale compressée, alors qu'un problème de segmentation séantique nécessite une carte de partitionnement de ses pixels. Les techniques d'apprentissage statisque sont l'outil principal pour la construction de ces représentations. Dans ce manuscrit, nous abordons l'apprentissage des représentations visuels dans trois problèmes différents: la recherche d'image, la correspondance sémantique et classification d'image.

Premièrement, nous étudions la représentation vectorielle de Fisher et sa dépendence sur le modèle de mélange Gaussien employé. Nous introduisons l'utilisation de plusieurs modèles de mélange Gaussien pour différents types d'arrière-plans, *e.g.*, différentes catégories de scène, et analyser la performance de ces représentations pour objet classification et l'impact de la catégorie de scène en tant que variable latente.

Notre seconde approche propose une extension de la représentation l'exemple SVM pipeline. Nous montrons d'abord que, en remplaçant la fonction de perte de la SVM par la perte carrée, on obtient des résultats similaires à une fraction de le coût de calcul. Nous appelons ce modèle la « square-loss exemplar machine », ou SLEM en anglais. Nous introduisons une variante de SLEM à noyaux qui bénéficie des même avantages coputationnelles mais affiche des performances améliorées. Nous présentons des expériences qui établissent la performance et l'efficacité de nos méthodes en utilisant une grande variété de représentations de base et de jeux de données de recherche d'images.

Enfin, nous proposons un réseau neuronal profond pour le problème de l'établissement sémantique correspondance. Nous utilisons des boîtes d'objets en tant qu'éléments de correspondance pour construire une architecture qui apprend simultanément l'apparence et la cohérence géométrique. Nous proposons de nouveaux scores géométriques de cohérence adaptés à l'architecture du réseau de neurones. Notre modèle est entraâiné sur des paires d'images obtenues à partir des points-clés d'un jeu de données de référence et évaluées sur plusieurs ensembles de données, surpassant les architectures d'apprentissage en profondeur récentes et méthodes antérieures basées sur des caractéristiques artisanales. Nous terminons la thèse en soulignant nos contributions et en suggérant d'éventuelles directions de recherche futures.

**Mots-clés** : Représentation d'image, modèle à noyaux, recherche d'image, réseau de neurone, correspondance de région, correspondance sémantique, vecteurs de Fisher.

# Acknowledgements

This work was supported by the ERC grants VideoWorld and the Institut Universitaire de France.

# Contents

# Chapter 1

# Introduction

The growing availability of data is one of the main drivers on innovation in the domain of computer vision. Thanks to social media, the widespread use of surveillance technology and the success of medical imagery, we produce more data than ever and ease the process of annotation.

The increase in volume and complexity of the data we use is shown in the types of datasets we use over time. Caltech 101 [26] was, at time of its publication in 2004, the main benchmark dataset for object detection, object recognition and image classification. It contains around 9000 instances of objects in 101 different classes and each instance is a cropped image centered on the main object and negligible background. Three years later, the PASCAL Visual Objects Classes Challenge [25] introduced a dataset for the same tasks of 12 thousand instances with non-centered objects with rich background information. This category of images makes the PASCAL dataset more complex and tasks of classification and detection more challenging. More recently, the ImageNet database [24] assembles 12 million images. This dataset also allows hierarchical labeling, adding a layer of complexity to the aforementioned classification and detection problems, where we are comparing and contrasting instances of the same category.

The sheer number of images and labels renders the recognition and classification tasks more difficult, but the content of their images has also made these datasets more challenging. Comparing Fig. 1-1, 1-2 and 1-3, we first notice a difference in resolution. Looking more carefully, we see in the more recent datasets more complex poses, noisy background information, blurry boundaries between foreground and background and instances of objects of a category different from the main object. The evolution of these datasets mirrors the boosting diversity of data available today both to research proposes and to commercial ventures.

The increasing complexity of these datasets demands an equally increasing complexity of the feature representations we employ to its images. Indeed, the increase of object poses, viewpoints and positions on a dataset demands a feature more robust to this variability. Similarly, exploring large datasets is time-consuming and demands more compact features that do not loose much information when reduced to a lower dimension.

The goal of this thesis is to explore the image representation problem. This is a

Figure 1-1: Examples of images of Caltech 101.


Figure 1-2: Examples of images of PASCAL VOC.

Figure 1-3: Examples of images of ImageNet.

wide subject that can take many different forms according to the task we want to address. For example, large image retrieval databases demand a compressed global visual feature representation of an image, whereas a semantic segmentation problem requires a clustering map of its pixels. Additionally, tasks that use similar representations may be interested in different types of information. Indeed, object classification and image retrieval use a global visual feature representation of its images. However, an object classification feature is not interested in representing the background information of the scene of the image, whereas the background is important for retrieval since it serves as the context of the image. Furthermore, designing good representations requires some common knowledge or annotated data. Most modern approaches are based on supervised machine learning methods that demand some kind of annotation of the data.

In this thesis, we present three different works in image representation in three different computer vision tasks: Object classification, image retrieval and semantic correspondence. We briefly introduce each work in the following sections.

## 1.1 Image classification with contextual Fisher vectors (CFV)

We consider the problem of image classification, *i.e.*, we wish to associate to a given image one or more semantic categories based on the image's content. We are more particularly interested in the object classification task, where the semantic categories correspond to objects present in the image. This task is one of the most fundamental problems of image understanding and is presented as a clear case of machine learning applied to a computer vision problem: Given an object category and a database of images previously labeled as +1 (contains the object) or 0 (does not contain the object), can we predict the label of a unlabeled image? Since this is a well established task in machine learning, which provides us good candidates to classification algorithm (SVM, kernel SVM, kNN) we concentrate in the construction of a robust image representation pipeline.

We propose in Chapter 2 two methods of improving the existing Fisher vector encoding pipeline:

- a generalization of Fisher vectors where we introduce the concept of multiple background types, each one providing a Fisher encoding.

- a projection of the original Fisher vector that minimizes the background information of the image.

The use of feature vector encoding a full image, such as Bag-of-words [21] or VLAD features [23], was the first very successful method for feature construction for image classification. Fisher vectors often get the spotlight due to high-order statistics and easy generalization of a probability law as a Gaussian mixture model (GMM). However, this generalization of a probability law over many images may fail to represent individual images if they do not share the same type of background information. For example, a probability law calculated over two images, one indoor and one outdoor, will fail to represent the background of both, turning the Fisher vector's strength into a weakness. In order to address this issue, we introduce the Contextual Fisher Vector (CFV) generalization, that considers different GMM for different scene categories, in order to better represent each category and thus eliminate the background information, improving its results in object classification.

We are also interested in the learning of the GMM used as probability law. More specifically, we question which descriptors are used to learn the distribution and also in how much the final feature representation is independent of background information, *i.e.*, descriptors of parts of the image that do not contain relevant information of the object category. Following the theoretical assumption that the Fisher encoding erases background descriptors, we suggest modifications in the standard Fisher vector pipeline in order to obtain a representation dependent only on the object foreground of an image.

Our experiments suggest the following:

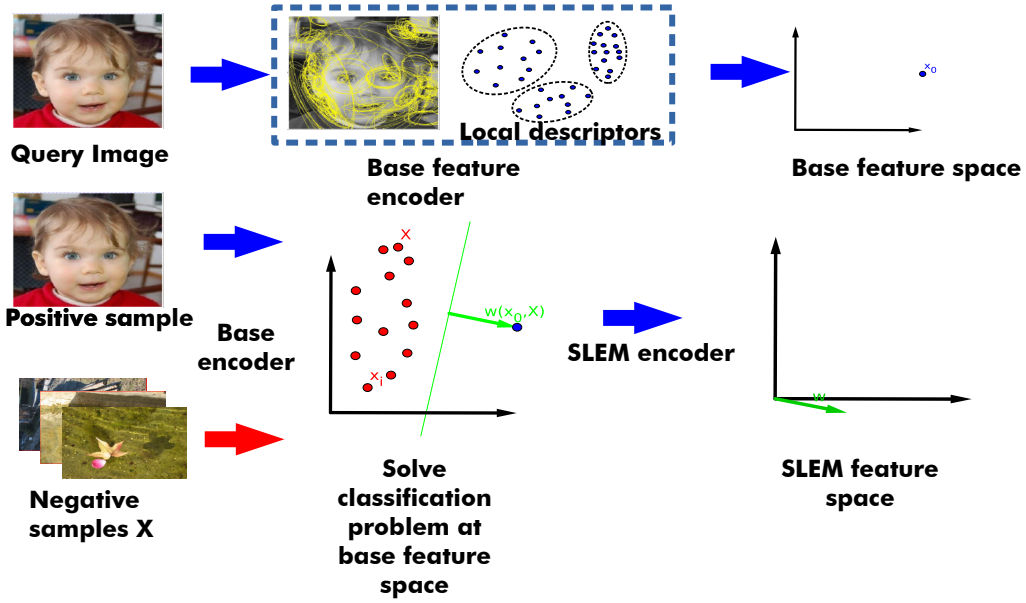


Figure 1-4: Pipeline of SLEM. First row encapsulates the construction of a base feature for a query image, which usually consists of extracting, embedding and aggregating local descriptors into a vector, here written as $x_0$. After repeating the process of base feature calculation a database of sample images and obtaining a matrix $X$ of base features, we solve a exemplar classifier by labeling $x_0$ as the lonely positive example (called *exemplar*) and the columns of $X$ as negatives. The solution $\omega$ to this classification problem, which is a function of $x_0$ and $X$, is our SLEM encoding of the query image.

– Our two central assumptions (Gaussian mixture model calculated separately on different contexts and only in background images) can slightly improve the foreground/background ratio of the Fisher vector representation, but do not improve classification results in fairly simple object classification tasks.

– We obtain significant improvements projecting Fisher vectors in a subspace that eliminates most of the background information, with or without depending on the foreground information. These results however come with no implicit theoretical justification and can overfit the training examples' background.

## 1.2 Image retrieval with square-loss exemplar machine (SLEM)

Chapter 3 addresses the problem of image retrieval by proposing a new feature encoding pipeline that represents an image by the solution of an exemplar classification problem. The exemplar classifier, most specifically the exemplar SVM (ESVM), was

introduced to be used in ensembles of weighted classifiers for object detection [62]. It consists of solving a unbalanced classification problem, of one positive *exemplar* against a large pool of negative samples. The premise is that an ESVM classifier learns discriminative information of the exemplar image instead of the image's object class. Such premise is used here to justify the use of the exemplar classifier as image representation. Indeed, recent work by Zepeda and Pérez has introduced the use of ESVM as a feature encoder for image retrieval [112].

This chapter presented here is an extended version of the paper *"Square-loss exemplar machines for image retrieval"* by Rezende, Zepeda, Ponce, Bach and Pérez published in CVPR 2017 [79]. Our work extends the approach of [112] in several directions: We first show that replacing the hinge loss by the square loss in the ESVM cost function significantly reduces encoding time with negligible effect on accuracy. We call this model square-loss exemplar machine, or SLEM. Then we introduce a kernelized SLEM which can be implemented efficiently through low-rank matrix decomposition, and displays improved performance. An important source of inefficiency of a kernel classifier with respect to a linear classifier is the computation of the kernel matrix. However, both kernelized and non-kernelized SLEM exploit the fact that the negative examples are fixed, so most of the SLEM computational complexity is relegated to an offline process independent of the positive examples. For kernel SLEM, fixing all but one sample across classifiers means simplifying the online calculations from $O(n^2)$ computation of a $(n+1) \times (n+1)$ kernel matrix to a $O(n)$ computation of the column and row corresponding to the positive exemplar.

Our experiments establish the performance and computational advantages of our approach. Our experiments are performed on three standard datasets for image retrieval, namely INRIA Holidays dataset [46], Oxford 5k buildings dataset and Oxford 105k buildings dataset [72]. We demonstrate both the time efficiency and the improvement of retrieval results using a large array of base features, both hand-crafted features (VLAD features [23]) and neural network learned (last convolutional layer outputs of AlexNet architecture [56], VGG+SPoC architecture [6] and NetVLAD features [1]) as well as densely extracted local descriptors. Our approach is tested for a several combinations of kernel functions and ranks of decompositions.

The contributions of this work are threefold:

- We introduce a kernelized variant of SLEM that enjoys similar computational advantages and improves retrieval performances.
- We propose a low-rank factorization of the kernel matrix for computational and storage efficiency.
- Our experiments show improved results on a variety of base features and we obtain state-of-the-art results for Inria Holidays dataset.

## 1.3 Semantic correspondence with convolutional neural networks (SCNet)

Chapter 4 addresses the problem of establishing semantic correspondences between images depicting different instances of the same object or scene category. Include different instances means a much larger change in appearance and spatial layout than the pictures of the same scene used in stereo vision, which we take here to include broadly not only classical (narrow-baseline) stereo fusion (e.g., [67, 80]), but also optical flow computation (e.g., [41, 77, 99]) and wide-baseline matching (e.g., [65, 104]). Due to such a large degree of variations, the problem of semantic correspondence remains very challenging. Most previous approaches to semantic correspondence [14, 43, 52, 59, 90, 104] focus on combining an effective spatial regularizer with hand-crafted features such as SIFT [61], DAISY [92] or HOG [94]. With the remarkable success of deep learning approaches in visual recognition, several learning-based methods have also been proposed for both stereo vision [29, 37, 110, 111] and semantic correspondence [18, 54, 114]. Yet, none of these methods exploits the geometric consistency constraints that have proven to be a key factor to the success of their hand-crafted counterparts. Geometric regularization, if any, occurs during post-processing but not during learning (e.g., [110, 111]).

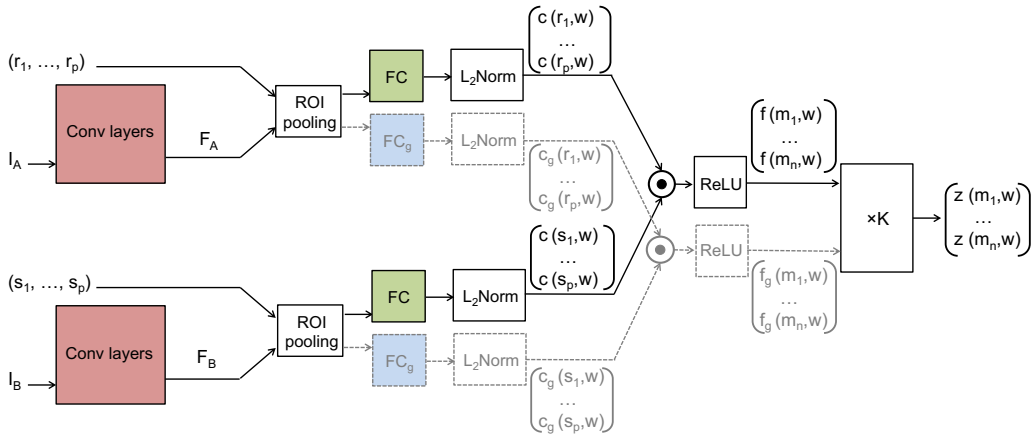


Figure 1-5: The SCNet architectures. Three variants are proposed: SCNet-AG, SCNet-A, and SCNet-AG+. The basic architecture, SCNet-AG, is drawn in solid lines. Colored boxes represent layers with learning parameters and the boxes with the same color share the same parameters. "$\times K$" denotes the voting layer for geometric scoring. A simplified variant, SCNet-A, learns appearance information only by making the voting layer an identity function. An extended variant, SCNet-AG+, contains an additional stream drawn in dashed lines. SCNet-AG learns a single embedding $c$ for both appearance and geometry, whereas SCNet-AG+ learns an additional and separate embedding $c_g$ for geometry.

We propose a convolutional neural network (CNN) architecture, called SCNet, for learning geometrically plausible semantic correspondence based on probabilistic Hough matching (PHM), which allows us to handle a large degree of intra-class and

scene variations. Following the proposal flow approach to semantic correspondence of Ham *et al.*[35], we use object proposals [64, 94, 116] as matching primitives, and explicitly incorporate the geometric consistency of these proposals in our loss function. Unlike [35] with its hand-crafted features, however, we train our system in an end-to-end manner using image pairs extracted from the PASCAL VOC 2007 keypoint dataset [25]. A comparative evaluation on several standard benchmarks demonstrates that the proposed approach substantially outperforms both recent deep architectures and previous methods based on hand-crafted features. This part of the chapter is an extended version of the paper "*SCNet: Learning semantic correspondence*" by Han, Rezende, Ham, Wong, Cho, Schmid and Ponce published in ICCV 2017 [36] and was made in collaboration with another PhD student, Kai Han, from the University of Hong Kong. The extent of the contribution of each of us is further identified in Chapter 4.

Additionally, we present a ongoing work of a new model of geometric consistency. We take inspiration from the local offset matching (LOM) of [35], which is not suited to a differentiable CNN architecture. We propose a new geometric model for local best appearance matching (LBAM) similar to LOM but adapted to the SCNet architecture (as presented in Fig. 1-5). The results presented show improved results with respect to PHM for hand-crafted appearance features, but incomplete results when applied to the SCNet learning pipeline.

Our main contributions can be summarized as follows:

- We introduce a simple and efficient model for learning to match regions using both appearance and geometry of matching object proposals.

- We propose a convolutional neural network, SCNet, to learn semantic correspondence with region proposals.

- We achieve state-of-the-art results on several benchmarks, clearly demonstrating the advantage of learning both appearance and geometric terms.

# Chapter 2

# Contextual Fisher Vectors

The problem of constructing a feature representation for object classification is connected to the elimination of background information. Indeed, the goal of object classification is to ensemble images based on the class of the foreground object, and the different contexts in which this object appears can act as distracting information. This chapter is dedicated to our study of an alternative formulations of Fisher vectors and how they can improve upon the existing pipeline. We first propose to calculate the codewords used as embedding from local descriptors that do not include foreground information. Later, we study how to reduce Fisher to the components that minimize the background information. Finally, we introduce the contextual Fisher vector, which consists of a concatenation of projected Fisher vectors calculated for different contexts of scene categories. Our experiments in a diverse group of dataset images show inconsistent results and we conclude this work is unsuccessful.

## 2.1 Introduction

We consider the problem of image classification, *i.e.*we wish to associate to a given image one or more semantic categories based on the image's content. We are more particularly interested in the object classification task, where the semantic categories correspond to objects present in the image. This task is one of the most fundamental problems of image understanding and is presented as a clear case of machine learning applied to a computer vision problem: Given an object category and a database of images previously labeled as +1 (contains the object) or 0 (does not contain the object), can we guess the label of a unlabeled image? Since this is a well established task in machine learning, which provides us good candidates to classification algorithm (SVM, kernel SVM, kNN) we concentrate in the construction of a robust image representation pipeline.

Fisher vectors follow a similar pipeline of many images features used in computer vision, such as Bag of Visual Words [21], VLAD [23] and Super Vector [115]. These features are built upon a set of local descriptors (SIFT [61] or color features [20]). For a given image, local descriptors are first extracted and then mapped by a function parameterized by the distribution of these descriptors. These mapped descriptors are

then pooled (either max-pooled, average pooled or some mixture, such as generalized max-pooling [66]) and adapted (e.g. whitening, normalizations, linear operations) before inputted to a classification algorithm (linear SVM, Nearest Neighbors, amount others classification algorithms). The resulting feature is a well established hand-crafted representation for image retrieval [71] and image classification [70]. This chapter tries to explore the object classification problem, which consists corresponds to a classify images according to the category of the main object in the image.

We are interested in the distribution used in the mapping of these descriptors and, more specifically, in the descriptors used to learn the distribution and also in how much the final feature representation is independent of background information, *i.e.*, descriptors of parts of the image that do not contain relevant information to the image classification task. Following the theoretical assumption that the Fisher encoding erases background descriptors, we suggest modifications in the standard Fisher vector pipeline in order to obtain a representation dependent only on the object foreground of an image.

Since we start from a theoretical interpretation of how Fisher vectors work, we test our implementation in a variety of datasets, each one representing a different level of complexity:

- A toy dataset where both foreground and background follow their assigned probability distribution and are i.i.d..

- EPFL Car Database [68], where different instances of the same object appear in the same background.

- Scene Understanding Database [102], a large dataset categorized by scene types, which we use as a shared background for the foreground objects normally found in these scenes.

Our experiments suggest that two of our central assumptions (Gaussian mixture model calculated separately on different contexts and only in background images) can slightly improve the ratio foreground/background on FV representation, but do not improve classification results in fairly simple object classification tasks. We obtain significant improvements projecting the FV in a subspace that eliminates most of the background information, with or without depending on the foreground information. These results however come with no implicit theoretical justification and can overfit the training example's background.

The contributions of this work are presented as follows:

- We propose a new framework of application of Fisher vectors to context-independent tasks, by aiming to eliminate the background information.

- We introduce a measurement of the foreground-background ratio of information present in a Fisher vector and propose a projection that maximizes it.

– We present results that contradicts the claims of our method and shed light on the statistical formulation used to justify the normalization of Fisher vectors.

## 2.2 Background

We consider the application of Fisher vector and similar image representations to the image classification task. Many hand-crafted image representations consist of extracting, embedding and aggregating local descriptors. These local descriptors, normally SIFT descriptors [61], RootSIFT [2] or local color statistics [20], can be computed densely or only at interest points. Then, a "visual vocabulary" is learned in an unsupervised way, by applying a clustering algorithm to a set of descriptors and associate each cluster to a "visual word", normally its geometric center. This vocabulary is then used for embedding and aggregation of the local descriptors of an image. The simplest method is the Bag-of-Visual Words (BoW) [21], where the vocabulary is learned by k-means and each local descriptor is hard-assigned to the word of the vocabulary to which it is the closest. The global feature is given by the average occurrence count of the words, leading to a histogram of visual words appearances. The BoW is amount the most popular pipelines for image classification and, due in part to its simplicity, has been multiple times extended, with soft-assignment [69, 95, 100] that allows for a more complex feature, sparse coding for the visual vocabulary representation of images [13, 97, 105] and implementation of some spatial information of the codewords occurrence with spatial pyramids [57].

Fisher vectors (FV) are, along side with the vector of locally aggregated descriptors (VLAD) [23], the handcrafted aggregation of local descriptors that succeed all the variants of BoW. Similarly to BoW, both are aggregation of hard-assigned embedded local descriptors. They differ in their methods for clustering descriptors (VLAD uses K-means, FV uses Gaussian mixtures model) and the residual vector they accumulate (VLAD accumulate derivatives w.r.t. the mean of the clusters, FV accumulate derivatives w.r.t. the mean and standard deviation of the clusters).

Neural network have consistently outperformed handcrafted features in classifications ever since Krizhevsky *et al.* [56] obtained state-of-the-art of the most challenging classification dataset. Most works that achieve comparable results to [56] nowadays extract low and mid-level descriptors as activations of a similar convolutional architecture [87, 89]. However, many handcrafted features are still used as part of a neural network architecture. Arandjelović *et al.* [1] constructed a network with a differentiable VLAD layer aggregate activations of mid-level (conv5) convolutional layers of an existing architecture, and He *et al.* [40] used spatial pyramids kernel aggregate activations of convolutional layers of images of different sizes without resizing them.

## 2.3 Fisher vectors

The Fisher vector encoding of an image is based on fitting a Gaussian mixture model (or any other parametric generative model) to a densely computed set of local

descriptors, and then encoding first and second-order information as the derivatives of the log-likelihood of the model with respect with to its parameters.

Given an image, we wish to encode it as a Fisher vector (FV). We extract a set $X = \{x_n\}_{1 \leq n \leq N}$ of local descriptors of dimension $D$. Let $u_\lambda$ denote a probabilistic density function of a Gaussian mixture parametrized by $\lambda = [\lambda_1, \lambda_2, ..., \lambda_K]^T$, where every $\lambda_i = \{\mu_i, \sigma_i, \gamma_i\}$, $1 \leq i \leq K$ is the parameter of one of the Gaussian distributions in the mixture distribution:

$$u_\lambda(x) = \sum_{i=1}^{K} \gamma_i \mathcal{N}(x|\mu_i, \Sigma_i), \tag{2.1}$$

$$\mathcal{N}(x|\mu_i, \Sigma_i) = (2\pi)^{-\frac{D}{2}} |\Sigma_i|^{-\frac{1}{2}} e^{-\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1}(x-\mu_i)}, \tag{2.2}$$

and $\Sigma_i$ is the square diagonal matrix with main diagonal equals to $\sigma_i^2$. For each local descriptor $x_n$ in $X$, we can calculate its Fisher score $z_n^\lambda = \nabla_\lambda \log u_\lambda(x_n)$, where [1]

$$\nabla_\lambda f(\lambda) = [\nabla_{\mu_1} f(\lambda); \nabla_{\sigma_1} f(\lambda); \ldots; \nabla_{\mu_K} f(\lambda); \nabla_{\sigma_K} f(\lambda)]\,. \tag{2.3}$$

The Fisher score $Z_\lambda$ of the sample $X$ is the mean of the score function of all the local descriptors:

$$Z_\lambda(X) = \frac{1}{N} \sum_{n=1}^{N} z_n^\lambda. \tag{2.4}$$

Since $f(\lambda) = \log u_\lambda(x_n)$ is a scalar function and $\mu_i$, $\sigma_i$ are $\mathbb{R}^D$ vectors, $\nabla_{\mu_i} f(\lambda)$ and $\nabla_{\sigma_i} f(\lambda)$ are $\mathbb{R}^D$ vectors. Therefore, $Z_\lambda$ is a $\mathbb{R}^{2DK}$ vector. We are interested in relative distances between Fisher scores, so we must also find a metric for $\mathbb{R}^{2DK}$. Jaakkola and Haussler [44] proposed the metric given by the inner product $K_{FK}(Z, Z') = ZF_\lambda^{-1}Z'$, where $F_\lambda$ is the Fisher information matrix

$$F_\lambda = \mathbb{E}_{x \sim u_\lambda} \left[ \nabla_\lambda \log u_\lambda(x) (\nabla_\lambda \log u_\lambda(x))^T \right]. \tag{2.5}$$

Since, by its definition, $F_\lambda$ is a symmetric positive-definite matrix, we can obtain a lower-triangular matrix $L_\lambda$ by the Cholesky decomposition of the Fisher Information Matrix: $F_\lambda^{-1} = L_\lambda L_\lambda^T$. Finally, we can obtain the Fisher vector of the samples $X$ as

$$\Psi_\lambda(X) = L_\lambda Z_\lambda(X). \tag{2.6}$$

$\Psi_\lambda(X)$ is a $2DK$ vector whose relative distance to another Fisher vector can be measured using the $l_2$ norm in $\mathbb{R}^{2DK}$.

1. In Eq. (2.3), the gradient w.r.t. $\gamma$ was omitted on propose. Most applications of Fisher vectors exclude this gradient because it does not add useful information to the derivative w.r.t. the means and variances, as seen on Figure 4 of [81]. Also, for an efficient calculation of the encoding of a couple of samples, it is useful to compare their gradients w.r.t. the same Gaussian distribution. Indeed, it can be done, following SSE2 CPU instructions, only if the encoding dimensions is divisible by 4 [51]. By excluding the gradient w.r.t. $\gamma$ and choosing $K$ even, we speed up our code considerably.

### 2.3.1 Normalization

Perronnin *et al.* [70] introduced several practices to obtain better results from FV. Here we highlight the practice of normalization, which improves results in classification as seen in [81]. These improvements are obtained by applying the following functions to the feature calculated in Eq. (2.6):

$$\varphi_1(x) = sign(x)|x|^{\frac{1}{2}}, \tag{2.7}$$

$$\varphi_2(x) = \frac{1}{||x||_2}x. \tag{2.8}$$

Equation (2.7) is a signed square-rooting, similar to the Hellinger kernel which has successfully improved results for Bag of Visual Words features [96]. Equation (2.8) is a $l_2$ normalization. Its justification is discussed in the subsection 2.3.2 and it is also one of the inspirations of this work. The post normalization FV is given by the equation

$$\Psi_\lambda(X) = \varphi_2\left(\varphi_1\left(L_\lambda Z_\lambda(X)\right)\right). \tag{2.9}$$

### 2.3.2 Application of Fisher vectors to classification

Fisher vectors are normally applied on classification using the following interpretation: Let $X = \{x_1, x_2, ..., x_N\}$ be the sample of local descriptors of in $\mathbb{R}^D$ an image and $u_\lambda$ a function from $\mathbb{R}^D$ to $\mathbb{R}$ that represents a Gaussian Mixture Model distribution (GMM) parametrized by $\lambda = \{\mu_i, \sigma_i, \gamma_i\}_{1\leq i\leq K}$. The Fisher vector $\Psi_\lambda(X)$ of $X$ with respect to $\lambda$ is given by the product of the Fisher information matrix $L_\lambda$ and global score $Z_\lambda(X)$:

$$z_n^\lambda = \nabla_\lambda \log u_\lambda(x_n),\ Z_\lambda(X) = \frac{1}{N}\sum_{n=1}^{N} z_n^\lambda. \tag{2.10}$$

Suppose that the samples in $X$ are i.i.d. and follow a distribution $p$ in $L^2(\mathbb{R}^D)$, then we have that almost surely, for $N \to \infty$,

$$Z_\lambda(X) \xrightarrow{a.s.} \mathbb{E}_{x\sim p}[\nabla_\lambda \log u_\lambda(x)] = \nabla_\lambda \int_{x\in\mathbb{R}^D} p(x)\log u_\lambda(x)dx. \tag{2.11}$$

We now assume that we can decompose $p$ into two parts, depending on the image: a proportion $\omega$ of the image contains class-specific information, and its descriptors follow a distribution $p_{fg}$, whereas the remaining $1-\omega$ of the image is filled with class-independent background, which follows the distribution $p_{bg}$. Hence, we write $p(x) = \omega p_{fg}(x) + (1-\omega)p_{bg}(x)$ and rewrite (2.11) as

$$Z_\lambda(X) \xrightarrow{a.s.} \omega\nabla_\lambda \int_{x\in\mathbb{R}^D} p_{fg}(x)\log u_\lambda(x)dx + (1-\omega)\nabla_\lambda \int_{x\in\mathbb{R}^D} p_{bg}(x)\log u_\lambda(x)dx. \tag{2.12}$$

Sánchez *et al.* [81] and many other works suppose that the background descriptors of any image follow the the same probabilistic distribution as the GMM $u_\lambda$ used to compute its Fisher vector. Hence, $\lambda$ is estimated in order to eliminate the dependency of the global score from the background descriptors, which, from Eq. (2.12),

$$\lambda = \underset{\substack{\mu_i,\sigma_i\in\mathbb{R}^D,\\ \gamma_i\geq 0,\ \sum\gamma_i=1}}{argmax} \int_{x\in\mathbb{R}^D} p_{bg}(x)\log u_\lambda(x) \approx \underset{\substack{\mu_i,\sigma_i\in\mathbb{R}^D,\\ \gamma_i\geq 0,\ \sum\gamma_i=1}}{argmax} \sum_{x\in X,\ x\sim p_{bg}} \log u_\lambda(x), \tag{2.13}$$

and under this assumption, we find

$$Z_\lambda(X) \xrightarrow{a.s.} \omega\nabla_\lambda \mathbb{E}_{x\sim p_{fg}}[\log u_\lambda(x)]. \tag{2.14}$$

The last equality follows from the fact that $\lambda$ maximizes the log-likelihood of the data generated by $u_\lambda$. This allows us to erase the background information of the image. A $l_2$-normalization is proposed by [81] as a way to eliminate the proportion $\omega$, thus making the FV encoding robust to changes in scale of the object.

## 2.4 Our model for Fisher vectors

The first problem of Eq. (2.12) is the assumption that the background follows the distribution $u_\lambda$. The distribution $u_\lambda$ is learned so that the GMM represent a codebook of the database and must represent discriminative keywords of the classes in the database. Indeed, [44, 81] learn $\lambda$ by optimizing a maximum likelihood with an EM algorithm using local descriptors collected over the whole images. Therefore, the foreground descriptors do not follow a distribution $p_{fg}$ different from $u_\lambda$; both foreground and background descriptors follow $u_\lambda$.

We propose to calculate the distribution $u_\lambda$ using only the background samples, using object bounding boxes as the boundary between foreground and background.

An alternative modification to the encoding pipeline is to project the non-normalized Fisher vectors along the linear space that maximizes the ratio foreground over background information. This projection is obtained by solving a generalized eigenvalue problem.

Another important problem of the interpretation of Section 2.3 is the variability of the backgrounds. The strength of model is that we can erase the background information, but when a database contains different types of scenes (indoor and outdoor, city and countryside, Brazil and China, etc.), the background information of an image will not be the same as the average information of all the images and the background term in (2.12) will not cancel out. We suggest to solve this by separating images according to the type of background information.

In the following subsections, we are going to detail these modifications to the Fisher encoding.

### 2.4.1 Separation of foreground and background local features

For a sample of local descriptors $X$, we write $X = X_{fg} \cup X_{bg}$, where $X_{fg} \cap X_{bg} = \emptyset$, where $X_{fg}$ and $X_{bg}$ are respectively the set of foreground and background descriptors in $X$. In practice, this separation is done using the ground-truth object proposal: $X_{fg}$ is the set of local descriptors extracted inside the object proposal and $X_{bg} = X \backslash X_{fg}$ is its complementary subset of $X$. Hence we assume for a dataset of $N$ images we use for computing $u_\lambda$, we have the sets of local descriptors $\{X^{(1)}, X^{(2)}, ..., X^{(N)}\}$ of images $\{1, 2, ..., N\}$, and for each sample, the foreground groundtruth is known, i.e., for each image $n$ , $X_{fg}^{(n)}$ and $X_{bg}^{(n)}$ are known. Since $u_\lambda$ is a GMM, the parameter $\lambda$ is obtained by maximizing a log-likelihood problem

We wish to compare two methods of calculation of the distribution $u_\lambda$;

– Full-image (fg+bg) GMM, where all local descriptors are used and proposal ground-truth is unnecessary:

$$\lambda_{fg+bg} = \arg\max_{\lambda} \left( \sum_{n=1}^{N} \sum_{x \in X^{(n)}} \log u_\lambda(x) \right). \tag{2.15}$$

– Background (bg) GMM, where we restrict the learning of the distribution to background descriptors:

$$\lambda_{bg} = \arg\max_{\lambda} \left( \sum_{n=1}^{N} \sum_{x \in X_{bg}^{(n)}} \log u_\lambda(x) \right). \tag{2.16}$$

(Eq. 2.13) is true only if $u_\lambda = p_{bg}$, which implies $u_\lambda$ must be parametrized by $\lambda_{bg}$. However, previous Fisher vectors applications [19, 71, 70, 81] did not make distinction between foreground and background descriptors, implying their GMM is parametrized by $\lambda_{fg+bg}$. We wish to compare both methods of GMM calculation in order to understand if the choice of probability distribution $u_\lambda$ influences the result on background information deletion and object classification results. For simplicity of notation, we denote by $u_{fg+bg}$ and $u_{bg}$ the GMM parametrized by $\lambda_{fg+bg}$ and $\lambda_{bg}$, respectively.

### 2.4.2 Foreground projection

In this section, we propose a linear projection designed to eliminate background information as much as possible preserving foreground information. This projection is inspired by the Linear discriminant analysis algorithm. We expect to see an improvement in object classification as a result of using the projected feature.

For any set $X$ of local descriptors, let $\Psi(X)$ be the $2DK$-dimension image feature of the sample $X$. In training time, we have the sets of local descriptors $\{X^{(1)}, X^{(2)}, ..., X^{(N)}\}$ of image samples $\{1, 2, ..., N\}$, and for each sample, separable in foreground and background descriptors. We define the matrices $M_{fg}$ and $B$ such that

$$M_{fg} = \left[\Psi(X_{fg}^{(1)}), \Psi(X_{fg}^{(2)}), ..., \Psi(X_{fg}^{(N)})\right], \tag{2.17}$$

$$B = \left[\Psi(X_{bg}^{(1)}), \Psi(X_{bg}^{(2)}), ..., \Psi(X_{bg}^{(N)})\right], \tag{2.18}$$

i.e., the $i$-th column of $M_{fg}$ and $B$ are the image feature of, respectively, the foreground and the background of image $i$, $i$ in $\{1, 2, ..., N\}$.
We want to find a $d$ dimensional projection that maximizes the ratio between the average squared norm of foreground encoding over the average squared norm of background encoding, for some $d < 2DK$. The value of $d$ is chosen so that we maximize classification precision. We can write this as finding a $\mathbb{R}^{2DK \times d}$ matrix $A^*$ argument of the problem

$$A^* = \underset{A \in \mathbb{R}^{2DK \times d}}{argmax} \frac{\sum_{n=1}^{N} ||A^T \Psi(X_{fg}^{(n)})||^2}{\sum_{n=1}^{N} ||A^T \Psi(X_{bg}^{(n)})||^2} = \underset{A \in \mathbb{R}^{2DK \times d}}{argmax} \frac{tr\left(A^T M_{fg} M_{fg}^T A\right)}{tr\left(A^T M_{bg} M_{bg}^T A\right)}. \tag{2.19}$$

Equation (2.19) stats a Generalized Eigenvalue problem: if $BB^T$ is a invertible matrix, there are $\alpha_1 > \alpha_2 > ... > \alpha_d > 0$ such that, for $1 \leq j \leq d$, $M_{fg} M_{fg}^T a_j = \alpha_j M_{bg} M_{bg}^T a_j$ where $a_j$ is the $j$-th column vector of $A^*$. However, the Generalized Eigenvalue problem does not insure the orthogonality of the column vectors of the matrix $A^*$. The linear projection $P$ over $d$ dimensions corresponding to $A^*$ is given by

$$P = A^*(A^{*T} A^*)^{-1} A^{*T}. \tag{2.20}$$

Our new image representation is the projection $P$ of its Fisher vector:

$$\Psi_p(X) = P\Psi(X). \tag{2.21}$$

**Solving the generalized eigenvalue problem** In order to simplify the notation, we call $F = M_{fg} M_{fg}^T$ and $B = M_{bg} M_{bg}^T + \epsilon Id_N$, where $Id_N$ is the $N \times N$ identity matrix and $\epsilon$ is a small positive real number. [2]

$$A^* = \underset{A \in \mathbb{R}^{2DK \times d}}{argmax} \frac{tr\left(A^T F A\right)}{tr\left(A^T B A\right)}. \tag{2.22}$$

Let us assume $A$ is a matrix of rank $d$ and can be written as the orthonormal

2. The following solution requires $B$ to be a invertible matrix.

basis of its image space:

$$A^* = \underset{\substack{A\in\mathbb{R}^{2DK\times d},\\ a_j^T BT a_j=1,\ 1\le j\le d}}{argmax} \quad tr\left(A^T F A\right) \tag{2.23}$$

$$\mathcal{L}(A,\nu) = tr\left(A^T F A\right) + \nu\left(1 - tr\left(A^T B A\right)\right) \tag{2.24}$$

$$= \sum_{j=1}^{d} a_j^T F a_j + \sum_{j=1}^{d} \nu_j \left(1 - a_j^T B a_j\right) \tag{2.25}$$

$$\frac{\partial}{\partial a_j}\mathcal{L}(A,\nu) = 2Fa_j - 2\nu_j B a_j, \tag{2.26}$$

$$\frac{\partial}{\partial a_j}\mathcal{L}(A,\nu) = 0 \Rightarrow (F - \nu_j B)a_j = 0, \tag{2.27}$$

*i.e.*, $a_j$ is an eigenvector of the matrix $B^{-1}F$

### 2.4.3 Contextual Fisher vectors

**Groups of images and context** A solution to the problem of variability of the backgrounds is to separate the training images accordingly with different types of image depending on their background type (*i.e.*, contextual information about where the photo was taken), learn a mixtures of Gaussians for each of these groups and calculate the Fisher vector of an image w.r.t. each of these GMM. We assume we can divide any set of images in different contexts, where a context is defined as a subset of images that share the same type of image, in such a way that an image in this set is contained in exactly one context. Under this assumption, having one GMM for every context and calculating, for every image, the FV with respect to each GMM, every image we will have only one background-free Fisher vector, i.e. one Fisher vector for which (2.12) and (2.14) are true. This is the FV corresponding to the context in which the image is contained. Also, we can not concatenate these different FV into one vector, or else the size can become a prohibitive factor.

**Separation by context** We will assume that each one of the training image is associated to one context $c \in \{1, 2, ..., C\}$ out of $C$ possible ones in the training images set. We fix the number $K$ of Gaussians we want every mixture to have. Each context $c$ is then associated to a probabilistic density $u_{\lambda_c}$, with parameters $\lambda_c = [\lambda_{c1}, \lambda_{c2}, \ldots, \lambda_{cK}]$ is calculated as the GMM parameter learned from the images of this context. Here, $\lambda_{ck} = (\mu_{ck}, \sigma_{ck}, \gamma_{ck})$ is the parameters of the $k$-th gaussian of the c-th context.

**Moment matrices** In (2.12), the background information was erased because $\lambda$ was optimized such that

$$\nabla_\lambda \mathbb{E}_{x\sim u_\lambda}[\log u_\lambda(x)] = \nabla_\lambda \int_{x\in\mathbb{R}^D} u_\lambda(x)\log u_\lambda(x)dx = 0. \tag{2.28}$$

However, in the presence of many contexts and since the context of an image is hidden information, Eq. (2.28) does not hold for every couple samples-context. We obtain the elimination of the background term in (2.12) if the samples $X$ are in the same context with respect which we calculate the Fisher vector. We therefore denote the moment matrix between the contexts $i, j \in \{1, 2, ..., C\}$, by

$$A_{ij} = D_{\lambda_i}\left(\mathbb{E}_{x\sim u_{\lambda_j}}[\log u_{\lambda_i}(x)]\right) = D_{\lambda_i}\left(\int_{x\in\mathbb{R}^D} u_{\lambda_j}(x)\log u_{\lambda_i}(x)dx\right), \tag{2.29}$$

where

$$D_\lambda f(\lambda) = \begin{pmatrix} \nabla_{\mu_1} f(\lambda) & \nabla_{\mu_2} f(\lambda) & \dots \nabla_{\mu_K} f(\lambda) \\ \nabla_{\sigma_1} f(\lambda) & \nabla_{\sigma_2} f(\lambda) & \dots \nabla_{\sigma_K} f(\lambda) \end{pmatrix}. \tag{2.30}$$

Note that $A_{ii} = 0$ and $A_{ij}$ is a $2D \times K$ matrix, where each one of the $K$ columns is a vector in $\mathbb{R}^{2D}$ and the $k$-th column corresponds to the derivative with respect to $\lambda_{ik}$. Consider the following homogeneous system under constraints on $v$ in $\mathbb{R}^K$:

$$\begin{cases} A_{ij}v &= \mathbf{0},\ \forall 1 \le j \le C,\ j \ne i, \\ \|v\|_2 &= 1. \end{cases} \tag{2.31}$$

We call $N_i$, one of the solutions of Eq. (2.31), the *normal to the residual moment* of the context $i$. We can only guarantee the existence of a solution to Eq. (2.31) if $2D(C-1) < K$. Otherwise, we can not guarantee the existence of a non-trivial solution of a homogeneous system of $K$ variables and $2D(C-1)$ equations. The values of $D$, $K$ and $C$ must be chosen such that this inequality is true.

**Construction of CFV** We denote by $\psi_i(X)$, $i \in \{1, 2, ..., C\}$ a $2D \times K$ matrix constructed by rearranging in order the entries of the Fisher vector $\Psi_{\lambda_i}(X)$ along the lines of the matrix, i.e.,

$$\mathcal{Z}_i(X) = \begin{pmatrix} Z_{\lambda_i}(X)_1 & Z_{\lambda_i}(X)_{2D+1} & \dots & Z_{\lambda_i}(X)_{2D(K-1)+1} \\ Z_{\lambda_i}(X)_2 & Z_{\lambda_i}(X)_{2D+2} & \dots & Z_{\lambda_i}(X)_{2D(K-1)+2} \\ \vdots & \vdots & \ddots & \vdots \\ Z_{\lambda_i}(X)_{2D} & Z_{\lambda_i}(X)_{4D} & \dots & Z_{\lambda_i}(X)_{2DK} \end{pmatrix}. \tag{2.32}$$

We can rewrite Eq. (2.11) for $\mathcal{Z}_i(X)$ by equally rewriting Eq. (2.3) with a different arrangement. We have

$$\mathcal{Z}_i(X) \to^{a.s.} D_{\lambda_i}\left(\int_{x\in\mathbb{R}^D} p(x)\log u_{\lambda_i}(x)dx\right), \tag{2.33}$$

for any $i$ in $\{1, 2, ..., C\}$. Let $q$ denote once again the distribution followed by the local descriptors of an object class. Now, from our hypothesis that every image is contained in exactly one context, there is an $\omega$ in $[0, 1]$ and $j$ in $\{1, 2, ..., C\}$ such that $p(x) = \omega q(x) + (1 - \omega)u_{\lambda_j}(x)$. Then, we can introduce this equality in equation above, obtaining an equivalent to (2.12)

$$\mathcal{Z}_i(X) \to^{a.s.} \omega \underbrace{\nabla_{\lambda_i} \int_{x \in \mathbb{R}^D} q(x) \log u_{\lambda_i}(x) dx}_{function\ of\ q\ and\ i} + (1 - \omega) A_{ij}. \tag{2.34}$$

$$\mathcal{Z}_i(X) N_i \to^{a.s.} \omega \left( \nabla_{\lambda_i} \int_{x \in \mathbb{R}^D} q(x) \log u_{\lambda_i}(x) dx \right) N_i$$

The first thing to note in (2.34) is that it is valid for context $i$, independent if it is the background type of the samples $X$. Secondly, the first term depends only on the class-specific function $q$ and context $i$ with respect which we calculated the Fisher encoding. Finally, the last term is proportional to the moment matrix corresponding to $i$ and $j$, and all of its lines are orthogonal to $N_i$, independent of $j$.

$$\psi_i(X) = L_\lambda \mathcal{Z}_i(X) \tag{2.35}$$

Thus, $\psi_i(X)N_i$ is a $2D + 1$ vector and converges almost surely to a function that does not depend on its background type, which means we are able to erase its background information.

We can finally define the CFV of the sample $X$ as given by

$$\Phi(X) = [\psi_1(X)N_1;\ \psi_2(X)N_2;\ ...,\ \psi_C(X)N_C]. \tag{2.36}$$

$\Phi(X)$ is a vector of dimension $2DC$.

## 2.5 Datasets

### 2.5.1 Toy database

Firstly, we test our hypothesis in artificially generated data that simulates perfectly the assumptions of Subsection 2.3.2:

- Local descriptors of an image are i.i.d..
- Background descriptors follow a mixture of 128 Gaussian distributions;
- We suppose foreground descriptors follow a GMM of $q$ Gaussian distributions.
- The parameters of both $p_{fg}$ and $p_{bg}$ are randomly generated.
- For a sample of size $N$ of an image, $\lfloor \omega N \rfloor$ of these descriptors are foreground descriptors and the remaining $\lceil (1 - \omega)N \rceil$ are background descriptors.

We fix the descriptors' dimensions $D = 64$ to imitate the PCA-SIFT dimension.

### 2.5.2 EPFL car database

For the preliminary experiences, in used the EPFL multi-views Car Database [68]. This dataset contains 20 sequences of cars as they rotate 360 degrees. There is one image approximately every 3-4 degrees. For our experiment, we grouped some cars that share the same background and make it a context, as seen in Fig. 2-1. We created 2 contexts from 6 cars; **context 1** contains 436 images of 4 cars and **context 2** contains 234 images of 2 cars. For each context, we calculated a context specific GMM. The general context GMM is learned from samples from both contexts and samples from the remaining images not included in any context. Using the fact that this database has ground-truth bounding boxes for the objects, we can calculate the GMM of each context using only background local descriptors, as described in Eq. (2.16), and compare the Fisher encoding for $u_{fg+bg}$ and $u_{bg}$.

Figure 2-1: Examples of images in context 1 of EPFL Car database.

Figure 2-2: Examples of images in context 2 of EPFL Car database.

Figure 2-3: Examples of images classified in no context of EPFL Car database.

### 2.5.3 SUn database

Scene Understanding (or SUn) [102] database consists of 130k images labeled in about 900 scenes categories and 4000 object categories. The object boundaries are handily labeled by the online annotation tool LabelMe. We use images of the same scene category as images of the same context and calculate one GMM for each context. Differently from the previous dataset, SUn database's images present more variability in scenes and objects, and hence is more suited to report classification results.

Our experiments are conducted in images of four scene categories (highway, street, bathroom and building facade) and three object classes (person, car and toilets).

These scenes and object categories are chosen such that each object can be associated to at least one context (toilet to bathroom, person to street, car to street and highway) and pairs of contexts can be identified as similar and therefore more ambiguous (such as street and highway) or distinct and easier to differentiate (such as highway and bathroom).

Figure 2-4: Examples of images of highway in SUn database.

Figure 2-5: Examples of images of street in SUn database.

Figure 2-6: Examples of images of bathroom in SUn database.

## 2.6 Experiments

### 2.6.1 Experimental setup

All images in each dataset are resized at $1000 \times 1000$. The sample of local descriptors we used in this work is dense SIFT. These descriptors are extracted from $6 \times 6$, $12 \times 12$ and $24 \times 24$ patches on a regular grid every 4 pixels. The 128 dimensional SIFT descriptor is reduced to 64 dimensions using PCA.

One inconvenient of using SIFT of such small patches is the proportion of null gradient patches, giving a null local descriptor. These null descriptors represent an important proportion of the samples and do not cluster with any other local

descriptor. By calculating the a GMM using these samples, the biggest Gaussian (by number of elements generated) have zero mean and a variance much smaller than other clusters. Sánchez *et al.* [81] deal with null descriptors with what they call 'variance flooring': impose a minimum value to the variance of the clusters. However, the gradient of the log-likelihood w.r.t. the variance where the mean is zero does not depend on its variance nor on the samples size, meaning it does not converge as expected in Eq. (2.12). We decided then to dropout the null local descriptors.

On training time, we first separate images according to context. This is usually done using labels of scenes categories as label of context. Using ground-truth bounding boxes, we extract dense SIFT over the background of images only to compute background GMM. We gather around $10^6$ descriptors of each context, reduce dimensionality with PCA and learn $\lambda$ by solving Eq. 2.16. We also calculate a general context GMM, using samples of all contexts and solving Eq. 2.15, to compare its performance to the context specific GMM.

Following the observations of Sánchez *et al.* [81] that the gradient w.r.t. the prior probabilities does not impact the average precision of Fisher vectors in classification takes and slows down Fisher vector calculation, we consider only the gradients w.r.t. the mean and the variance of each Gaussian distribution. The final encoding of a samples has thus size $2DK$, where $D = 64$ is the dimension of the resized PCA-SIFT descriptor and $K$ is the number of Gaussian distribution.

### 2.6.2 Foreground-background ratio

In order to measure the elimination of background information of a samples $X$, we define the ratio $r$ of the FV calculated over the foreground descriptors and the FV calculated over the background descriptors. We define $r$ as a function of the separable sample $X = X_{fg} \cup X_{bg}$ as being

$$r(X, \lambda) = \frac{\|\Psi_\lambda(X_{fg})\|}{\|\Psi_\lambda(X_{bg})\|}. \tag{2.37}$$

Given our theoretical assumptions stated in Section 2.4, we expect to obtain higher values of foreground-background ration:

- For background GMM-calculated FV w.r.t. full-image GMM-calculated FV.
- For context-specific GMM-calculated FV w.r.t general-context GMM-calculated FV.

### 2.6.3 Toy experiments.

In each of the following experiments, after fixing $q$ and randomly assign the parameters of a GMM of $q$ Gaussian distributions as the probability distribution of the foreground descriptors, we generate foreground and background descriptors following its respective distributions. We learn a GMM of $K$ Gaussian distributions from the background descriptors, for $K$ in $\{2, 4, 8, 16, 32, 64, 128\}$. Figure 2-7 plots $ln(r(X, \lambda_{bg}))$

for several foreground classes, each following a randomly generated functions $q$ where we increase the number of Gaussian distributions. We observe that the complexity of the foreground distribution $q$ does not impact the ratio as much as the complexity of the background distribution. Indeed, we observe a linear drop in $ln(r)$ – thus an exponential drop in $r$ – for exponential increases of $K$. These results indicate that the hypothesis of a Fisher vector mostly independent of background information applies better to simple background information, and this can be noted in our simplified data.

Figure 2-8 presents changes in other simulated variable, namely the size $N$ of the sample $X$ and the fraction $\omega$ of foreground descriptors in $X$. The ration $r$ increases as the size $N$ increases and as the proportion $1 - \omega$ of the background descriptors increases. These indicate that as the sheer number of background descriptors increases we are closer to the convergence hypothesized in Eq. (2.11). For all cases, we observe decreasing average value of $r$ as $K$ increases. Despite measuring the dependency of a representation to the foreground information, $r$ does not measure if this representation of the foreground is discriminative. Indeed, higher values of $K$ better approximate the background distribution and improve classification results [81].

### 2.6.4 Background GMM vs full image GMM

Calculating the GMM parameter $\lambda$ only on background descriptors is our first assumption. For each of the following experiments, we compare the average and standard deviation of $r(X, \lambda_{bg})$ and $r(X, \lambda_{fg+bg})$ over a set of images. We calculate these GMM for different numbers of Gaussian distributions $K$ and plot the $ln(r)$ over $K$.

**Toy database.** Figure 2-9 compares the ratios for a set of $N = 10000$ descriptors, generated from two mixtures of 128 Gaussian models each; 1000 from the foreground GMM, 9000 from the background GMM (*i.e.*, $\omega = 0.1$). We learn $\lambda_{bg}$ and $\lambda_{fg+bg}$ from these descriptors for different values of $K$ and plot $ln(r)$ for each $K$. We observe a higher ratios $r$ on background GMM (in blue) for all values of $K$ w.r.t. full-image GMM (in red).

**EPFL cars.** We repeat the experiment for EPFL cars. We solve Eq.(2.15) and Eq. (2.16) on general context, *i.e.*, we do not separate descriptors in their contexts. Results are shown in Fig. 2-9. On average, ratio $r$ is higher for background GMM, but contrary to the results for toy database, we observe a higher standard deviation and a large overlap of ratios. The results also show smaller values of $r$ compared to toy database, and that both $r(\lambda_{bg})$ and $r(\lambda_{fg+bg})$ increase for larger number of Gaussian distributions $K$, contrary to toy database. Interestingly, the average ratio $r(\lambda_{bg})$ is bigger than 1, (*i.e.*, the norm of the foreground feature is bigger than the background feature), whereas the average $r(\lambda_{fg+bg})$ is smaller than one (the norm of the background feature is bigger than the foreground feature).

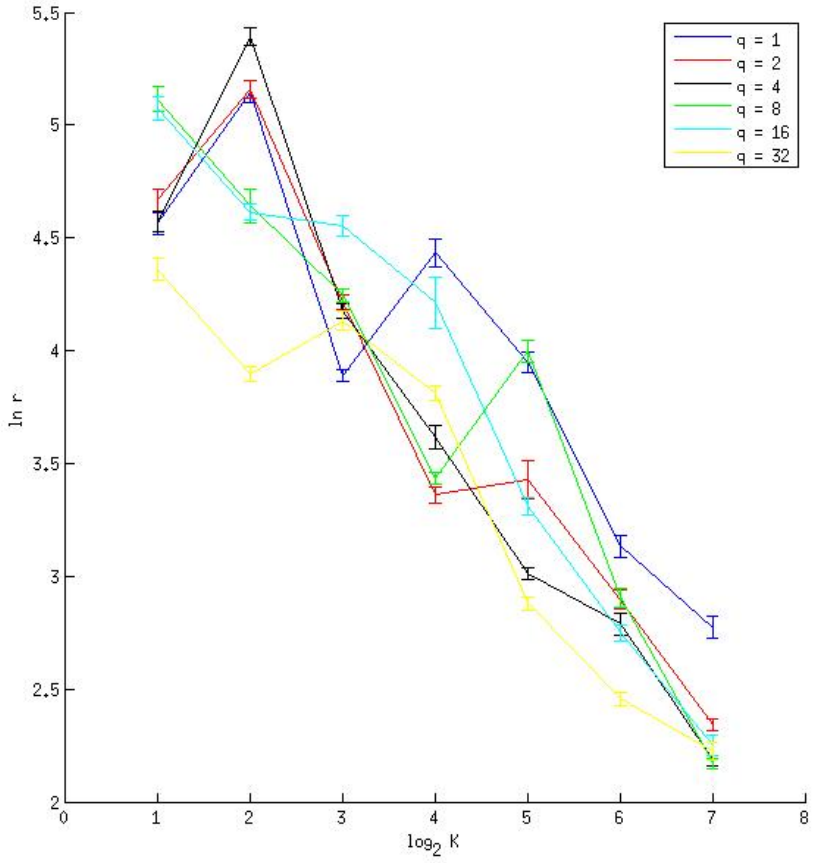


Figure 2-7: $ln(r(X, \lambda_{bg}))$ by number of distributions $K$. For convenience of notation, we extrapolate the notation $q$ and use it to denote the number of Gaussian distributions used to model the foreground distribution in the legend.

### 2.6.5 Context GMM

The experiments in this subsection test the hypothesis of context-specific GMM improving the ratio $r$ w.r.t. general-context GMM.

**EPFL cars.** We compare the ratio $r(\lambda_{bg})$ for each of the contexts of EPFL cars database. We calculate one GMM for each context and one general-context GMM. Fig. 2-10 compares the ratio for all three GMM: in blue, the GMM of the correct context, in red the wrong context (*i.e.*, the GMM calculated on the images of the other context) and in black the general context. We obtain the expected ratio improvements. Indeed, the proper-context GMM consistently perform higher average ratio then both general-context GMM and the GMM for the other context. For context 1, we obtain a clear separation, whereas context 2 has higher standard deviation and the improvement w.r.t. the general-context GMM is smaller.

Figure 2-11 displays the histogram $ln(r)$ for images of context 2. Each histogram

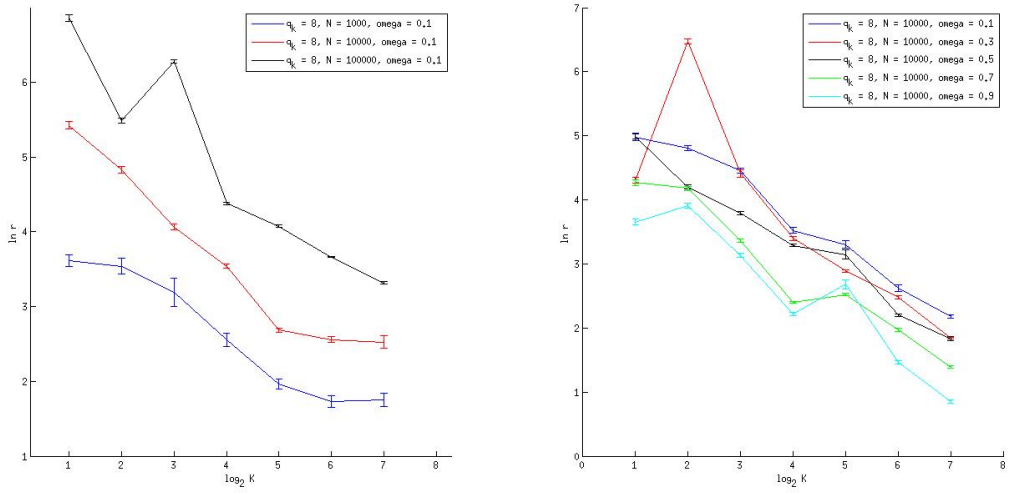


Figure 2-8: Examples of images in context 1.

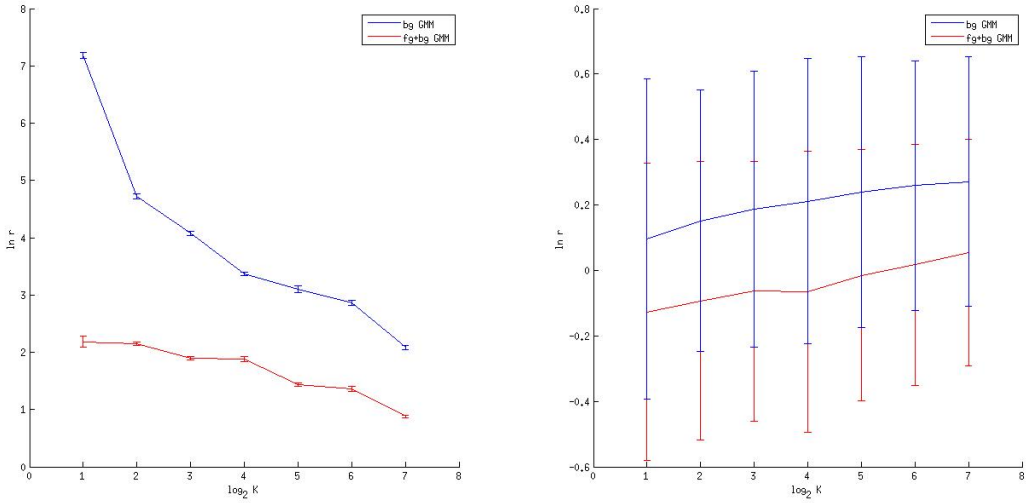


Figure 2-9: Bg GMM vs fg+bg GMM in toy database (left) and EPFL cars (right).

represent the foreground-background ratio for a different Fisher representation. Top-left and top-right of Fig. 2-11 display the histograms of $r(\lambda_{bg})$ for context 2 GMM and general-context GMM, respectively, both for $K = 256$. At the bottom, we display the product of the ratio for context 1 by the normal to the residual to the residual $N_1$, *i.e.*, $\|\Psi_{\lambda_1}(X_{fg})N_1\|/\|\Psi_{\lambda_1}(X_{bg})N_1\|$. Figure 2-11 shows that the product by he normal by the residual did not eliminate the background information, as it was intended. Instead, we observe a higher number of images with foreground-background ratios smaller than 1 and also a higher standard deviation.

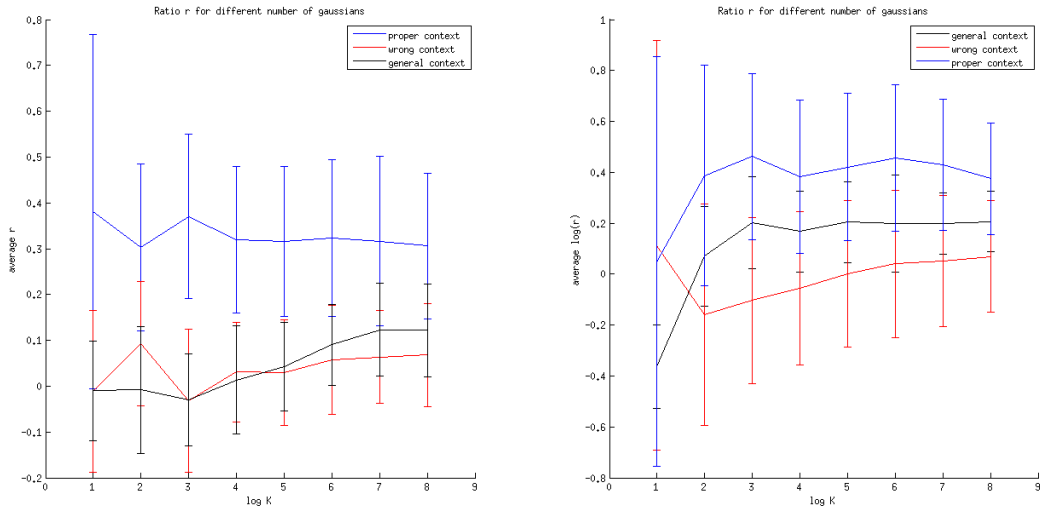


Figure 2-10: Average and standard deviation for foreground-background ratio for contexts of EPFL database. In blue, ratio for proper context GMM; in black, ratio for general context GMM; in red, ratio for wrong context GMM.

**SUn database.** We repeat this experiment for SUn database, that present a higher variability intra-context. We calculate one GMM for each of the four contexts and one general-context GMM, taking descriptors from the background all four contexts. We compare, for each context, $r(\lambda_{bg})$ of its proper context GMM (in blue) and $r(\lambda_{bg})$ of the general-context GMM. Results are shown in Fig. 2-12. Differently for EPFL cars, we observe no improvement of ratio when using context-specific GMM. The general-context GMM obtain similar average ratios (around exp(0.5)) across all contexts. The context-specific GMM for highway and building facade obtain higher average ratios then general-context but within its standard deviation. It is interesting to note that bathroom, despite being the most distinguishable among the contexts (*e.g.*, for being the only indoor scene category), has the least amount of change of ratio from context GMM to general-context GMM.

We conclude from these experiments that the context separation is only significant relative simpler, more uniform scene categories as context. For a complex database such as SUn, the distribution of local descriptors' intra-context is as good for modeling one particular image's background as the distribution of local descriptors from all contexts.

Since the context-specific GMM failed to eliminate background information better than general-context GMM, we do not proceed the application of CFV to classification, as we have no theoretical grounding.

### 2.6.6 Classification with projected Fisher vector

In this subsection, we compare object classification results for FV as implemented in [81] and our projected FV in SUn database. For simplicity, we learn a 16-Gaussian

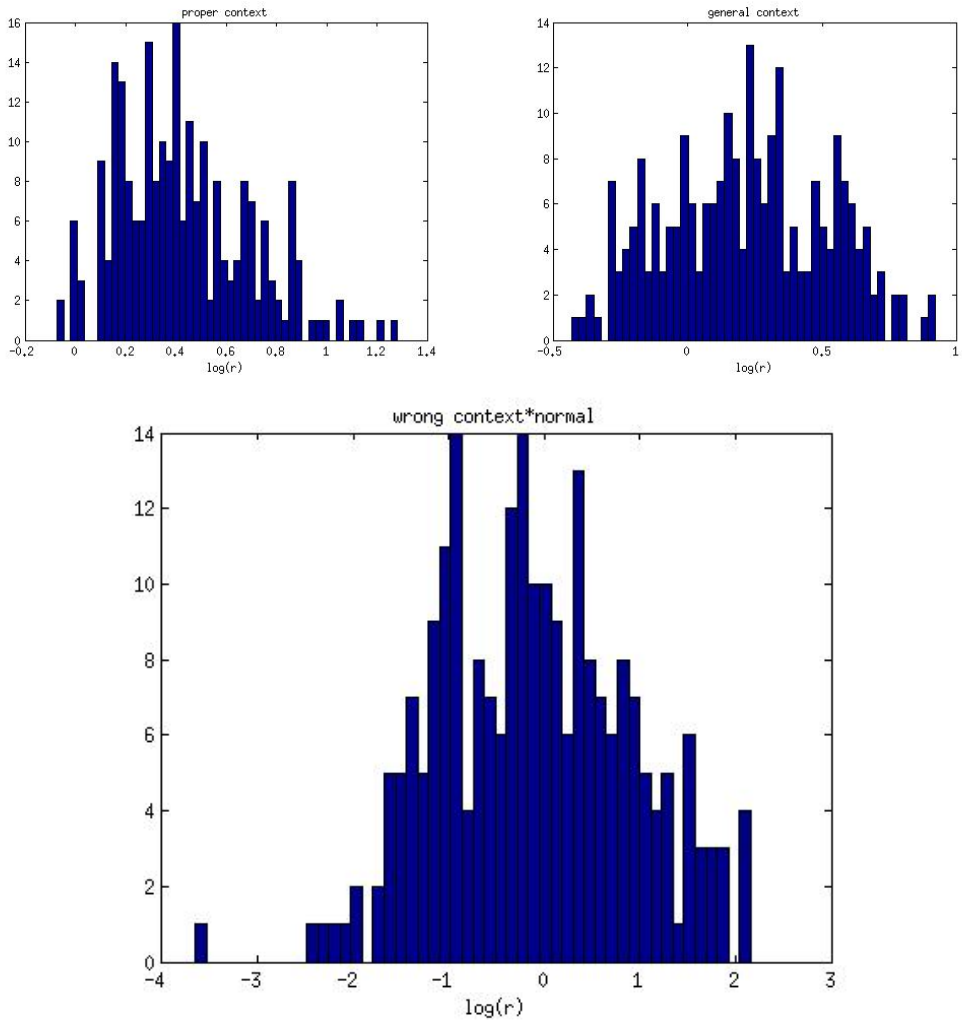


Figure 2-11: Histograms of values for $\ln(r(X, \lambda_{bg})$ for context 2, using: context 2 GMM (top-left), general-context GMM (top-right) and the product of context 1 GMM by the normal to the residual to context 1 (bottom).

distributions GMM. We classify with a linear SVM and compare methods by their average accuracy. For each experiment, we restrict the set of images to the images of a scene category. For example, we classify toilet for images of bathroom and cars for images of highway.

In these experiments, we plot the average accuracy by the dimension $d$ of our projection. Figure 2-13 and Figure 2-14 show the results of the projected FV (in blue) against the results of FV (in black) [3]. The experiments of cars in highway and toilet in bathroom are the only that show the expected result: a curve on $d$ that ascends above the performance of FV at a small dimension (around $d = 300$ for both cases) before descending and coinciding with FV for $d = 2DK$. Figure 2-14 shows

3. The dimension of FV does not change with $d$, we plot it as a straight line to better compare with the projected FV

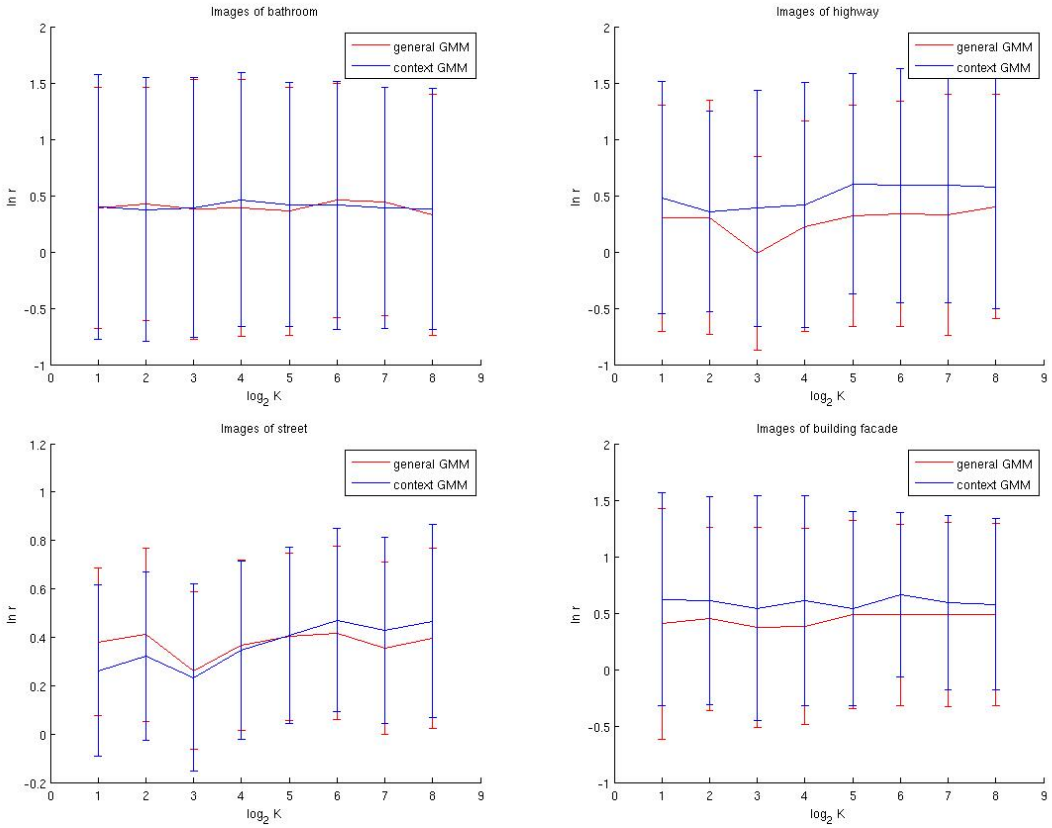


Figure 2-12: Average and standard deviation for foreground-background ratio for contexts of SUn database. In blue, ratio for proper context GMM; in red, ratio for general context GMM. Results for bathroom (top-left), highways (top-right), street (bottom-left) and building facade (bottom-right).

unsuccessful applications of our projection.

## 2.7 Conclusion

We propose a novel framework for Fisher vectors as a context-independent image representation, which is useful for object-based application such as object classification. We start from a statistical formulation present over most of the literature that justifies the elimination of background local descriptors and try to modify our representation in order to better fit this formulation. These modifications consist of:

- Gaussian mixture models that describe only background descriptors instead of full images.
- Projection of Fisher vectors over components that eliminate background information.
- Separation of images in contexts and calculation of one Fisher vector per context for all images, concatenated in a new context-free image representation called

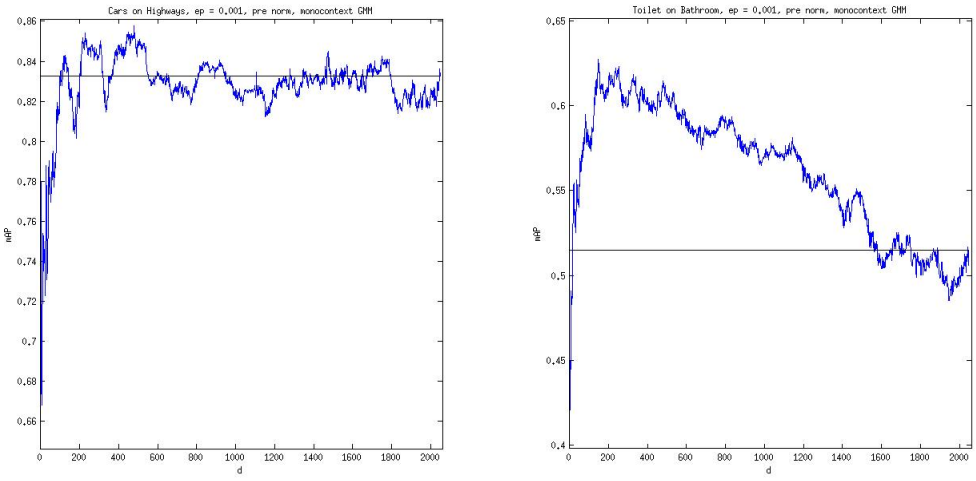


Figure 2-13: From left to right: Cars in highway images and toilets in bathroom images. Black line gives FV mean average precision, blue line gives projected-FV mean average precision for different values of $d$.

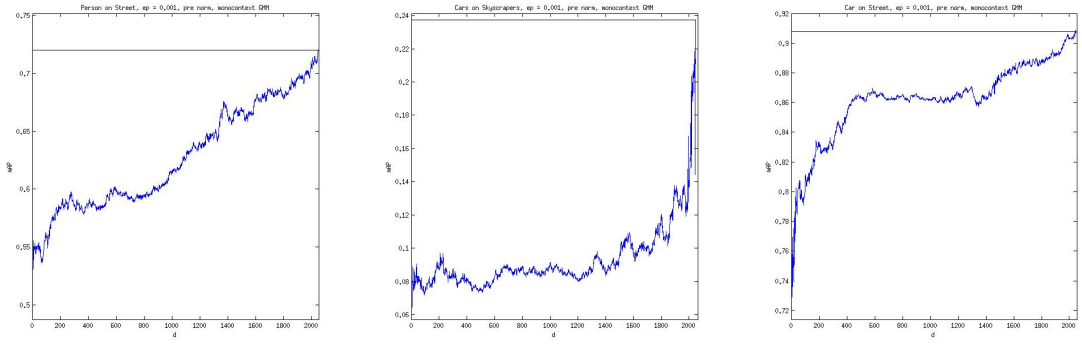


Figure 2-14: From left to right: Persons in street images, cars in skyscraper facade images and cars in street images. Black line gives FV mean average precision, blue line gives projected-FV mean average precision for different values of $d$.

contextual Fisher vector, or CFV.

Although the modifications proposed improve the foreground-background ratio for a hypothetical data that follows all assumptions and simple image contexts, these ratio improvements are not observed for more complex datasets that are coherent with the datasets used for classification today. While we are able to create larger foreground-background ratios with a projected Fisher vector, these do not improve performance for object classification.

The work presented in this chapter presents no published material as we judge our negative results are not fitting to most conferences. However, we believe our experiments present an interesting counter-argument to the formulation of Subsection 2.3.2, as we suppose it consists of an *a posteriori* formulation used to justify the

normalization of Fisher vectors for improving performance in a classification task. Furthermore, the improved performance obtained by addition of background components to our projected Fisher vector suggests that contextual information is indeed useful for object-based classification.

# Chapter 3

# Square-loss Exemplar Machines for Image Retrieval

Zepeda and Pérez [112] have recently demonstrated the promise of the exemplar SVM (ESVM) as a feature encoder for image retrieval. This paper extends this approach in several directions: We first show that replacing the hinge loss by the square loss in the ESVM cost function significantly reduces encoding time with negligible effect on accuracy. We call this model square-loss exemplar machine, or SLEM. We then introduce a kernelized SLEM which can be implemented efficiently through low-rank matrix decomposition, and displays improved performance. Both SLEM variants exploit the fact that the negative examples are fixed, so most of the SLEM computational complexity is relegated to an offline process independent of the positive examples. Our experiments establish the performance and computational advantages of our approach using a large array of base features and standard image retrieval datasets. This chapter is an extended version of the paper "*Square-loss exemplar machines for image retrieval*" by Rezende, Zepeda, Ponce, Bach and Pérez published in CVPR 2017 [79].

## 3.1 Introduction

The exemplar support vector machine (ESVM), originally proposed by Malisiewicz *et al.* [62], leverages the availability of large, unannotated pools of images within the context of supervised learning. It uses a large generic pool of images as a set of negative examples, while using a single image (the *exemplar*) as a positive example. Given these training sets, an SVM classifier is learned that can generalize well, despite the drastically limited size of the set of positive examples. This classifier has successfully been used in classification, object detection and label transfer [63]. Zepeda and Pérez [112] have proposed to treat instead the weights of the resulting classifier as a new feature vector for image retrieval. An ESVM feature is computed for each database and query image, by treating it as the only positive sample while keeping a fixed pool of generic negative images. Searching amounts to computing distances between the query and database ESVM features. Note that ESVM features can be derived from

arbitrary *base* features (*e.g.*, CNN activations) of the exemplar and the images in the generic negative pool.

One drawback of the ESVM feature encoding approach is that computing the classifier requires solving an optimization problem for each positive example (*i.e.*, each query and each database image). This can be time consuming for the large negative pool sizes required for good ESVM feature performance. In this work, we propose using the square loss instead of the hinge loss, in effect converting the ESVM problem into a ridge regression, one that can be solved in closed form. We dub the corresponding classifier a *square-loss exemplar machine* (or *SLEM*). The square loss has been used before to replace the hinge loss in classification tasks (e.g., [88, 107]), and to compare ESVMs to classical classifiers such as the linear discriminant analysis (*LDA*) [55]. In contrast, we propose here to use SLEMs as feature encoders for image retrieval.

Since computing the SLEM features requires inverting a large matrix related to the training set's covariance matrix, we propose an efficient way to compute this inverse. Similarly to the cross-validation method of residual error of [15], we exploit the fact that only a single (positive) example changes in the training set when computing SLEM features for different images. We show experimentally that our representation matches and even improves upon the performance of ESVM features on three standard datasets using a wide range of base features at a fraction of the original computational cost.

The contributions of this work are threefold:

- We introduce a kernelized variant of SLEM that enjoys similar computational advantages and improves retrieval performances.

- We propose a low-rank factorization of the kernel matrix for computational and storage efficiency.

- Our experiments show improved results on a variety of base features and we obtain state-of-the-art results for Inria Holidays dataset.

## 3.2 Background

### 3.2.1 Exemplar classification

The exemplar support vector machine, originally proposed by Malisiewicz *et al.* [62], leverages the availability of large, unannotated pools of images within the context of supervised learning. This classifier has successfully been used in classification, object detection and label transfer [63].

Shrivastava *et al.* [85] incorporates a linear exemplar classifier in an image matching framework. The classifier obtained is used as scoring function, as it is obtained from discriminating the sample positive image (called exemplar) from the pool of

"background" images. Aytar and Zisserman [4, 5] propose an enhanced ESVM framework for pose-based object retrieval. It fits the exemplar classifier to a ridge regression of a learned vocabulary of part-level features. This formulation suppresses false negatives successfully, but its usefulness is constrained by the learning of this vocabulary, which requires more positive data for a supervised learning the parts (or, in case of parts learned from a different dataset, a compatible set of objects) and a geometry-aware feature representation (*i.e.* HOG descriptors [22]).

Zepeda and Pérez [112] have proposed to treat instead the weights of the resulting classifier as a new feature vector for image retrieval. Where previous methods used the exemplar classifier for ranking the established representations, *i.e.* scoring the images by the inner product of its representation and the classifier, this work suggests to use the classifier as the representation: The inner product of two classifiers measures the similarity between the corresponding images.

### 3.2.2 Image retrieval

A suitable image representation for content-based image retrieval must support effective (discriminative) and efficient (fast) comparisons between a query picture and images stored in some large database. These representations must be robust to large image variations due to camera pose, color differences and scene illumination, amongst others.

Many successful approaches to image retrieval rely on unsupervised models of codebook learning, such as $K$-means [23] or Gaussian mixtures [70, 81]. These approaches aggregate local descriptors of an image by weighted average [71], triangular embedding [47] or generalized max-pooling [66] into a global feature descriptor. Before the neural networks *renaissance*, these representations usually outperformed methods that exploit supervised learning of image features directly [10, 75].

Today, with the success of convolutional architectures, global image descriptors are often obtained by aggregating and/or pooling their last convolutional layers, by fine-tuning existing models or by addition of new differentiable layers to an existing architecture. Babenko and Lempitisky [6] introduced the idea of a weighted sum-pooling, called SPoC, of the final layer of the state-of-the-art architecture for image recognition trained on ImageNet [87]. This model is effective and yet very simple. Later works built upon this idea by proposing the learning of the convolutional layers and more sofisticated pooling methods. Kalanditis *et al.* [49] introduced a cross-dimensional weighted (CroW) aggregation, with spatially-localized pooling followed weighted sum-pooling across all channels. Tolias *et al.* [93] encodes several image regions, max-pools and aggregates them into a compact feature. This aggregation method, called RMAC, is choosen by Gordo *et al.* [33] as a pooling method in an end-to-end learned architecture.

Radenović *et al.* [74] fine-tunes a CNN from a large set of unlabeled images. A 3D model of these images is obtained from a Structure from Motion (SfM) method. This model is used to determine the selection of training data for the fine-tuning of the neural network. Finally, Arandjelović *et al.* [1] implements a differentiable variant of a VLAD [23] layer in order to learn end-to-end a robust feature for scene recognition.

The resulting neural network produces state-of-the-art features for some of the most important benchmarks datasets for content-based retrieval. Even if the produced global feature is not as compact as some of the previously mentioned above, a lower dimension representation is obtained by PCA and whitening.

## 3.3 The square-loss exemplar machine

In this section, we revisit the exemplar SVM model proposed in [62] as an instance of a more general family of classifiers. Then, we introduce the square loss exemplar machine (SLEM) as a simple variant of this model and study its properties.

### 3.3.1 Exemplar classifiers

We are given base features in $\mathbb{R}^d$ at training time, one positive example $x_0$ in $\mathbb{R}^d$ and a set of negative examples $X = [x_1, x_2, ..., x_n]$ in $\mathbb{R}^{d \times n}$, each column of $X$ representing one example by a vector in $\mathbb{R}^d$. We are also given a loss function $l : \{-1, 1\} \times \mathbb{R} \to \mathbb{R}^+$. Learning an exemplar classifier from these examples amounts to minimizing the function

$$J(\omega, \nu) = \theta \; l(1, \omega^T x_0 + \nu) + \frac{1}{n} \sum_{i=1}^{n} l(-1, \omega^T x_i + \nu) + \frac{\lambda}{2} \|\omega\|^2, \tag{3.1}$$

w.r.t. $\omega$ in $\mathbb{R}^d$ and $\nu$ in $\mathbb{R}$. In Eq. (3.1), $\lambda$ and $\theta$ are respectively a regularization parameter on $\omega$ and a positive scalar adjusting the weight of the positive exemplar.

Given a cost $l$, we define the corresponding *exemplar classifiers* of $x_0$ with respect to $X$ as the weights $\omega^\star(x_0, X)$ that minimizes the loss function $J$:

$$\left(\omega^\star, \nu^\star\right) = \underset{(\omega,\nu) \in \mathbb{R}^d \times \mathbb{R}}{\arg\min} \; J(\omega, \nu). \; ^1 \tag{3.2}$$

The exemplar SVM [62, 63] is an instance of this model where $l$ is the hinge loss, which is convex. The solution of Eq. (3.2) can thus be found by stochastic gradient descent [11] individually for each positive sample, as it has been done by Malisiewicz *et al.* [62, 63] and Zepeda and Pérez [112]. The next section shows how to calculate all exemplar classifiers simultaneously by changing the loss function.

### 3.3.2 The square loss

Now, let us study the same learning problem for the square-loss function $l(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2$. As in the case of the hinge loss, the minimization of Eq. (3.1) is a convex problem.

1. Depending on the loss function $l$, $\nu^\star(x_0, X)$ may not be unique.

However it is now a ridge regression problem, whose unique solution can be found in closed form as

$$\begin{cases} \omega^\star &= \dfrac{2\theta}{\theta+1} U^{-1}(x_0 - \mu), \\ \nu^\star &= \dfrac{\theta-1}{\theta+1} - \dfrac{1}{\theta+1}(\theta x_0 + \mu)^T \omega^\star, \end{cases} \tag{3.3}$$

where:

$$\begin{cases} \mu = & \frac{1}{n}\sum_{i=1}^n x_i, \\ U = & \frac{1}{n} X X^T - \mu\mu^T \\ & + \frac{\theta}{\theta+1}(x_0 - \mu)(x_0 - \mu)^T + \lambda \mathrm{Id}_d, \end{cases} \tag{3.4}$$

where $\mathrm{Id}_d$ is the identity matrix of size $d$.

**Woodbury identity.** We can simplify Eq. (3.3) by modifying $U$ in Eq. (3.4). Let us define $A = \frac{1}{n} X X^T - \mu\mu^T + \lambda \mathrm{Id}_d$ as the regularized covariance matrix and assume its inverse $A^{-1}$ known. The matrix $U$ now reads $U = A + \frac{\theta}{\theta+1}\delta\delta^T$, where $\delta = x_0 - \mu$ is the centered (w.r.t. the negatives' mean) positive sample. The Woodbury identity [101] gives us

$$U^{-1} = A^{-1} - \frac{\theta}{\theta\delta^T A^{-1}\delta + \theta + 1} A^{-1}\delta\delta^T A^{-1}. \tag{3.5}$$

Substituting (3.5) in (3.3) yields

$$\begin{aligned} \omega^\star &= \frac{2\theta}{\theta+1}\left(A^{-1}\delta - \frac{\theta}{\theta\delta^T A^{-1}\delta + \theta + 1} A^{-1}\delta(\delta^T A^{-1}\delta)\right) \\ &= \frac{2\theta}{\theta\delta^T A^{-1}\delta + \theta + 1} A^{-1}\delta. \end{aligned} \tag{3.6}$$

Equation (3.6) shows how to compute many exemplar classifiers simultaneously, by solving a single linear system in $A$. Also note that the positive sample weight $\theta$ does not influence the direction of the optimal vector $\omega^\star$, only its norm. This means that if search and ranking are based on the normalized feature $\frac{1}{\|\omega^\star\|}\omega^\star$, *e.g.*, using cosine similarity, $\theta$ does not influence the matching score of the SLEM vectors of two different images. This sets SLEM appart from ESVM which requires this parameter to be calibrated [62, 112]. We can thus set the value of $\theta$ to any positive real number.

### 3.3.3 LDA and SLEM

It is interesting to note the relationship between SLEM and the classical linear discriminant analysis (LDA). Let us return to Eq. (3.1) and suppose that we have multiple positive samples. It can be shown that in this case, the corresponding linear classifier of Eq. (3.1) for the square loss is also given by (3.3), where $x_0$ denotes this time the center of mass of the positive samples *if* the positives have the *same*

covariance matrix $\Sigma$ as the negative samples $X$.

This equal-covariance assumption is of course quite restrictive, and probably unrealistic in general. It is interesting to note, however, that this is exactly the assumption made by linear discriminant analysis. As shown in [39] for example, LDA is a (non-regularized) linear classifier with decision function $\omega^T x + \nu$, where

$$\omega = \Sigma^{-1}(x_0 - \mu), \nu = -\frac{1}{2}(x_0 + \mu)^T \omega. \tag{3.7}$$

This shows that, for a single positive sample, SLEM and LDA are very similar: Indeed, taking $\lambda = 0$ (*i.e.*no regularization) and $\theta = 1$, we have $\nu^\star = \nu$, $A = \Sigma$ and that the vectors $\omega$ of Eq. (3.7) and $\omega^\star$ of Eq. (3.6) have the same direction, reducing SLEM to LDA. Many interesting properties of LDA have been used recently for classification tasks [30, 38]. With our simple generalization of LDA, we hope to obtain superior results. The importance of robustness in post-processing methods for image retrieval has been addressed by Jégou and Chum [45].

### 3.3.4 Recursive square loss exemplar machine?

One interesting process proposed in [112] is the recursive exemplar SVM encoding (or RESVM). The proposed idea is to implement the encoding pipeline $k$ times, $k > 0$, taking the output of the $(k-1)$-th ESVM pipeline as input of the $k$-th ESVM pipeline. Here, we consider the base features as the output of the of the 0-th ESVM pipeline. Indeed, if write $\omega^\star$ as a function of the positive exemplar $x_0$ and the pool of negative examples $\mathcal{N} = \{x_1, x_2, ..., x_n\}$ [2], *i.e.*,

$$\omega^\star(x_0, \mathcal{N}) = \underset{\omega \in \mathbb{R}^d}{\arg\min} \left( \theta \; l(1, \omega^T x_0 + \nu^\star) + \frac{1}{n} \sum_{x_i \in \mathcal{N}} l(1, \omega^T x_i + \nu^\star) + \frac{\lambda}{2} \|\omega\|^2 \right), \tag{3.8}$$

and we denote the baseline encodings as $\omega^{(0)} = x_0$ and $\mathcal{N}^{(0)} = \mathcal{N}$, the $k$-th recursion of ESVM is given by $\omega^{(k)}(x_0, \mathcal{N}) = \omega^\star(\omega^{(k-1)}, \mathcal{N}^{(k-1)})$, where $\mathcal{N}^{(k)} = \{\omega^\star(z, \mathcal{N}^{(k-1)} \setminus \{z\}),\ z \in \mathcal{N}^{(k-1)}\}$. Note that the application of the $k$-th pipeline depends on the application of the $(k-1)$-th pipeline on the negative examples. In [112], this means taking the pool of negatives equals to $\mathcal{N} \setminus \{x_i\}$ for each negative example $x_i$. However, for SLEM, it means recalculating the matrix $A$ for each different pool of negative examples. This makes a recursive SLEM much more time costly than a regular SLEM, negating the computational advantages of our approach. Thus, we have not implemented recursive SLEM. One could also imagine keeping the positive example among the negative ones, which would avoid recomputing A at each iteration, but does not make much sense. Indeed, it would mean solving a classification problem where the only positive example is also a negative example.

2. In the remaining of this manuscript, we treated the pool of negatives as a matrix $X$. Exceptionally in this section, we treat the pool as a set in order to facilitate our notation

## 3.4 The kernel SLEM

### 3.4.1 Kernel methods

Let us recall a few basic facts about kernel methods for supervised classification. We consider a reproducing kernel Hilbert space (RKHS) $H$ formed by real functions over some set $X$, and denote by $k$ and $\varphi$ the corresponding reproducing kernel and feature map (which may not admit a known explicit form) over $X$, respectively. We address the following learning problem over $H \times \mathbb{R}$:

$$\min_{h \in H, \nu \in \mathbb{R}} \frac{1}{n} \sum_{i=1}^{n} l(y_i, \langle \varphi(x_i), h \rangle + \nu) + \frac{\lambda}{2} \|h\|_H^2, \tag{3.9}$$

where the pairs $(x_i, y_i)$ in $X \times \{-1, 1\}$, $i = 1 \dots n$ are training samples, and $\langle h, h' \rangle$ is the inner product of element $h$ and $h'$ in $H$. We dub problems with the general form of (3.9) *affine* supervised learning problems since, given some fixed element $h$ of $H$ and some scalar $\nu$, $\langle h, h' \rangle + \nu$ is an affine function of $h'$, whose zero set defines an affine hyperplane of $H$ considered itself as an affine space.

Let $K$ denote the kernel matrix with entries $k_{ij} = \langle \varphi(x_i), \varphi(x_j) \rangle$ and rows $k_i^T = [k_{i1}, k_{i2}, ..., k_{in}]$, $i$ in $\{1, \dots, n\}$. We assume from now on that $l$ is convex and continuous. Under this assumption, Eq. (3.9) admits an equivalent formulation

$$\min_{\alpha \in \mathbb{R}^n,\ \nu \in \mathbb{R}} \left( \frac{1}{n} \sum_{i=1}^{n} l(y_i, k_i^T \alpha + \nu) + \frac{\lambda}{2} \alpha^T K \alpha \right), \tag{3.10}$$

and any solution $(\alpha^\star, \nu^\star)$ to (3.10) provides a solution $(h^\star, \nu^\star)$ to (3.9) with $h^\star = \sum_{i=1}^{n} \alpha_i^\star \varphi(x_i) + \nu^\star$. This result follows from the Riesz representation theorem [83, 98].

Assuming our reproducing kernel is semidefinite positive, $K$ is a semidefinite positive matrix and can be decomposed as $K = BB^T$. Using this factorization, the kernelized problem can be expressed as

$$\min_{\beta \in \mathbb{R}^r, \nu \in \mathbb{R}} \left( \frac{1}{n} \sum_{i=1}^{n} l(y_i, b_i^T \beta + \nu) + \frac{\lambda}{2} \|\beta\|^2 \right), \tag{3.11}$$

where $b_i^T$ denotes the $i$-th row of $B$ and $r$ is the number of columns of $B$. If $(\beta^\star, \nu^\star)$ is the solution of Eq. (3.11), the corresponding vector $\alpha^\star$ (or, more correctly, *a* corresponding vector of dimension $n \geq r$) can be computed by $\alpha^\star = P\beta^\star$, where $P$ is the pseudoinverse of $B^T$.

Note that Eq. (3.11) is written as the usual form of a linear classifier. In particular, it allows us to write the kernel learning problem (3.9) as an instance of Eq. (3.1) by setting $\theta = \frac{1}{n}$ (we set this value of $\theta$ for the remaining of this work) and $y_i = -1$ for all but one training sample. For our approach, we wish to solve (3.11) for many positive exemplars against the same set of negative training samples. In the following subsections, we show how to take advantage of the fixed negative samples to efficiently solve (3.11).

### 3.4.2 Offline preprocessing of negative samples

Let us now return to the (kernelized) SLEM, taking $l$ as the square loss. In order to calculate offline all operations that are dependent only on negative samples, let us denote by $K$ the kernel matrix of the negative samples $X$. The preprocessing phase consists of the calculation of the decomposition $B$ and the constants of Eq. (3.6):

$$\begin{cases} \mu = \frac{1}{n}\sum_{i=1}^{n} b_i^T, \\ A = \frac{1}{n} B^T B - \mu\mu^T + \lambda \mathrm{Id}_r. \end{cases} \tag{3.12}$$

These operations are done offline and their results are stored.

### 3.4.3 Online addition of a positive sample

We now wish to write Eq. (3.11) as an exemplar classifier, with one positive example $x_0$ and $n$ negative examples $X$. We denote by $K'$ the augmented kernel matrix obtained by adding this sample,

$$K' = \begin{bmatrix} k_{00} & k_0^T \\ k_0 & K \end{bmatrix}, \tag{3.13}$$

where $k_{00} = \langle \varphi(x_0), \varphi(x_0) \rangle$ is a scalar and $k_0 = [\langle \varphi(x_0), \varphi(x_i) \rangle]_{1 \leq i \leq n}$ is a vector in $\mathbb{R}^n$. The following lemma shows how the factorization of $K'$ can be derived from the factorization of its sub-matrix $K$ and the solution of a $n \times n$ linear system.

**Lemma 1.** *The augmented kernel matrix $K'$ can be factorized as $K' = B'B'^T$ with*

$$B' = \begin{bmatrix} u & v^T \\ 0 & B \end{bmatrix}, \; v = B^\dagger k_0, \; u = \sqrt{k_{00} - ||v||^2}, \tag{3.14}$$

*where $B^\dagger$ is the pseudoinverse of $B$.*

*Proof.* For $B'$ defined by (3.14), we have that

$$B'B'^T = \begin{bmatrix} u^2 + \|v\|^2 & v^T B^T \\ Bv & BB^T \end{bmatrix} = \begin{bmatrix} k_{00} & v^T B^T \\ Bv & K \end{bmatrix}. \tag{3.15}$$

Since $K'$ is positive semidefinite, $k_0$ must lie in the column space $\mathcal{B}$ of $B$. Indeed, if we suppose $k_0$ does not belong to $\mathcal{B}$, then it can be decomposed uniquely as $k_0 = s + t$, $s \in \mathcal{B}$ and $t \in \mathcal{B}^\perp$, with $t \neq 0$. In one hand, $K'$ being semidefinite positive implies that $[1, -at^T] K' [1; -at] = k_{00} - 2a\|t\|^2 \geq 0$ for all real value $a$. Here, we use matlab notation for horizontal and vertical staking. In the other hand, for $a$ large enough, $k_{00} - a\|t\|^2 \leq 0$, which is a contradiction. Hence $v = B^\dagger k_0$ is an exact solution of $Bv = k_0$. The fact that $k_{00} - \|v\|^2$ is non-negative comes from the fact that the Schur complement $K - k_0 k_0^T / k_{00}$ of $k_{00}$ in $K'$ is itself positive semidefinite. Indeed, since

the matrix $k_{00}K - k_0k_0^T = B(k_{00}\mathrm{Id}_r - vv^T)B^T$ is also positive semidefinite. Thus $v^T(k_{00}\mathrm{Id}_r - vv^T)v = \|v\|^2(k_{00} - \|v\|^2) \geq 0$. □

This lemma allows us to add a positive sample to Eq. (3.11). With a positive exemplar, it now reads

$$\frac{1}{n}(b_0'^T\beta + \nu - 1)^2 + \frac{1}{n}\sum_{i=1}^{n}(b_i'^T\beta + \nu + 1)^2 + \frac{\lambda}{2}\|\beta\|^2, \tag{3.16}$$

with $b_i'^T$ being the $(i+1)$-th row of $B'$, $i$ in $\{0, 1, ..., n\}$. In particular, $b_0' = [u;\ v]$ and, for $i > 0$, $b_i' = [0;\ b_i]$. The solution $(\beta^\star, \nu^\star)$ in $\mathbb{R}^{r+1} \times \mathbb{R}$ can be computed just as before by Eq. (3.3), replacing $x_0$ by $b_0'$, $\mu$ by $\mu' = \frac{1}{n}\sum_{i=1}^{n} b_i'$ and $X$ by the $(r+1) \times n$ matrix $Q$ of columns $b_1', b_2', ..., b_n'$. The solution $\alpha^\star$ is now calculated as $\alpha^\star = P'\beta^\star$, where $P' = [u^{-1}\ 0^T; -u^{-1}Pv\ P]$ is the pseudoinverse of $B'^T$. $\alpha^\star$ can be expressed by the linear system

$$\begin{bmatrix}\alpha_0 \\ \hat{\alpha}\end{bmatrix} = \begin{bmatrix}\frac{1}{u} & 0^T \\ -\frac{1}{u}Pv & P\end{bmatrix}\begin{bmatrix}\beta_0 \\ \hat{\beta}\end{bmatrix}. \tag{3.17}$$

### 3.4.4 Similarity score

Once the optimal parameters $(\beta, \nu)$ from (3.16) and the coordinates $u$, $v$ of $b_0'$ from (3.14) have been found [3], they can be used directly for measuring similarity between matching images.

Suppose two image descriptors $x_0$ and $x_0'$ are given and we wish to calculate the similarity score between their SLEM representations $h$ and $h'$, denoted by $s(h, h')$. We write $h' = \alpha_0'\varphi(x_0') + \sum_{i=1}^{n}\alpha_i'\varphi(x_i) + \nu'$. Using Eq. (3.17) and ignoring biases $\nu$ and $\nu'$ which have empirically no influence, $s(h, h')$ is given by:

$$\begin{aligned} s(h, h') &= \langle h, h' \rangle \\ &= \hat{\alpha}^T K\hat{\alpha}' + \alpha_0 k(X, x_0)^T\hat{\alpha}' + \alpha_0' k(X, x_0')^T\hat{\alpha} \\ &\quad + \alpha_0\alpha_0' k(x_0, x_0') \\ &= \hat{\beta}^T\hat{\beta}' + \lambda^{-2}(k(x_0, x_0') - v^Tv). \end{aligned} \tag{3.18}$$

For a given image whose descriptor is $x_0$, we need to store $x_0$, $\hat{\beta}$ and $v$ to calculate its similarity score to whichever other image for SLEM. Since we assume the base feature $x_0$ has dimension $d$ and $\hat{\beta}$ and $v$ each have dimension $r$, we store a vector of dimension $d + 2r$ for each image.

## 3.5 Efficient implementation

When compared to the linear square-loss classifier of Section 3.3.2, one drawback of the kernelized approach is that the dimension of our problem grows with the number

3. We drop the "$\star$" in this subsection to avoid cluttering the notation.

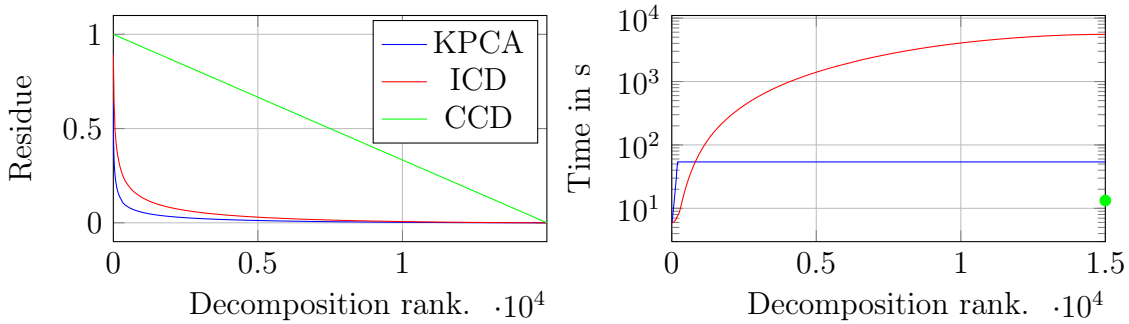


Figure 3-1: Comparison between complete Cholesky decomposition (CCD, in green), incomplete Cholesky decomposition (ICD, in red) and kernel PCA (KPCA, in blue), varying the rank of $B$. Left: Residue for each decomposition. Right: Time of calculation. We use SPoC features [6] of 15000 sample images to construct the the kernel matrix $K$.

$n$ of negative samples. The offline factorization $BB^T$ of $K$ demands $O(nr)$ storage and $O(nr^2)$ time. This factorization can be obtained in two ways: full-rank and low-rank decomposition. In this section we propose three different decompositions of $K$ and discuss their respective merits.

### 3.5.1 Full-rank decomposition

**CCD:** The complete Cholesky decomposition (CCD) is the most used factorization of positive-definite matrices in kernel-based learning due to its time efficiency [8]. We use it as our default decomposition. We make sure $K$ is positive-definite by adding $\epsilon$ to its diagonal, where $\epsilon = \min(0, -\lambda_{min})$ and $\lambda_{min}$ is the smallest eigenvalue of $K$ [4]. Therefore, $B$ also has rank $n$ and can be calculated by CCD from the identity $BB^T = K + \epsilon \mathrm{Id}_n$.

### 3.5.2 Low-rank decomposition

One of the major constraints of large scale retrieval is the minimization of storage. As discussed in Section 3.4.4, for each database image we store its base representation plus a $2r$ vector. Hence, we aim to decompose $K$ at a small rank $r$. Two classical methods can be used to obtain a low-rank decomposition of K.

**ICD:** The incomplete Cholesky decomposition (ICD) is widely used in machine learning [8, 28]. It is similar to CCD, and greedily chooses which column of $K$ to add to the decomposition based on the gain in approximation error [9]. The algorithm stops after $r$ steps, obtaining the factor $B$ in time $O(nr^2)$.

4. $K$ is, by the definition of a kernel matrix, a positive-semidefinite matrix and so its eigenvalues are non-negative. In practice, when we compute $K$ for a Gaussian kernel, a few eigenvalues are negative and of absolute value smaller than $10^{-8}$, due to floctuations of the Gaussian variables. We set $\epsilon = 10^{-8}$ to avoid the recalculation $\lambda_{min}$.

**KPCA:** Kernel PCA (KPCA) [84] computes the factor $B$ by performing a singular value decomposition of $K$ (truncated singular value decomposition for very small values of $r$), and making each column of the factor correspond to one of the top $r$ singular vectors. The resulting matrix $B$ is guaranteed to be the best $r$-rank approximation of $K$ according to the Frobenius norm. The computational cost of KPCA is, however, $O(n^2 r)$ [32].

We plot in Fig. 3-1 the residue $\mathrm{tr}(K - BB^T)/\mathrm{tr}(K)$ when we vary the number of columns $r$ of $B$ and we see a faster convergence for the kernel PCA decomposition when $r$ goes to $n$. We also compare their computation time, which shows that KPCA is slower than ICD for small values of $r$ and faster for values of $r$ such that the residue is small.

When comparing computation time, KPCA is slower than ICD for small values of $r$ and faster for values of $r$ such that the residue is small. Also, as discussed above, KPCA gives a smaller residue $\mathrm{tr}(K - BB^T)/\mathrm{tr}(K)$. From these comparisons, we set KPCA as our default low-rank decomposition. The only case ICD is more appropriated than KPCA is for very large number of negatives $n$, for which the time complexity of KPCA becomes an issue, and very small rank $r$. This particular case is further studied in Section 3.6.5.

## 3.6 Experimental evaluation

### 3.6.1 Datasets and evaluation protocol

We perform experiments on three standard datasets for image retrieval.

- The INRIA *Holidays* dataset [46] consists of 1491 images divided in 500 groups of matching images. We manually rotate by 90 degrees some images that are not in their natural orientation to compensate for the fact that CNN features are not rotation invariant [1, 7, 33, 49, 74].
- The *Oxford5k* dataset [72] consists of 5063 images separated in 55 groups of matching images, each group associated to a landmark of Oxford. We use the "full" crop, ignoring the region of interest of each image.
- The *Oxford105k* dataset [72] is a large-scale dataset containing the same images and queries from Oxford5k plus *Flickr100k*, a collection of $10^5$ distractor Flickr images.

Figure 3-2: Examples of images in INRIA Holidays dataset. The center image is rotated 90 degrees clockwise to their natural orientation.

Figure 3-3: Examples of images in Oxford buildings dataset.

As pool of negative images to build SLEMs, we use the Flickr100k for both Holidays and Oxford5k. When evaluating Oxford105k, where Flickr100k is part of the database, we us instead the *Paris* dataset [73] as negative samples.

### 3.6.2 Kernels

For the experiment of SLEM as a feature encoder, we have tested two different kernels, each with a scalar parameter $\gamma$.
**Gaussian SLEM:**

$$k_1(x, y) = e^{-\gamma\|x-y\|^2}; \tag{3.19}$$

**Poly SLEM:**

$$k_2(x, y) = x^T y + \gamma(x^T y)^2. \tag{3.20}$$

Note the non-kernel version of SLEM is equivalent to a SLEM for the linear kernel. In order to distinguish it from other versions, we will refer to the non-kernel version as **Linear SLEM** for the remaining of this work.

### 3.6.3 Base visual features

We test our feature encoder for four different base features: the hand-crafted VLAD image representation and three learned features derived from the activation coefficients of deep Convolutional Neural Networks.

We use the same VLAD variant of [23] used in [112] that relies on densely-extracted rootSIFT [2] local descriptors, per-cluster normalization, PCA-based rotations, and root normalization. Like [112], we use 64 clusters, for a final feature of size 8192.

The first CNN features we use consist of the activation coefficients of the previous-to-last layer of the AlexNet architecture [56], based on a publicly available pre-trained model [48]. These are also the features used in [112].

The SPoC features [6], which are tailored specifically for the image retrieval application, consist of spatially-weighted sums of the activations of the last convolutional layer of the 19-layer VGG network [87].

Finally, we use the NetVLAD features [1], trained for place recognition. These features are obtained by adding a differentiable version of the VLAD algorithm [23] as a layer at the end of a convolutional architecture.

| Dataset | **Holidays** | | | | **Oxford 5k** | | | | **Oxford 105k** | |
|---|---|---|---|---|---|---|---|---|---|---|
| Model, features | VLAD | SPoC | AlexNet | NetVLAD | VLAD | SPoC | AlexNet | NetVLAD | SPoC | NetVLAD |
| Baseline | 72.7 | 76.5 | 68.2 | 85.4 | 46.3 | 54.4 | 40.6 | 67.5 | 50.1 | 65.6 |
| PCAW | 75.5 | 81.7 | 69.2 | 88.3 | 50.9 | 63.7 | 45.0 | 69.1 | 55.5 | 66.1 |
| LDA | 54.7 | 82.2 | 64.1 | 74.3 | 29.6 | 62.2 | 42.5 | 72.7 | 52.4 | 40.7 |
| ESVM [112] | $77.5^{3}$ | $84.0^{3}$ | 71.3 | $91.4^{2}$ | $57.2^{3}$ | 62.1 | 43.9 | 72.5 | 56.5 | 67.5 |
| Linear SLEM | $78.0^{2}$ | 82.3 | 72.1 | $91.3^{3}$ | **59.3** | $64.1^{3}$ | $46.2^{3}$ | $72.9^{3}$ | $56.7^{3}$ | $68.0^{3}$ |
| Gaussian SLEM (16) | 76.8 | 80.3 | 71.2 | $91.4^{2}$ | 52.8 | 63.0 | 43.5 | 71.9 | 55.8 | 67.4 |
| Gaussian SLEM (32) | 77.4 | 81.7 | $72.0^{3}$ | $91.4^{2}$ | 54.9 | 63.1 | 44.0 | 71.1 | 56.0 | 67.8 |
| Gaussian SLEM (fr) | **78.1** | $86.2^{2}$ | **72.9** | **91.7** | $59.0^{2}$ | **64.9** | $47.0^{2}$ | **74.4** | $59.5^{2}$ | $70.0^{2}$ |
| Poly SLEM (16) | 76.9 | 82.3 | 71.4 | $91.3^{3}$ | 53.0 | 63.6 | 43.6 | 71.4 | 56.1 | 67.5 |
| Poly SLEM (32) | 77.3 | 82.4 | $72.1^{2}$ | **91.7** | 54.9 | 63.6 | 44.1 | 71.6 | 56.3 | 67.9 |
| Poly SLEM (fr) | **78.1** | **86.3** | **72.9** | **91.7** | **59.3** | $64.8^{2}$ | **47.3** | $74.1^{2}$ | **62.5** | **70.2** |

Table 3.1: Mean average precision (mAP) results for INRIA Holidays and Oxford buildings datasets, expressed as percentages. In this table, we present our results for VLAD [23], sum-pooling of convolutional features (SPoC) [6], activation coefficients from the previous-to-last CNN layer (AlexNet) [56] and activation of NetVLAD layer [1]. In parentheses, the rank of he decomposition ('fr' for full rank decomposition). For each column, we show in **bold** the best results and index the second and third best.

### 3.6.4 Image retrieval results

We use all the those base features of the previous subsection as baseline. Since Babenko and Lemptisky [6] and Arandjelović *et al.* [1] have improved retrieval results by applying PCA followed by whitening to their features, we also apply this post-processing to our base features as a second baseline (PCAW), compressing base feature dimension to half of the original. We then compare the baselines with the original ESVM, LDA and several variants of our approach (SLEM), since all the those methods are based on similar ideas. For LDA, we follow our description in section 3.3.3 and calculate $\Sigma$ as the convariance of the negative samples. The results are presented in Table 3.1. For the large-scale dataset that is Oxford105k, we limit our experiments to our best performing base features, SPoC and NetVLAD.

The version of each feature used as *baseline* changes according to the dataset. For VLAD and AlexNet, we considere the version presented in [112] as baseline and a rotated, whitened and compressed to half its original dimensions as PCAW. For SPoC, we apply LDA, ESVM and SLEM to the baseline 512 dimensions feature, as a replacement of PCAW used in [6] For NetVLAD, we use the rotated and compressed version of the trained networks correspondent to each dataset as suggested by [1]. For Holidays we obtain better results with whitened NetVLAD features and for Oxford5k, non-whitened.

Linear SLEM performs similarly to ESVM while being much more time efficient (Fig. 3-4). The fact that a hinge-loss classifier does not outperform a square-loss classifier can seem counter-intuitive, but both have been shown to be equivalent for

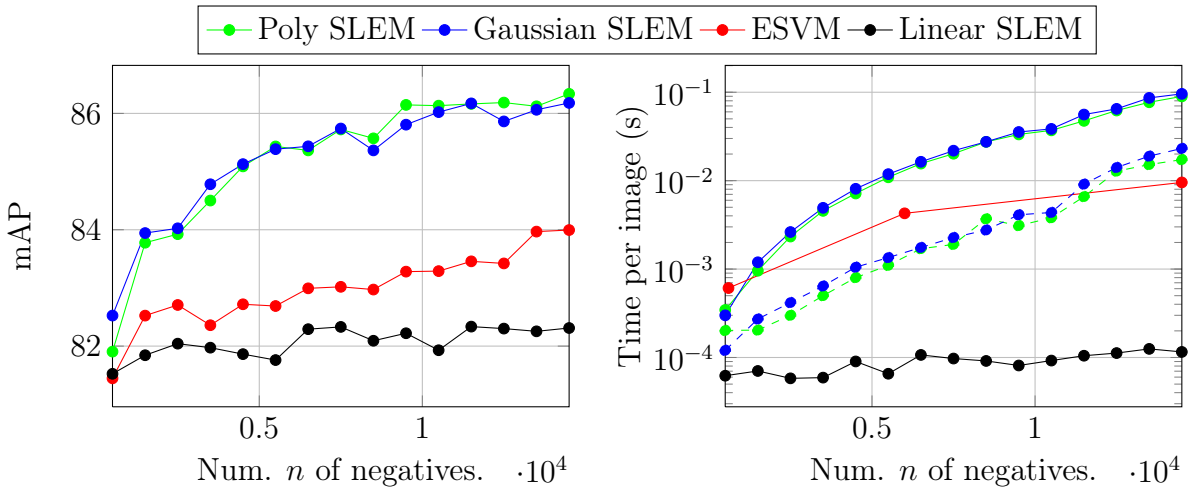


Figure 3-4: Results for INRIA Holidays, using SPoC features and different variants of full-rank SLEM. We use $T = 10^5$ iterations for all $n$ to report mAP for ESVM, as suggested by [112], but report timings using $T = 1.66n$ and the values reported in Table 1 of [112]. Left: mAP; Right: computation time in solid line, *online* computational cost in dashed line.

binary classification under mild constraints [107].

We use both Gaussian SLEM and Polynomial SLEM with two decompositions: one full-rank CCD decomposition indicated by (fr) and two low-rank KPCA decompositions indicated by the rank of the decomposition. We train our exemplar classifiers for 15000 negative samples. For all the experiments we calibrate the regularization cost $\lambda$, as well as the parameter $\gamma$ similarly to the calibration in [112].

The full-rank variant outperforms all methods for all base features, although the gains when compared to linear SLEM are not always significant (*e.g.* for VLAD features). We notice significant improvement for SPoC in both Holidays and Oxford and for AlexNet and NetVLAD in Oxford.

### 3.6.5 Time and storage scalability

In this section we compare the time efficiency of our method and the ESVM, as well as discuss which method and decomposition to use according to the number of negative samples.

In Fig. 3-4, we see that the linear SLEM efficiency does not change with $n$. Indeed, if $d$ is the dimension of the base representation, $A$ is a $d \times d$ matrix for linear SLEM, whereas for a full-rank kernel, $A$ is $n \times n$. This explains the increasing running time for Gaussian and polynomial kernels: storage and solving a $n \times n$ system does not scale for large number of negative samples.

Retrieval results for full-rank kernelized SLEM in Fig. 3-4 suggest we can benefit from larger sets of negative samples. We, however, limit our full-rank experiments to $n = 15000$ negative samples due to the $O(n^3)$ complexity of the offline step. When we consider only the online procedure of our model, *i.e.* the calculation of $\beta^\star$, our

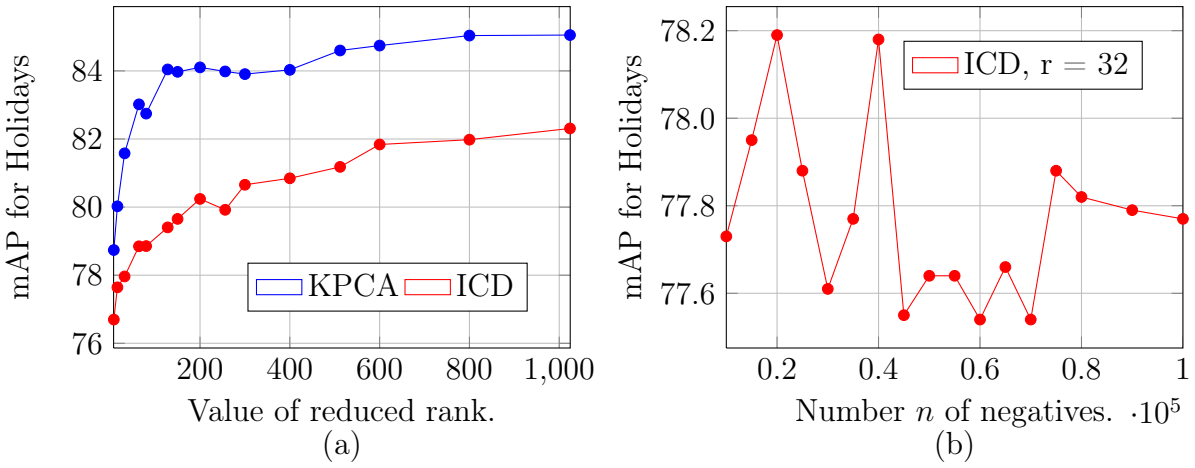


Figure 3-5: mAP for Holidays using SPoC + Poly SLEM for different low-rank decompositionsand value of ranks: At (a), $n = 15000$ negatives. We perform two low-rank decompositions and compare its results at similar ranks. At (b), rank fixed at $r = 32$. At this high number of negative images, KPCA is constraint by its computation time, making ICD the only viable low-rank decomposition.

kernelized model has a similar time efficiency to ESVM. Therefore, we can process the kernel SLEM for the Gaussian and polynomial kernels in similar running time to ESVM if we pre-process our negative samples offline. For low-rank decompositions, we present in Fig. 3-5(a) a comparison in average precision between KPCA and ICD decompositions using SPoC on the Holidays dataset, fixing $n$ and varying $r$. The superior results justify our preference for KPCA, despite its less efficient offline step. The only advantage of ICD over KPCA is its time complexity, linear in the number of negative samples, that allows a bigger number of negative samples. In Fig. 3-5(b) we show results for ICD for bigger pools of negative samples, to which a KPCA decomposition would be too time consuming. The results suggest that the performance of ICD SLEMs are not sensitive to the number of negative examples at a fixed small rank.

As shown by Fig. 3-5(a), the mAP for the low-rank KPCA approximation increases with the rank. Its maximum value for the features of the figure is 86.3 for full rank, $r = 15,000$ (Table 3.1, col. 2). Figure 3-5(a) also shows, however, that reasonable mAP values (around 84) are obtained for a much smaller rank of 200. In keeping with the usual practice in image retrieval, we limit even further the rank in the results shown in Tables 3.1 and 3.4, with very small ranks (16 and 32) that yield a total feature dimension similar to the base representation, and allow a direct comparison to methods using these features directly. This rather extreme compression is also justified in part by the fact that these very-low rank factorizations already capture a reasonable part of the problem structure. Indeed, the relative residual error for SPoC features on Holiday with 15,000 negatives is only 0.39 for $r = 16$ and 0.31 for $r = 32$. For reference, the relative error decreases to 0.08 for $r = 600$, and 0.05 for $r = 1024$.

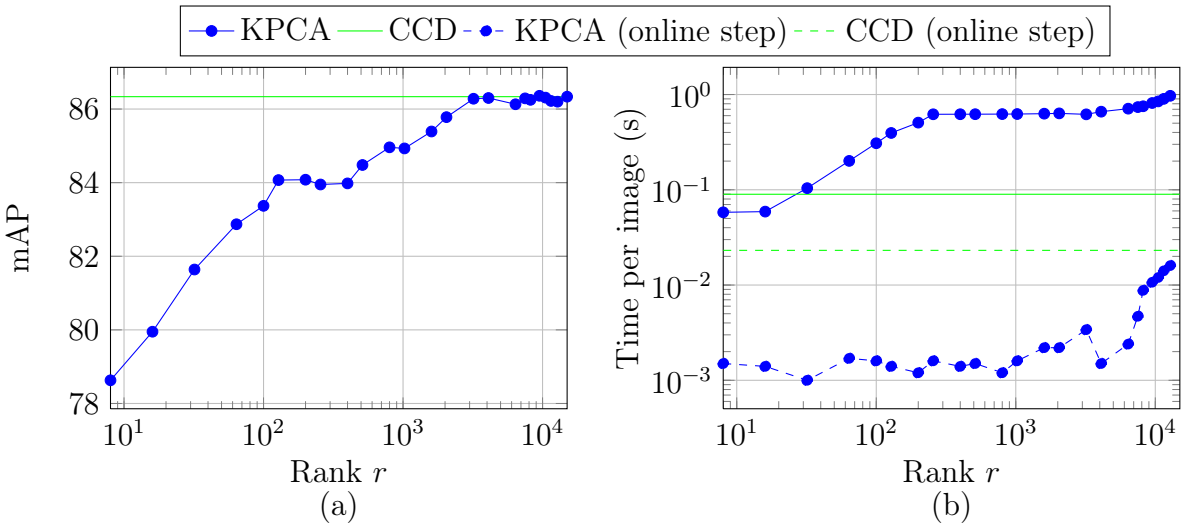


Figure 3-6: Comparison of results for low-rank decomposition of SLEM, varying the rank $r$ (in blue) with full-rank SLEM (in green). Results are shown for INRIA Holidays, using SPoC features and 15000 negative samples; At (a), mAP; at (b), computation time in solid line, *online* computational cost on dashed line.

### 3.6.6 Low-rank decomposition evaluation

We present a more complete evaluation of low-rank decomposition kernel SLEM for higher ranks. In Section 3.6.4, we limit our experiments to ranks equal to 16 and 32 for a best comparison to the state of the art. In this section, we use SPoC features as our base encoding, the polynomial kernel as our kernel function, Holidays dataset images as positives and $n = 15000$ images of Flickr100k as negative samples. The results for a low-rank KPCA decomposition are presented in Fig. 3-6, varying the rank $r$ through a higher range of values, and compared to the full-rank Cholesky decomposition. As $r$ increases, average precision increases and quickly approaches the full-rank performance. As reference, we obtain mAP 80.0 for $r = 16$, 84.0 for $r = 128$, 85.4 for $r = 1600$, 86.2 for $r = 3200$ and to the mAP 86.3 for full rank. However, the computational cost eventually overcomes that of the full-rank method. This is due to the higher complexity of the singular value decomposition. For values of $r$ bigger than 64, a KPCA is more time consuming than CCD.

As seen in 3.1, a low-rank decomposition of Poly SLEM obtain the same results as the full rank for NetVLAD features. We thus expand this experiment to higher ranks. The results are shown in Table 3.2.

### 3.6.7 Spatial pyramid matching kernel

All previous kernel functions take as input image representations in a fixed sized linear space. In this section, we propose using the spatial pyramid kernel [34], that takes as input a set of local descriptors and its coordinates in the image.

We revisit the spatial pyramid scheme presented in [57] using the generalized

| Features | rank | dim | **Hol.** | **Ox5k** |
|---|---|---|---|---|
| Arandjelović *et al.* [1] | - | 4096 | 88.3 | 69.1 |
| NetVLAD + Linear SLEM | - | 4096 | 91.3 | 72.9 |
| NetVLAD + Poly SLEM | 16 | 4128 | 91.3 | 71.2 |
| NetVLAD + Poly SLEM | 32 | 4160 | 91.7 | 71.7 |
| NetVLAD + Poly SLEM | 64 | 4224 | 91.3 | 71.9 |
| NetVLAD + Poly SLEM | 128 | 4352 | 91.2 | 72.1 |
| NetVLAD + Poly SLEM | 256 | 4608 | 91.0 | 71.9 |
| NetVLAD + Poly SLEM | 512 | 5120 | 90.8 | 72.5 |
| NetVLAD + Poly SLEM | 1024 | 6144 | 91.1 | 73.1 |
| NetVLAD + Poly SLEM | 15000 | 304096 | 91.7 | 74.1 |

Table 3.2: Completed version of the state of the art table of our submission, with higher-rank decompositions.

| Dataset | **Holidays** | | **Oxford 5k** | |
|---|---|---|---|---|
| Method, feaures | rootSIFT | CNN | rootSIFT | CNN |
| Baseline (SPM) | 53.3 | 51.8 | 24.1 | 36.4 |
| SP SLEM | 66.9 | 70.2 | 42.4 | 45 |

Table 3.3: Results for spatial pyramids kernel when compared with a spatial pyramid matching from the local descriptors

intersection function of histograms [12]. We test this kernel for rootSIFT extracted in multiscales, as used in [57], and learn 200 codewords with K-means. Inspired by the results of [40], we also test it for activations of the last convolutional layer of the network of [87]. Each neuron of the last layer correspond to a codeword. The results are presented in Table 3.3. As baseline, we compare with the spatial pyramid matching model presented in [57]. The improvements obtained with respect to the baseline are significantly larger then others kernels, and its use must be further studied.

### 3.6.8 Comparison to the state-of-the-art

We compare the state-of-the-art global descriptors for Holidays and Oxford 5k to both SPoC and NetVLAD features improved by linear SLEM and low-rank Poly SLEM in Table 3.4. We do not include re-ranking nor query expansion. We perform PCA and whitening to compress both descriptors to 256 and 512, as done in [1, 6] and compare the results by brackets of dimension. We also add a bracket of the full 4096-dimension NetVLAD for completeness, so we include our best performance. Our approach outperforms the state of the art for Holidays by 2.5 points in 256 dimensions and 0.8 in 512 dimensions, despite not using the best performing descriptors [33] as base features.

| Features | rank | dim | **Hol.** | **Ox5k** |
|---|---|---|---|---|
| Babenko *et al.*[6] | - | 256 | 80.2 | 58.9 |
| Radenović *et al.* [74] | - | 256 | 81.5 | <u>77.4</u> |
| Arandjelović *et al.* [1] | - | 256 | 86.0 | 62.5 |
| Kalantidis *et al.* [49] | - | 256 | 83.1 | 65.4 |
| SPoC + Linear SLEM | - | 256 | 81.5 | 64.7 |
| SPoC + Poly SLEM | 16 | 288 | 80.1 | 63.6 |
| SPoC + Poly SLEM | 32 | 320 | 81.8 | 63.6 |
| NetVLAD + Linear SLEM | - | 256 | <u>88.5</u> | 65.9 |
| NetVLAD + Poly SLEM | 16 | 288 | 87.7 | 65.5 |
| NetVLAD + Poly SLEM | 32 | 320 | 88.3 | 65.6 |
| Radenović *et al.* [74] | - | 512 | 82.5 | 79.7 |
| Arandjelović *et al.* [1] | - | 512 | 86.7 | 65.6 |
| Kalantidis *et al.* [49] | - | 512 | 84.9 | 68.2 |
| Gordo *et al.* [33] | - | 512 | 89.1† | **83.1**† |
| SPoC + Linear SLEM | - | 512 | 82.3 | 64.1 |
| SPoC + Poly SLEM | 16 | 544 | 82.3 | 63.0 |
| SPoC + Poly SLEM | 32 | 576 | 82.4 | 63.1 |
| NetVLAD + Linear SLEM | - | 512 | 89.3 | 72.3 |
| NetVLAD + Poly SLEM | 16 | 544 | <u>89.9</u> | 71.9 |
| NetVLAD + Poly SLEM | 32 | 576 | <u>89.9</u> | 72.3 |
| Arandjelović *et al.* [1] | - | 4096 | 88.3 | 69.1 |
| NetVLAD + Linear SLEM | - | 4096 | 91.3 | 72.9 |
| NetVLAD + Poly SLEM | 16 | 4128 | 91.3 | 71.2 |
| NetVLAD + Poly SLEM | 32 | 4160 | **91.7** | 71.7 |

Table 3.4: Compared results to state-of-the-art features at similar dimensions, without re-ranking or query augmentation. The results using Poly SLEM add 32 or 64 dimensions to the original feature (for $r = 16$ or $r = 32$, respectively). Underlined results are the best at each dimension bracket and bold results are the general best. † indicates the previous state-of-the-art.

## 3.7 Conclusion and future work

In this chapter, we have addressed the problem of image retrieval using the kernelized square-loss exemplar machines, and its efficient implementation. The main novelty of the paper is two-fold: First, using the square loss, which avoids retraining for each additional positive training example and calibrating one of its parameters; second, kernelizing the method while keeping a reasonable memory footprint through the use of low-rank approximations. Similar ideas have of course been used in other contexts in machine learning [8, 28, 84, 88, 107]. Our work is, however, to our knowledge, the first to apply these ideas to examplar-based classifiers, in particular in the context of image retrieval. We have obtained significant improvements over the base features we tested and outperformed similar encoders on different datasets. As future

work, we plan to work on a convolutional implementation similar to [1] so its parameters can be learned in a supervised manner. The use of other kernel functions is worth investigating. The polynomial kernel performs similarly to the Gaussian kernel, even though the Hilbert space obtained from the Gaussian kernel has infinite dimensions and the Hilbert space obtained from the polynomial kernel does not. The spatial pyramid kernel can be used to improve representations based on local descriptors and it offers the better boosts when compared with its baseline. Experimentation of kernels that allow different base features are worth pursuing. Finally, our method constructs a generic feature encoding and therefore can be used in many other computer vision problems, such as object classification and scene recognition.

# Chapter 4

# Learning Proposal Flow

This chapter addresses the problem of establishing semantic correspondences between images depicting different instances of the same object or scene category. Previous approaches focus on either combining a spatial regularizer with hand-crafted features, or learning a correspondence model for appearance only. We propose instead a convolutional neural network architecture, called SCNet, for learning a geometrically plausible model for semantic correspondence. SCNet uses region proposals as matching primitives, and explicitly incorporates geometric consistency in its loss function. It is trained on image pairs obtained from the PASCAL VOC 2007 keypoint dataset, and a comparative evaluation on several standard benchmarks demonstrates that the proposed approach substantially outperforms both recent deep learning architectures and previous methods based on hand-crafted features.

This chapter is an extended version of the paper "*SCNet: Learning semantic correspondence*" by Han, Rezende, Ham, Wong, Cho, Schmid and Ponce published in ICCV 2017 [36]. This work was done in collaboration with another PhD student, Kai Han, from the University of Hong Kong. The extent of the contribution of each of us is found at the beginning of each section.

## 4.1 Introduction

The goal of this chapter is to establish semantic correspondences across images that contain different instances of the same object or scene category, and thus feature much larger changes in appearance and spatial layout than the pictures of the same scene used in stereo vision, which we take here to include broadly not only classical (narrow-baseline) stereo fusion (e.g., [67, 80]), but also optical flow computation (e.g., [41, 77, 99]) and wide-baseline matching (e.g., [65, 104]). Due to such a large degree of variations, the problem of semantic correspondence remains very challenging. Most previous approaches to semantic correspondence [14, 43, 52, 59, 90, 104] focus on combining an effective spatial regularizer with hand-crafted features such as SIFT [61], DAISY [92] or HOG [22]. With the remarkable success of deep learning approaches in visual recognition, several learning-based methods have also been proposed for both stereo vision [29, 37, 110, 111] and semantic correspondence [18, 54, 114]. Yet, none of

these methods exploits the geometric consistency constraints that have proven to be a key factor to the success of their hand-crafted counterparts. Geometric regularization, if any, occurs during post-processing but not during learning (e.g., [110, 111]).

We propose a convolutional neural network (CNN) architecture, called SCNet, for learning geometrically plausible semantic correspondence, which allows us to handle a large degree of intra-class and scene variations. Following the proposal flow approach to semantic correspondence of Ham *et al.*[35], we use object proposals [64, 94, 116] as matching primitives, and explicitly incorporate the geometric consistency of these proposals in our loss function. Unlike [35] with its hand-crafted features, however, we train our system in an end-to-end manner using image pairs extracted from the PASCAL VOC 2007 keypoint dataset [25]. A comparative evaluation on several standard benchmarks demonstrates that the proposed approach substantially outperforms both recent deep architectures and previous methods based on hand-crafted features.

Our main contributions can be summarized as follows:

- We introduce a simple and efficient model for learning to match regions using both appearance and geometry.
- We propose a convolutional neural network, SCNet, to learn semantic correspondence with region proposals.
- We achieve state-of-the-art results on several benchmarks, clearly demonstrating the advantage of learning both appearance and geometric terms.

## 4.2 Background

### 4.2.1 Semantic correspondence

Classical approaches to stereo matching and optical flow estimate pixel-level dense correspondences between two nearby images of the same scene.

SIFT Flow [59] extends classical optical flow to establish correspondences across similar but different scenes. It uses dense SIFT descriptors to capture semantic information beyond naive color values, and leverages a hierarchical optimization technique in a coarse-to-fine pipeline for efficiency. Kim *et al.* [52] and Hur *et al.* [43] propose more efficient generalizations of SIFT Flow. Instead of using SIFT features, Yang *et al.* [104] use DAISY [92] for an efficient descriptor extraction. Inspired by an exemplar-LDA approach [38], Bristow *et al.* [14] use whitened SIFT descriptors, making semantic correspondence robust to background clutter.

Recently, Ham *et al.* [35] introduces proposal flow that uses object proposals as matching elements for semantic correspondence robust to scale and clutter. This work shows that the HOG descriptor gives better matching performance than deep learning features [56, 86]. Taniai *et al.* [90] also use HOG descriptors, and show that jointly performing cosegmentation and establishing dense correspondence is helpful

in both tasks. Despite differences in feature descriptors and optimization schemes, these semantic correspondence approaches use a spatial regularizer to ensure flow smoothness on top of hand-crafted or pre-trained features.

### 4.2.2 Deep learning for correspondence

Recently, CNNs have been applied to classical dense correspondence problems such as optical flow and stereo matching to learn feature descriptors [109, 110, 111] or similarity functions [37, 109, 110].

FlowNet [29] uses an end-to-end scheme to learn optical flow with a synthetic dataset. The learned network generalizes well to existing benchmark datasets, but is still outperformed by traditional optical flow methods (e.g., EpicFlow [78]). Several recent approaches also use supervision from reconstructed 3D scenes and stereo pairs [37, 109, 110, 111]. MC-CNN [110] and its efficient extension [111] train CNN models to predict how well two image patches match and use this information to compute the stereo matching cost. DeepCompare [109] learns a similarity function for patches directly from images of a 3D scene, which allows for various types of geometric and photometric transformations (e.g., rotation and illumination changes).

These approaches are inherently limited to matching images of the same physical object/scene. In contrast, Long *et al.* [60] use CNN features pre-trained for ImageNet classification tasks (due to a lack of available datasets for learning semantic correspondence) with performance comparable to SIFT flow. To overcome the difficulty in obtaining ground truth for semantic correspondence, Zhou *et al.* [114] leverage 3D models, and uses flow consistency between 3D models and 2D images as a supervisory signal to train a CNN.

Another approach to generating ground truth is to directly augment the data by densifying sparse keypoint annotations using warping [35, 50]. Although this approach only provides approximate "ground truth" data, we will show that it is good enough to learn semantic correspondence. The universal correspondence network (UCN) of Choy *et al.* [18] learns semantic correspondence using an architecture similar to [111], but adds a convolutional spatial transformer networks for improved robustness to rotation and scale changes. Recently, Kim *et al.* [54] have combined learned semantic descriptors with the proposal flow [35] framework and obtained state-of-the-art performance. These approaches to learning semantic correspondence [18, 114] or semantic descriptors [54] typically perform better than traditional hand-crafted ones, but they do not incorporate geometric consistency between regions or object parts in the learning process. In contrast, our network learns semantic correspondence by using both appearance and geometric informations.

## 4.3 Our approach

We consider the problem of learning to match regions with arbitrary positions and sizes in pairs of images. This setting is general enough to cover all cases of region sampling used in semantic correspondence: sampling a dense set of regular local regions

as in typical dense correspondence [14, 52, 59, 91] as well as employing multi-scale object proposals [3, 42, 64, 94, 116]. In this work, following proposal flow [35], we focus on establishing correspondences between object proposal boxes. Subsection 4.3.3 was mainly developped by Han. The remaining of this chapter is equally implemented by both authors or the pre-existing probabilistic Hough matching model from [17].

### 4.3.1 PHM model

Our basic model for matching starts from the probabilistic Hough matching (PHM) approach of [17, 35]. In a nutshell, given some potential match $m$ between two regions, and the supporting data $\mathcal{D}$ (a set of potential matches), the PHM model can be written as

$$P(m|D) \approx P_a(m) \sum_x P_g(m|x) \sum_{m' \in \mathcal{D}} P_a(m') P_g(m'|x), \tag{4.1}$$

where $P_a(m)$ is the probability that the match $m = [r, s]$ between two regions $r$ and $s$ is correct based on appearance only, and $P_g(m|x)$ is the probability it is correct based on geometry only, computed using the offset $x$ (*e.g.*, position and scale change) between the corresponding regions. This turns out to be an effective spatial matching model that combines appearance similarity with global geometric consistency measured by letting all matches vote on the potential offset $x$ [17, 35].

In our learning framework, we consider similarities rather than probabilities, and rewrite the PHM score for the match $m$ as

$$\begin{aligned} z(m, w) &= f(m, w) \sum_x g(m, x) \sum_{m' \in \mathcal{D}} f(m', w) g(m', x) \\ &= f(m, w) \sum_{m' \in \mathcal{D}} [\sum_x g(m, x) g(m', x)] f(m', w), \end{aligned} \tag{4.2}$$

where $f(m, w)$ is a parameterized appearance similarity function between the two regions in the potential match $m$, $x$ is as before an offset variable (position plus scale), and $g(m, x)$ measures the geometric compatibility between the match $m$ and the offset $x$.

Now assuming that we have a total number of $n$ potential matches, and identifying matches with their indices, we can rewrite this score as

$$\begin{aligned} z(m, w) &= f(m, w) \sum_{m'} K_{mm'} f(m', w), \\ \text{where } K_{mm'} &= \sum_x g(m, x) g(m', x), \end{aligned} \tag{4.3}$$

and the $n \times n$ matrix $K$ is the *kernel matrix* associated with the feature vector $\varphi(m) = [g(m, x_1), \ldots, g(m, x_s)]^T$, where $x_1$ to $x_s$ form the finite set of values that the offset variable $x$ runs over: indeed $K_{mm'} = \varphi(m) \cdot \varphi(m')$. [1]

1. Putting it all together in an $n$-vector of scores, this can also be rewritten as $z(w) = f(w) \odot$

Given training pairs of images with associated true and false matches, we can learn our similarity function by minimizing with respect to $w$

$$E(w) = \sum_{m=1}^{n} l[y_m, z(m, w)] + \lambda\Omega(w), \tag{4.4}$$

where $l$ is a loss function, $y_m$ is the the ground-truth label (either 1 [true] or 0 [false]) for the match $m$, and $\Omega$ is a regularizer (*e.g.*, $\Omega(w) = ||w||^2$). We use the hinge loss and $L_2$ regularizer in this work. Finally, at test time, we associate with any region $r$ the region $s$ maximizing $z([r, s], w^*)$, where $w^*$ is the set of learned parameters.

### 4.3.2 Similarity function and geometry kernel

There are many possible choices for the function $f$ that computes the appearance similarity of the two regions $r$ and $s$ making up match number $m$. Here we assume a trainable embedding function $c$ (as will be shown later, $c$ will be the output of a CNN in our case) that outputs a $L_2$ normalized feature vector. For the appearance similarity function between two regions $r$ and $s$, we then use a rectified cosine similarity:

$$f(m, w) = \max(0, c(r, w) \cdot c(s, w)), \tag{4.5}$$

that sets all negative similarity values to zero, thus making the similarity function sparser as well as insensitive to negative matches distant enough during training, with the added benefit of giving nonnegative weights in Eq. (4.2).

Our geometry kernel $K_{mm'}$ records the fact that two matches (roughly) correspond to the same offset: Concretely, we discretize the set of all possible offsets into bins. Let us denote by $h$ the function mapping a match $m$ onto the corresponding bin $x$, we now define $g$ by

$$g(m, x) = \begin{cases} 1, & \text{if } h(m) = x \\ 0, & \text{otherwise.} \end{cases} \tag{4.6}$$

Thus, the kernel $K_{mm'}$ simply measures whether two matches share the same offset bin or not:

$$K_{mm'} = \begin{cases} 1, & \text{if } h(m) = h(m') \\ 0, & \text{otherwise.} \end{cases} \tag{4.7}$$

In practice, $x$ runs over a grid of predefined offset values, and $h(m)$ assigns match $m$ to the nearest offset point. Our kernel is sparse, which greatly simplifies the computation of the score function in Eq. (4.3): Indeed, let $B_x$ denote the set of matches associated with the bin $x$, the score function $z$ reduces to

$$z(m, w) = f(m, w) \sum_{m' \in B_{h(m)}} f(m', w). \tag{4.8}$$

---

$Kf(w)$, where $z(w) = (z(1, w), \ldots, z(n, w))^T$, "$\odot$" stands for the elementwise product between vectors, and $f(w) = (f(1, w), \ldots, f(n, w))^T$.

This trainable form of the PHM model from [17, 35] can be used within Eq. (4.4).

Note that since our simple geometry kernel is only dependent on matches' offsets, we obtain the same geometry term value of $\sum_{m' \in B_{h(m)}} f(m', w)$ for any match $m$ that falls into the same bin $h(m)$. This allows us to compute this geometry term value only once for each non-empty bin $x$ and then share it for multiple matches in the same bin. This sharing makes computing $z$ several times faster in practice. [2]

### 4.3.3 Gradient-based learning

The feature embedding function $c(m, w)$ in the model above can be implemented by any differentiable architecture, for example a CNN-based one, and the score function $z$ can be learned using stochastic gradient descent. Let us now consider the problem of minimizing the objective function $E(w)$ defined by Eq. (4.4). [3] This requires computing the gradient with respect to $w$ of the function $z$:

$$\begin{aligned} \nabla z(m, w) = [\sum_{m' \in \mathcal{D}} K_{mm'} f(m', w)] \nabla f(m, w) \\ + f(m, w) \sum_{m' \in \mathcal{D}} K_{mm'} \nabla f(m', w). \end{aligned} \tag{4.9}$$

Denoting by $n$ the size of $\mathcal{D}$, this involves $n$ evaluations of both $f$ and $\nabla f$. Computing the full gradient of $E$ thus requires at most $n^2$ evaluations of both $f$ and $\nabla f$, which becomes computationally intractable when $n$ is large enough. The score function of Eq. (4.8) with the sparse kernel of Eq. (4.7), however, greatly reduces the gradient computation:

$$\begin{aligned} \nabla z(m, w) = [\sum_{m' \in B_{h(m)}} f(m', w)] \nabla f(m, w) \\ + f(m, w) \sum_{m' \in B_{h(m)}} \nabla f(m', w). \end{aligned} \tag{4.10}$$

Note that computing the gradient for match $m$ involves only a small set of matches falling into the same offset bin $h(m)$.

2. If the geometry kernel is dependent on something other than offsets, e.g., matches' absolute position or their neighborhood structure, this sharing is not possible.

3. We take $\Omega(w) = 0$ for simplicity in this section, but tackling a nonzero regularizer is easy.

### 4.3.4 Back-propagation for Hough Voting

Given two images associated with $p$ and $q$ proposals respectively. The appearance similarity matrix $F$ is obtained by considering all possible matches between the two:

$$F = \begin{bmatrix} f_{11} & f_{12} & f_{13} & \dots & f_{1q} \\ f_{21} & f_{22} & f_{23} & \dots & f_{2q} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ f_{p1} & f_{p2} & f_{p3} & \dots & f_{pq} \end{bmatrix}. \tag{4.11}$$

The geometric similarity matrix $V$ contains geometric scores assigned to the matches:

$$V = \begin{bmatrix} v_{11} & v_{12} & v_{13} & \dots & v_{1q} \\ v_{21} & v_{22} & v_{23} & \dots & v_{2q} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ v_{p1} & v_{p2} & v_{p3} & \dots & v_{pq} \end{bmatrix}, \tag{4.12}$$

$$v_{ij} = \mathrm{vec}(F)^T g_{ij} \tag{4.13}$$

where vec denotes column-wise vectorization, and the geometry voting vector $g_{ij}$ for a match $m$ is a $(p \times q) \times 1$ vector with each element being 0 or 1, such that $v_{ij}$ is a partial sum of the elements in $F$, which corresponds to $\sum_{m' \in B_{h(m)}} f(m', w)$ in the paper (Eq. (8)).

To learn SCNet, we need to back-propagate for the voting. To achieve this, we need to first calculate $dL/dV$, which is a $p \times q$ matrix, and $L$ stands for loss.

According to the chain rule, we have $dL/dF = dL/dV \cdot dV/dF$, and $dV/dF$ is a $(p \times q) \times (p \times q)$ matrix.

$$\begin{aligned} dL/df_{ij} &= dL/dV \cdot dV/df_{ij} \\ &= [dL/dv_{11}, dL/dv_{21}, \dots, dL/dv_{pq}] \\ &\quad [dv_{11}/df_{ij}, dv_{21}/df_{ij}, \dots, dv_{pq}/df_{ij}]^T \\ &= [dL/dv_{11}, dL/dv_{21}, \dots, dL/dv_{pq}] g_{ij} \end{aligned} \tag{4.14}$$

By vectorizing $dL/dV$, we simplify the gradient estimation for $dL/dF$ (index follows Matlab fashion) as

$$\begin{aligned} dL/dF &= [dL/dv_{11}, dL/dv_{21}, \dots, dL/dv_{pq}] \\ &\quad \begin{bmatrix} dv_{11}/df_{11} & dv_{11}/df_{21} & \dots & dv_{11}/df_{pq} \\ dv_{21}/df_{11} & dv_{21}/df_{21} & \dots & dv_{21}/df_{pq} \\ \vdots & \vdots & \ddots & \vdots \\ dv_{pq}/df_{11} & dv_{pq}/df_{21} & \dots & dv_{pq}/df_{pq} \end{bmatrix} \\ &= [dL/dv_{11}, dL/dv_{21}, \dots, dL/dv_{pq}] \\ &\quad [g_{11}, g_{21}, g_{31}, \dots, g_{pq}] \end{aligned} \tag{4.15}$$

For efficiency, we first compute the offset matrix $H$ for all candidate matches.

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} & \dots & h_{1p} \\ h_{21} & h_{22} & h_{23} & \dots & h_{2p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ h_{p1} & h_{p2} & h_{p3} & \dots & h_{pq} \end{bmatrix}, \tag{4.16}$$

where $h_{ij}$ is the offset bin index corresponding to the match $m = (r_i, r_j)$, that is a match between $r_i$ in image $I_A$ and $r_j$ in image $I_B$. Note that there are only $t$ unique numbers in $H$, which means that there are $t$ different offsets among the $p \times q$ pairs of possible matches.

For example, $H$ can be in the form of

$$H = \begin{bmatrix} x_1 & x_2 & x_3 & \dots & x_4 \\ x_2 & x_1 & x_2 & \dots & x_4 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_1 & x_4 & x_4 & \dots & x_3 \end{bmatrix}$$

In this example, since $h_{11} = h_{n1} = h_{22} = x_1$, then $g$ vector for offset $x_1$ is

$$g_{x_1} = \text{vec}( \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & \dots & 0 \end{bmatrix} )$$

Since matches $m = (r_i, r_j)$ with the same offset index have the same gradient $dL/df_{ij}$, $dL/df_{ij}$ can be computed only once by Eq.(4.14) and shared among them.

## 4.4 SCNet architecture

Among many possible architectures implementing the proposed model, we propose using a convolutional neural network (CNN), dubbed *SCNet*, that efficiently processes regions and learns our matching model. Three variants, SCNet-AG, SCNet-A, SCNet-AG+, are illustrated in Fig. 1-5.

In each case, SCNet takes as input two images $I_A$ and $I_B$, and maps them onto feature maps $F_A$ and $F_B$ by CNN layers. Then, given region proposals $(r_1, \dots, r_p)$ and $(s_1, \dots, s_p)$ for the two images, parallel ROI pooling layers [31, 40] extract feature maps of the same size for each proposal. This is an efficient architecture that shares convolutional layers over all region proposals.

These architecture were implemented jointly by the main authors, but mostly by Han. Han worked on the implementation of the learned fully-connected layer and the extraction of object proposals and I worked on the implementation of the ROI pooling and normalization layers, both unavailable to the library we used for our code.

**SCNet-AG.** The proposal features are fed into a fully-connected layer, mapped onto feature embedding vectors, and normalized into unit feature vectors $c(r_i, w)$ and $c(s_j, w)$, associated with the regions $r_i$ and $s_j$ of $I_A$ and $I_B$, respectively. The value of $f(m, w)$ for the match $m$ associated with regions $r_i$ and $s_j$ is computed as the rectified dot product of $c(r_i)$ and $c(s_j)$ (Eq. (4.5)), which defines the appearance similarity $f(m, w)$ for match $m$. Geometric consistency is enforced with the kernel described in Sec. 4.3.2, using a voting layer, denoted as "$\times K$", that computes score $z(m, w)$ from the appearance similarity and geometric consensus of proposals. Finally, matching is performed by identifying the maximal $z(m, w)$ scores, using both appearance and geometric similarities.

**SCNet-A.** We also evaluate a similar architecture without the geometry term. This architecture drops the voting layer (denoted by $\times K$ in Fig. 1-5) from SCNet-AG, directly using $f(m, w)$ as a score function. This is similar to the universal correspondence network (UCN) [18]. The main differences are the use of object proposals and the use of a different loss function.

**SCNet-AG+.** Unlike SCNet-AG, which learns a single embedding $c$ for both appearance and geometry, SCNet-AG+ learns an additional and separate embedding $c_g$ for geometry that is implemented by an additional stream in the SCNet architecture (dashed lines in Fig. 1-5). This corresponds to a variant of Eq. (4.8), as follows:

$$z^+(m, w) = f(m, w) \sum_{m' \in B_{h(m)}} f_g(m', w), \tag{4.17}$$

where $f_g$ is the rectified cosine similarity computed by $c_g$. Compared to the original score function, this variant allows the geometry term to learn a separate embedding function for geometric scoring. This may be beneficial particularly when a match's contribution to geometric scoring needs to be different from the appearance score. For example, a match of rigid object parts (wheel of cars) may contribute more to geometric scoring than that of deformable object parts (leg of horses). The separate similarity function $f_g$ allows more flexibility in learning the geometric term.

**Implementation details.** We use the VGG16 [87] model that consists of a set of convolutional layers with $3 \times 3$ filters, a ReLU layer and a pooling layer. We find that taking the first 4 convolutional layers is a good trade-off for our semantic feature extraction purpose without loosing a localization accuracy. These layers output features with 512 channels. For example, if the net takes input of $224 \times 224 \times 3$ images, the convolutional layers produce features with the size of $14 \times 14 \times 512$. For the ROI pooling layer, we choose a $7 \times 7$ filter following the fast R-CNN architecture [31], which produces a feature map with size of $7 \times 7 \times 512$ for each proposal. We use randomized prim (RP) algorithm for region proposal [64], since it is gives the best results of [35]. To transform the feature map for each proposal into a feature vector, we use the $FC$ layer with the size of $7 \times 7 \times 512 \times 2048$. The 2048 dimensional

feature vector associated with each proposal are then fed into the $L_2$ normalization layer, followed by the dot product layer, ReLU, our geometric voting layer, and loss layer. The convolutional layers are initialized by the pretrained weights of VGG16 and the fully connected layers have random initialization. We train our SCNet by mini-batch SGD, with learning rate 0.001, and weight decay 0.0005. During training, each mini-batch arises from a pair of images associated with a number of proposals. In our implementation, we generated 500 proposals for each image, which leads to $500 \times 500$ potential matches.

For each mini-batch, we sample matches for training as follows. (1) Positive sampling: For a proposal $r_i$ in $I_A$, we are given its ground truth match $r_i'$ in $I_B$. We pick all the proposals $s_j$ in $I_B$ with $\text{IoU}(s_j, r_i') > T_{pos}$ to be positive matches for $r_i$. (2) Negative sampling: Assume we obtain $k$ positive pairs w.r.t $r_i$. We also need to have $k$ negative pairs w.r.t $r_i$. To achieve this, we first find the proposals $s_t$ in $I_B$ with $IoU(s_t, r_i') < T_{neg}$. Assuming $p$ proposals satisfying the IoU constraint, we find the proposals with top $k$ appearance similarity with $r_i$ among those $p$ proposals. In our experiment, we set $T_{pos} = 0.6$, and $T_{neg} = 0.4$.

## 4.5 Experimental evaluation

In this section we present experimental results with detailed analysis. The experiments and quantitative results presented were divided between the two main authors: Han performed the region matching experiments and I performed the dense matching experiments.

### 4.5.1 Experimental details

**Dataset.** We use the PF-PASCAL dataset that consists of 1300 image pairs selected from PASCAL-Berkeley keypoint annotations [4] of 20 object classes. We divide the dataset into 700 training pairs, 300 validation pairs, and 300 testing pairs. The image pairs for training/validation/testing are distributed proportionally to the number of image pairs of each object class. In training, we augment the data into a total of 1400 pairs by horizontal mirroring. For region proposals, unless stated otherwise, we choose to use the method of Manen *et al.*(RP) [64]. The use of RP proposals is motivated by the superior result reported in [35]. In testing we use 1000 proposals for each image as in [35], while in training we use 500 proposals for efficiency.

**Evaluation metric.** We use three metrics to compare the results of SCNet to other methods. First, we use the probability of correct keypoint (PCK) [106], which measures the precision of dense flow at sparse keypoints of semantic relevance. It is calculated on the Euclidean distance $d(\phi(p), p^*)$ between a warped keypoint $\phi(p)$ and ground-truth one $p^*$ [5]. Second, we use the probability of correct regions (PCR).

4. http://www.di.ens.fr/willow/research/proposalflow/

5. To better take into account the different sizes of images, we normalize the distance by dividing by the diagonal of the warped image, as in [18]

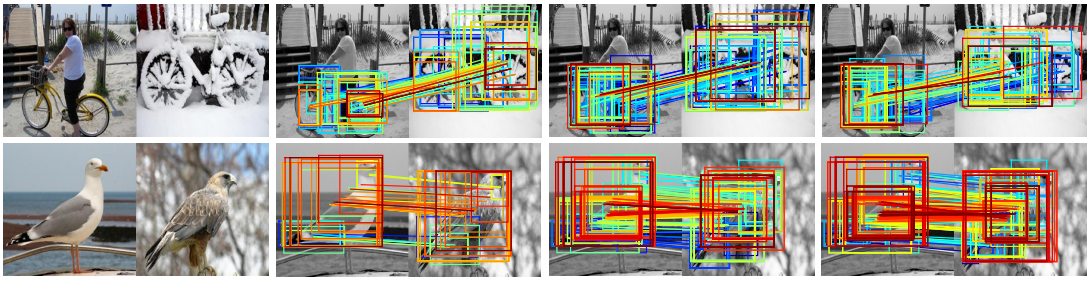

Figure 4-1: Region matching examples. From left to right: Input image pair; NAM matching result; SCNet-A matching result; SCNet-AG+ matching result. Number of correct matches: Bike (NAM[37], SCNet-A[104], SCNet-AG+[107]); Bird (NAM[63], SCNet-A[124], SCNet-AG+[159]).

Introduced in [35] as an equivalent of the the PCK for region based correspondence, PCR measures the precision of a region matching between region $r$ and its correspondent $r^*$ on the intersection over union (IoU) score $1 - \text{IoU}(\phi(r), r^*)$. Both metrics are computed against a threshold $\tau$ in $[0, 1]$ and we measure PCK@$\tau$ and PCR@$\tau$ as the percentage correct below $\tau$. Third, we capture the quality of matching proposals by the mean IoU of the top $k$ matches (mIoU@$k$). Note that these metrics are used to evaluate two different types of correspondence. Indeed, PCK is an evaluation metric for dense flow field, whereas PCR and mIoU@$k$ are used to evaluate region-based correspondences [35].

### 4.5.2 Proposal flow components

We use the PF-PASCAL dataset to evaluate region matching performance. This setting allows our method to be tested against three other methods in [35]: NAM, PHM and LOM. We also compare our SCNet-learned feature against whitened HOG [22], the best performing handcraft of [35]. Additional results on the PF-WILLOW dataset are available in the supplementary material.

**Qualitative comparison.** Region matching results for NAM, SCNet-A, and SCNet-AG+ are shown in Figure 4-1. In this example, at the IoU threshold 0.5, the numbers of correct matches are shown for all methods. We can see that SCNet models perform significantly better than NAM with HOG feature, and SCNet-A is outperformed by SCNet-AG+ that learns a geometric consistency term.

**Quantitative comparison.** Figure 4-2 compares (at left) SCNet methods with the proposal flow methods [35] on the PF-PASCAL dataset. Our SCNet models outperform the other methods that use the HOG feature. Our geometric models (SCNet-AG, SCNet-AG+) substantially outperform the appearance-only model (SCNet-A), and SCNet-AG+ slightly outperform SCNet-AG. This can also be seen from the area under curve (AuC) presented in the legend. This clearly show the effectiveness of deep learned features as well as geometric matching. In this comparison, we fix the

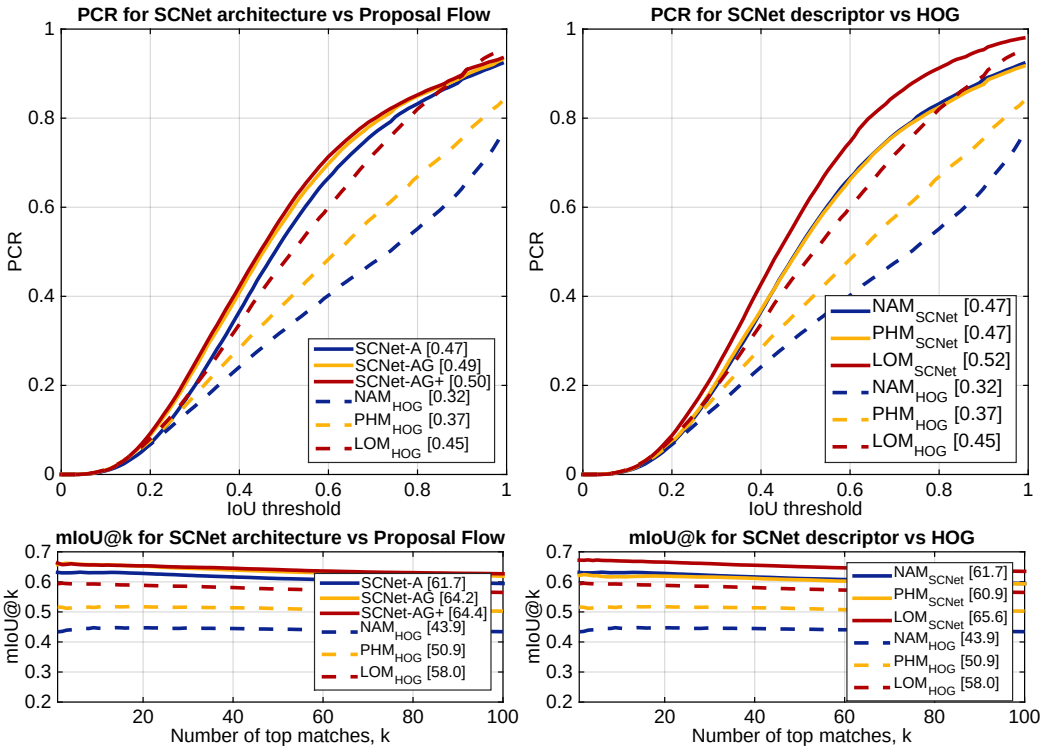


Figure 4-2: PCR (top) and mIoU@$k$ (bottom) plots on PF-PASCAL. AuC is shown in the legend.

VGG16 layer and only learn the FC layers. In our experiment, we also learned all layers including VGG 16 and the FC layers in our model (fully finefuned), but the improvement over the partially learned model was marginal. Figure 4-2 also shows (at right) the performance of NAM, PHM, LOM matching when replacing HOG feature with our learned feature in SCNet-A. We see that SCNet feature greatly improves all the matching methods. Interestingly, LOM using SCNet feature outperforms our best performing SCNet model, SCNet-AG+. However, the LOM method is more than 10 times slower than SCNet-AG+: On average, the method takes 0.21$s$ for SCNet-A feature extraction and 3.15$s$ for the actual matching process whereas our SCNet-AG+ only takes 0.33$s$ in total. Most of the time in LOM is spent in computing its geometric consistency term.

**Results with different object proposals.** SCNet can be combined with any region proposal methods. In this experiment, we train and evaluate SCNet-AG+ on PF-PASCAL with four region proposal methods: randomized prim (RP) [64], selective search (SS) [94], random uniform sampling (US), and sliding window (SW). US and SW are extracted using the work of [42], and SW is similar to regular grid sampling used in other popular methods [52, 59, 77]. Figure 4-3 compares matching performance in PCR and mIoU@$k$ when using the different proposals. RP performs best, and US performs worst with a large margin. This shows that the region proposal process is an important factor for matching performance.

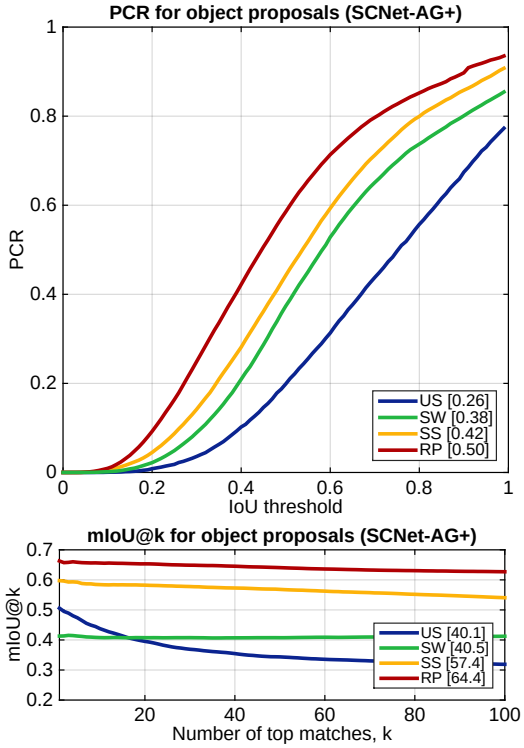


Figure 4-3: SCNet-AG+ on PF-PASCAL with different proposals. PCR (top) and mIoU@$k$ (bottom) plots on PF-PASCAL. AuC is shown in the legend.

**Results on PF-WILLOW** To evaluate transferability performance of the models, we also test them on PF-WILLOW [35], and compare with state-of-the-art results on the datasets. In this case direct comparison to other learning-based methods may not be fair in the sense that our models are trained on a different dataset.

Figure 4-4 compares (at left) SCNet with proposal flow on the PF-WILLOW dataset. SCNet models outperform the other methods that use HOG features. Figure 4-4 shows (at right) the performance of NAM, PHM, LOM matching when replacing HOG features with our learned features in SCNet-A. We see that SCNet features significantly improve all the matching methods.

### 4.5.3 Flow field

Given a sparse region matching result and its corresponding scores, we generate dense semantic flow using a densifying technique in [35]. The results are evaluated on PF-PASCAL dataset. To evaluate transferability performance of the models, we also test them on other datasets such as Caltech-101 [27] and PASCAL Parts [113] datasets, and compare with state-of-the-art results on the datasets. In these cases direct comparison to other learning-based methods may not be fair in the sense that our models are trained on a different dataset.

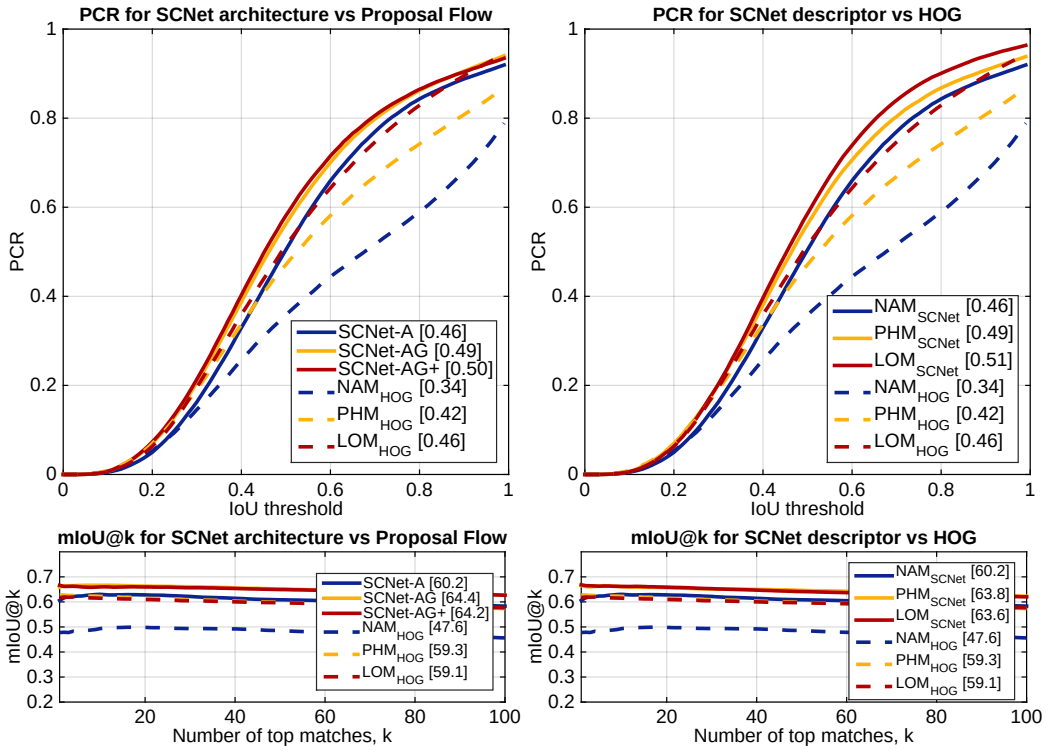


Figure 4-4: PCR (top) and mIoU@$k$ (bottom) plots on PF-WILLOW. AuC is shown in the legend.

| Method | aero | bike | bird | boat | bottle | bus | car | cat | chair | cow | d.table | dog | horse | moto | person | plant | sheep | sofa | train | tv | mean |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $NAM_{HOG}$ [35] | 72.9 | 73.6 | 31.5 | 52.2 | 37.9 | 71.7 | 71.6 | 34.7 | 26.7 | 48.7 | 28.3 | 34.0 | 50.5 | 61.9 | 26.7 | 51.7 | 66.9 | 48.2 | 47.8 | 59.0 | 52.5 |
| $PHM_{HOG}$ [35] | 78.3 | 76.8 | 48.5 | 46.7 | 45.9 | 72.5 | 72.1 | 47.9 | 49.0 | 84.0 | 37.2 | 46.5 | 51.3 | 72.7 | 38.4 | 53.6 | 67.2 | 50.9 | 60.0 | 63.4 | 60.3 |
| $LOM_{HOG}$ [35] | 73.3 | 74.4 | 54.4 | 50.9 | 49.6 | 73.8 | 72.9 | 63.6 | 46.1 | 79.8 | 42.5 | 48.0 | 68.3 | 66.3 | 42.1 | 62.1 | 65.2 | 57.1 | 64.4 | 58.0 | 62.5 |
| UCN-ST* [18] | 30.8 | 26.6 | 34.8 | 22.9 | 51.2 | 29.2 | 31.1 | 34.1 | 20.5 | 27.8 | 18.2 | 34.3 | 31.1 | 20.1 | 38.5 | 42.4 | 41.5 | 35.9 | 16.8 | 32.3 | 36.0 |
| UCN-ST [18] | 64.8 | 58.7 | 42.8 | 59.6 | 47.0 | 42.2 | 61.0 | 45.6 | 49.9 | 52.0 | 48.5 | 49.5 | 53.2 | 72.7 | 53.0 | 41.4 | **83.3** | 49.0 | **73.0** | 66.0 | 55.6 |
| SCNet-A | 67.6 | 72.9 | 69.3 | 59.7 | **74.5** | 72.7 | 73.2 | 59.5 | 51.4 | 78.2 | 39.4 | 50.1 | 67.0 | 62.1 | **69.3** | **68.5** | 78.2 | 63.3 | 57.7 | 59.8 | 66.3 |
| SCNet-AG | 83.9 | 81.4 | **70.6** | 62.5 | 60.6 | 81.3 | 81.2 | 59.5 | 53.1 | 81.2 | **62.0** | 58.7 | 65.5 | 73.3 | 51.2 | 58.3 | 60.0 | 69.3 | 61.5 | **80.0** | 69.7 |
| SCNet-AG+ | **85.5** | **84.4** | 66.3 | **70.8** | 57.4 | **82.7** | **82.3** | **71.6** | **54.3** | **95.8** | 55.2 | **59.5** | **68.6** | **75.0** | 56.3 | 60.4 | 60.0 | **73.7** | 66.5 | 76.7 | **72.2** |

Table 4.1: Per-class PCK on PF-PASCAL at $\tau = 0.1$. For all methods using object proposals, we use 1000 RP proposals [64].

**Evaluation metric for flow field** Since matching images vary in sizes, the metric needs to take into account the size of target images in measuring distances between predicted keypoints and groundtruth keypoints, *i.e.*, the predicted keypoint is deemed correct if it lies at most $\alpha L$ of the groundtruth keypoint. *Ham et al.* [35] set $L = max(h, w)$, where $h \times w$ is the size of the groundtruth object bounding box of the target image, whereas *Choy et al.* [18] set $L = \sqrt{H^2 + W^2}$ for experiments on CUB2011 and $L = max(H, W)$ for all other experiments, with $H \times W$ being the size of the target image. In evaluating PCK, this work follows the protocol of [18].

**Results on PF-PASCAL.** We compare SCNet-A, SCNet-AG, and SCNet-AG+ with proposal flow methods [35] and UCN [18] on PF-PASCAL dataset, and summarize the result in Table 4.1. The UCN is retrained on the PF-PASCAL dataset for fair comparison. The results of UCN* are obtained using the author-provided network

| Method | PCK@0.05 | PCK@0.1 | PCK@0.15 |
|---|---|---|---|
| SIFT [59] | 0.247 | 0.380 | 0.504 |
| DAISY [104] | 0.324 | 0.456 | 0.555 |
| LSS [82] | 0.347 | 0.504 | 0.626 |
| DASC [53] | 0.255 | 0.411 | 0.564 |
| DeepD. [86] | 0.187 | 0.308 | 0.430 |
| DeepC. [109] | 0.212 | 0.364 | 0.518 |
| MatchN [37] | 0.205 | 0.338 | 0.476 |
| LIFT [108] | 0.197 | 0.322 | 0.449 |
| LIFT [6] [108] | 0.224 | 0.346 | 0.489 |
| VGG [87] | 0.224 | 0.388 | 0.555 |
| FCSS [54] | 0.354 | 0.532 | 0.681 |
| UCN-ST-GL | 0.215 | 0.334 | 0.440 |
| UCN-ST-PF | 0.291 | 0.417 | 0.513 |
| SCNet-A | 0.390 | **0.725** | **0.873** |
| SCNet-AG | **0.394** | 0.721 | 0.871 |
| SCNet-AG+ | 0.386 | 0.704 | 0.853 |

Table 4.2: Fixed-threshold PCK on PF-WILLOW. PCK is averaged over all classes.

that is trained on a different subset of PASCAL VOC 2011 dataset. In our benchmark, all SCNet models significantly outperform UCN [18] and all methods in [35], and SCNet-AG+ performs best. Figure 4-5 presents six examples of dense semantic matching for PF-PASCAL. Ground truths are presented as circles and predicted keypoints are presented as squares. We observe a better performance of SCNet-AG and SCNet-AG+ in both examples.

**Results on PF-WILLOW.** The results are summarized in Table 4.2. On this dataset where the data has a different distribution, SCNet-A slightly outperforms the AG and AG+ variants. We also evaluate UCN-ST with two different models: GoogLenet pretrained model [89] (denoted by UCN-ST-GL), which is used as initialization for training, and PF-PASCAL trained model (denoted by UCN-ST-PF). We observe that all SCNet models significantly outperform UCN-ST-PF, while both of them have been trained on the same dataset.

**Results on Caltech-101.** We also evaluate our approach on the Caltech-101 dataset [27]. Following the experimental protocol in [52], we randomly select 15 pairs of images for each object class, and evaluate matching accuracy with three metrics: Label transfer accuracy (LT-ACC) [58], the IoU metric, and the localization error (LOC-ERR) of corresponding pixel positions. Table 4.3 compares quantitatively the matching accuracy of SCNet to the state of the art. It shows that SCNet-AG+ outperforms other approaches, for all metrics. Our results verify that SCNet models successfully learn semantic correspondence.

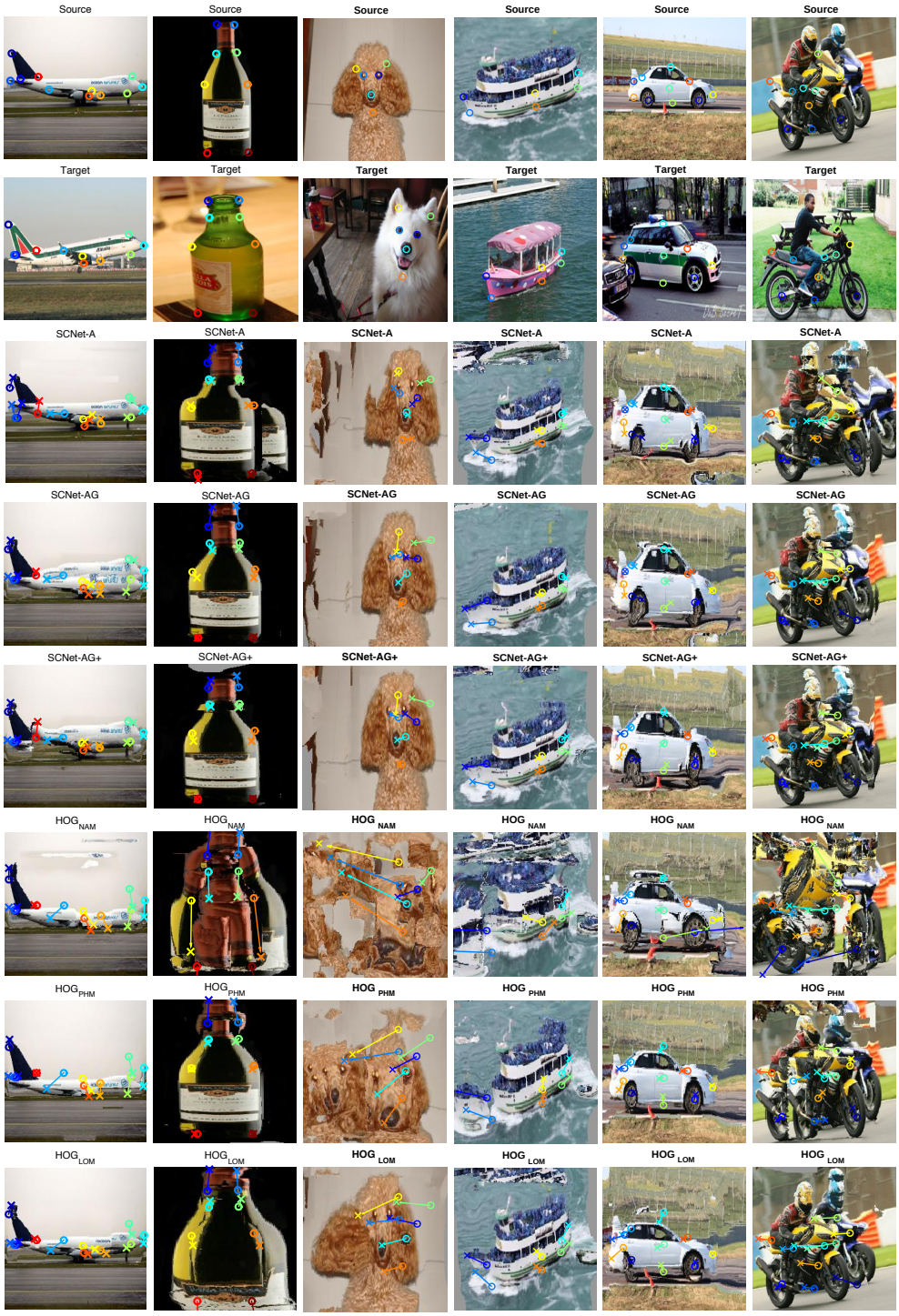


Figure 4-5: Quantitative comparison of dense correspondence for PF-PASCAL (*airplane*, *bottle*, *boat* and *dog*) and PF-WILLOW (*car* and *motorcycle*) image pairs. At first row, both instances (source and target) of the pairs are shown with its respective keypoints. At second and third rows, warped images of the source to the target using different methods are visualized. We show the keypoints of the target image in circles and the predicted keypoints of the source in crosses, with a vector that depicts the matching error.

| Methods | LT-ACC | IoU | LOC-ERR |
|---|---|---|---|
| $\text{NAM}_{\text{HOG}}$ [35] | 0.70 | 0.44 | 0.39 |
| $\text{PHM}_{\text{HOG}}$ [35] | 0.75 | 0.48 | 0.31 |
| $\text{LOM}_{\text{HOG}}$ [35] | 0.78 | 0.50 | 0.26 |
| DeepFlow [77] | 0.74 | 0.40 | 0.34 |
| SIFT Flow [59] | 0.75 | 0.48 | 0.32 |
| DSP [52] | 0.77 | 0.47 | 0.35 |
| SCNet-A | 0.78 | 0.50 | 0.28 |
| SCNet-AG | 0.78 | 0.50 | 0.27 |
| SCNet-AG+ | **0.79** | **0.51** | **0.25** |

Table 4.3: Results on Caltech-101.

## 4.6 Future approaches

In this section, we discuss novel models of incorporating geometrical consistency to our SCNet architecture. The model presented in Section 4.3 is fully derived from the probabilistic Hough matching of [17] and had been successfully implemented in region based matching in [35]. We propose new geometric voting strategies based on localized voting and limiting the number of voting matches by their appearance similarity. For the remaining of this section, the set of all possible matches is written as $\mathcal{D} = \mathcal{R} \times \mathcal{R}'$, where $\mathcal{R}$, $\mathcal{R}'$ are the set of object proposals from the source and target image, respectively.

This section has so far been developed independently from Han and implemented independently from other authors of [36].

### 4.6.1 New geometric voting strategies

The matching score $z$ can be generalized as the product of the appearance matching $f$ score and a voting score $v$:

$$z(m, w) = f(m, w)\ v(m, w|\mathcal{D}) \tag{4.18}$$

For the voting score of Eq. (4.3), every match $m'$ in $\mathcal{D}$ geometric consistent with $m$, *i.e.*, has approximately the same offset, 'votes' for $m$ and the vote is weighted by its appearance score $f(m', w)$. It is unclear however if this is the best strategy for select which pairs vote. Indeed, large sets of object proposals contain many outlier regions, resulting in many mediocre matches. Even if an outlier match has low appearance score, it still influences the matching score of the matches of similar offset. Conversely, relative distance between the matching regions of $m$ and $m'$ do not weight on their voting score. We can eliminate these matching by restricting voting to matches with good appearance, matches of neighboring regions in the source image or by measuring geometric compatibility proportionally to the relative position of the matches. Ham *et al.* [35] solved this by suggestion a local offset matching (LOM),

but this is not suitable for our SCNet architecture since the calculation of the local offset is non-differentiable.

We present below our new voting strategies adapted to our architecture that can minimize the problems of PHM. These voting strategies are then implemented for region matching on PF-WILLOW dataset, using 500 object proposals and HoG [22] as appearance feature. We compare the results of region matching of our models with PHM and LOM voting using both measures for region matching: PCR@$\tau$ and mIoU@$k$, as explained in subsection 4.5.1.

**kNN-PHM** LOM proposes a voting method that truncate voting to the best match of each region in contrast to PHM strategy of unrestricted voting. Truncated voting eliminates background clutter but it also eliminates geometric consistent matches from the voting model. We propose an alternative voting method less restrictive that LOM but that exclude most outliers, call kNN-PHM. For each region $r$, we want the the $k$ best appearances matches to vote, for a certain constant $k$. We define $\mathcal{S}_k(r)$ as the subset $\mathcal{D}$ of the matches from $r$ with the $k$ highest appearance scores. The truncated set of matches $\mathcal{T}_k$ is defined as the union of these subsets for all proposals :

$$\mathcal{T}_k = \bigcup_{r \in \mathcal{R}} \mathcal{S}_k(r). \tag{4.19}$$

The geometrical score is given then by

$$v_{kNN-PHM}(m, w|\mathcal{D}) = \sum_{m' \in \mathcal{T}_k} K_{mm'} f(m', w) \tag{4.20}$$

$$= \sum_{m' \in \mathcal{T}_k \cap B_{h(m)}} f(m', w). \tag{4.21}$$

Notice in particular that if $p$ object proposals are extracted from the source image, $\mathcal{T}_p = \mathcal{D}$ and kNN-PHM for $k = p$ is equivalent to PHM.

We test kNN-PHM for different numbers of voting neighbors (10, 50, 100 and 500) and show the results in Fig. 4-6. Although we see an improvement for a reduced set of voting matches, this improvement negligible. AuC of PCR@$\tau$ improves at most 0.005 and AuC of mIoU@$k$, 0.004. We also see that for $k$ too small, we observe worse region matching results than PHM.

**Local probabilistic Hough matching (LPHM)** We assume geometric consistency is more important for neighbor matching proposals. Objects classes that allow more deformations have little geometric consistency for non-adjacent parts. Hence, we implement the PHM strategy with local voting, *i.e.*, a match $m' = [r', s']$ only votes for a match $m = [r, s]$ if its source object proposal $r'$ has non-zero intersection with $r$. We therefore restrict voting to pairs of proposals that allow less geometrical deformation.

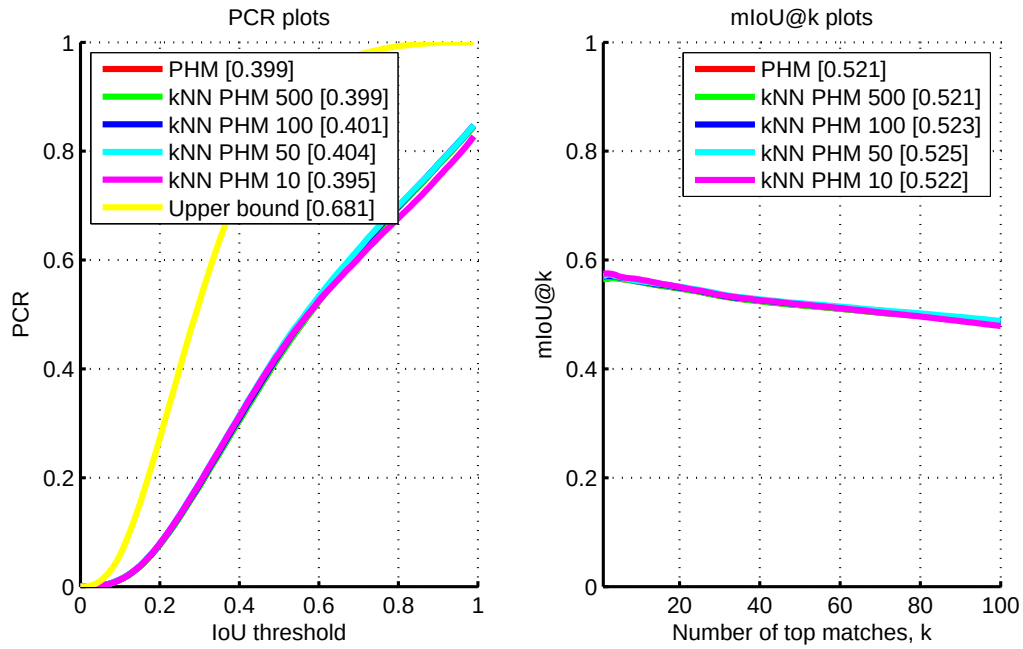


Figure 4-6: Results comparison for PHM and kNN-PHM on PF-WILLOW, when varying the value of $k$. We notice a marginal improvement for big truncated voting sets ($k = 50$, $k = 100$) w.r.t PHM ($k = 500$) and a decreasing performance for very small voting sets ($k = 10$). The (red) curve representing PHM is not visible since kNN-PHM 500 (green) coincides with it. We print both curves and AuC in order to check our implementation of kNN-PHM.

We define $\mathcal{N}(r) = \{\hat{r} \in \mathcal{R} \mid r \cap \hat{r} \neq \emptyset\}$ as the subset of $\mathcal{R}$ with all neighbor proposals of $r$, and the geometrical score as

$$\mathcal{T}_k(r) = \bigcup_{r' \in \mathcal{N}(r)} \mathcal{S}_k(r'), \tag{4.22}$$

$$v_{LPHM}(m, w|\mathcal{D}) = \sum_{m' \in \mathcal{T}_p(r)} K_{mm'} f(m', w). \tag{4.23}$$

We compare LPHM to PHM. Again, we observe no improvement. Restricting voting to local regions improves results slightly for non-rigid object classes, *i.e.*, objects that allow for deformations of its parts, (*e.g.*, duck) but shows worse results for rigid object classes (*e.g.*, wine bottle). Since our database matches pairs of images only if they have the same set of non-occluded keypoints, most pairs of images display the same pose and therefore do not contain deformations. The aspect of local voting is fundamental to LOM voting because of the local offset estimation, which is optimized for local matches and would not be robust to the clutter contained in all possible matches. Selecting voting matches locally does not improve geometric

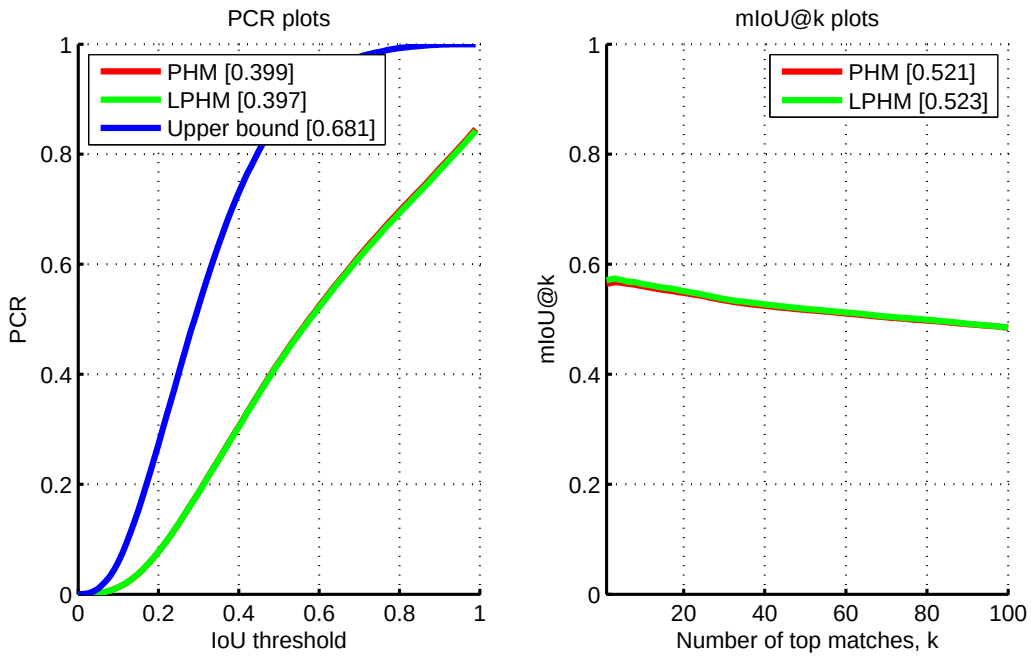


Figure 4-7: Results comparison for LPHM and PHM on PF-WILLOW.

voting in the scenario of PHM, where the geometric consistency of a pair of matches is binary and determined by a clustering of the offset space.

**Local best appearance matching (LBAM)** The issue with LOM applied to a differentiable pipeline is the use of a offset common to all voting matches. Indeed, the common local offset is the geometric median of the selected matches' offsets, which can not be expressed as a function of these matches.[7] We propose an alternative measure of geometric compatibility of two matches as a function of the distance of their offsets. We denote by $G$ the matrix that measures the compatibility of two matches as a Gaussian on the distance of their offsets, *i.e.*,

$$G_{mm'} = e^{-C\|\gamma(m)-\gamma(m')\|}, \tag{4.24}$$

where $C$ is a constant. By replacing the binary compatibility kernel $K$ by $G$, we allow voting matches to be weighted according to the quality of the pairs' geometric compatibility. Similarly to LOM, we also impose that voting matches to be neighbor proposals and only the best appearance match by neighbor proposals, limiting votes for $m$ to the subset $\mathcal{T}_1(r)$ of $\mathcal{R}$. We call our model the local best appearance matching, or LBAM, and define its geometric score as

7. The geometric median is obtained as the argument of minimum of a function of the matches' offsets [16].

$$v_{LBAM}(m, w|\mathcal{D}) = \sum_{m' \in \mathcal{T}_1(r)} G_{mm'} f(m', w). \tag{4.25}$$

Results on PF-WILLOW (Fig. 4-8) show that LBAM and LOM perform similarly (AUC of 0.46) and both outperform the established PHM (AuC of 0.40). We repeat this experiment on PF-PASCAL using 1000 proposals. Figure 4-9 shows again LBAM outperforming PHM (AuC of 0.36 for LBAM and 0.32 for PHM), but both are outperformed by LOM (AuC of 0.42).

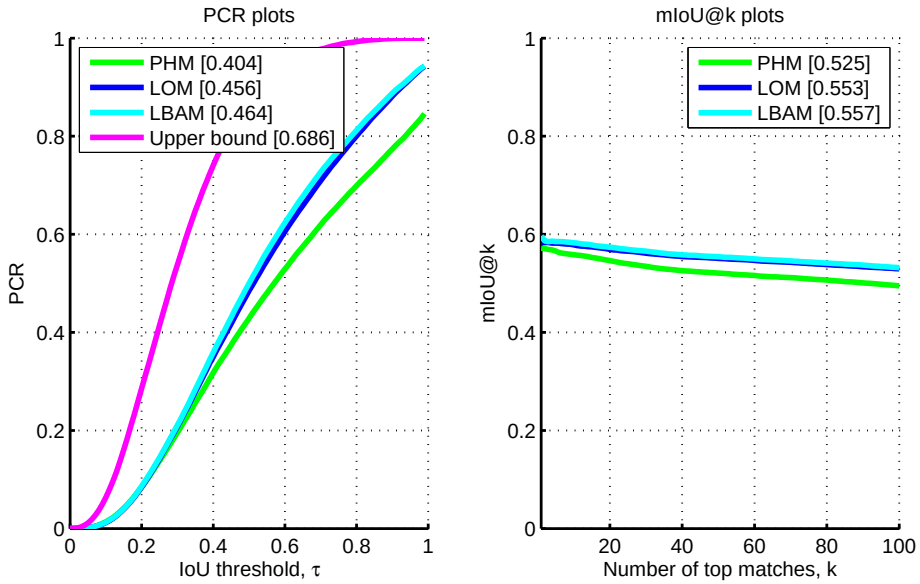


Figure 4-8: Results comparison for PHM, LOM and LBAM on PF-WILLOW. LBAM performs similarly to LOM and outperforms PHM.

Based on these experiments, the remaining of this section is dedicated to the implementation of a SCNet architecture where LBAM is used as our voting layer for geometric scoring (represented by '$\times K$' in Fig. 1-5).

### 4.6.2 Gradient learning and back-propagation for LBAM

Building a neural network architecture able to learn appearance feature using local best appearance matching requires a differentiable voting layer.

For simplicity of notation, we write $z_m = z(m, w)$, $f_m = f(m, w)$ and $v_m = v(m, w|\mathcal{D})$. Equations (4.18) and (4.24) can be rewritten as

$$z_m = f_m v_m, \tag{4.26}$$

$$v_m = \sum_{m' \in \mathcal{D}} f_{m'} G_{mm'} \delta_{mm'}, \tag{4.27}$$

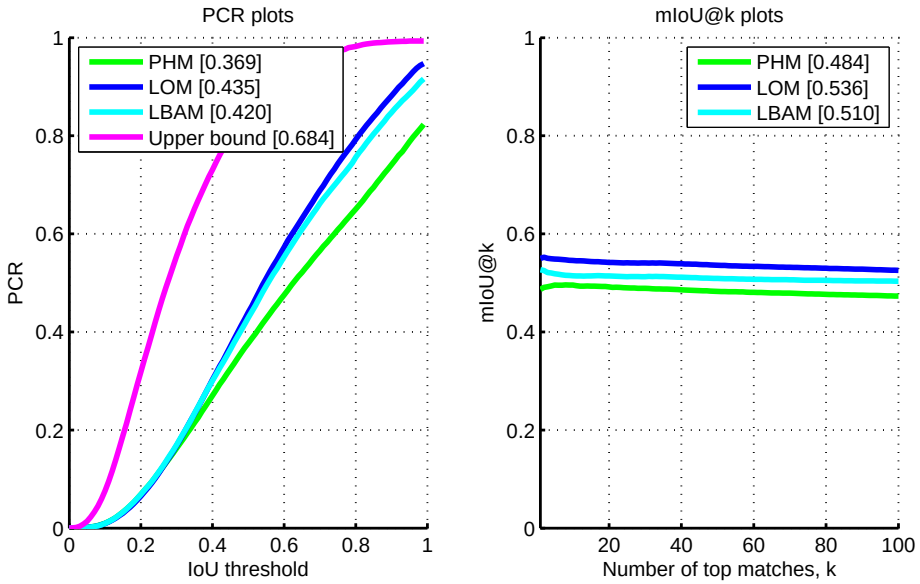


Figure 4-9: Results comparison for PHM, LOM and LBAM for PF-PASCAL. We use all dataset, train, validation and test, similarly to the experiments in PF-WILLOW.

where the binary variable $\delta_{mm'}$ maps if the match $m'$ votes in the geometric score matches of match $m$:

$$\delta_{mm'} = \begin{cases} 1, & \text{if } m' \in \mathcal{T}_1(r) \\ 0, & \text{otherwise.} \end{cases}$$

Similarly to the notation of subsection 4.3.4, let us consider the $p \times q$ matrices $Z = (z_m)_{1 \leq m \leq n}$, $V = (v_m)_{1 \leq m \leq n}$ and $F = (f_m)_{1 \leq m \leq n}$, where $p, q$ are respectively the number of proposals of the source and target image and $n = pq$. We are interested in the back-propagation of the geometric voting, *i.e.*, calculate the matrices $p \times q$ matrices $dL/dV$ and $dL/dF$, where $L$ is the loss written as

$$L = \sum_{m=1}^{n} [y_m - z_m]_+ + \lambda \|w\|^2. \tag{4.28}$$

Therefore, applying the chain rule accordingly:

$$dL/dZ = -\mathbb{1}_{y_m - z_m \geq 0}; \tag{4.29}$$
$$dL/dV = dL/dZ \cdot F; \tag{4.30}$$
$$dL/dF = dL/dV \cdot dV/dF; \tag{4.31}$$
$$dL/df_m = \sum_{m' \in \mathcal{D}} dL/dv_{m'} . dv_{m'}/df_m \tag{4.32}$$

$$= \sum_{m' \in \mathcal{D}} \frac{dL}{dv_{m'}} G_{mm'} \delta_{mm'}. \tag{4.33}$$

For PHM scores, the gradient computation of $z_m$ benefits from the sparsity of matrix $K$, as explained in subsection 4.3.3. A similar sparsity is obtained for LBAM due to the restriction to one match by proposal. This sparsity is represented by $\delta_{mm'}$ and allows us to compute only parts of the $n \times n$ matrix $G$. We thus consider the sparse $n \times n$ matrix $\Delta$, where

$$\Delta = \begin{bmatrix} \delta_{11} & \delta_{12} & \delta_{13} & \dots & h_{1n} \\ \delta_{21} & \delta_{22} & \delta_{23} & \dots & \delta_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \delta_{n1} & \delta_{n2} & \delta_{n3} & \dots & \delta_{nn} \end{bmatrix}, \tag{4.34}$$

and the $m$-th row of $\Delta$ determines the selected matching voting for a given match $m$. Note that two rows are identical if its matches have the same proposal on the source image, which means we only have $p$ unique rows to calculate.

### 4.6.3 Learning with LBAM

This subsection is dedicated to our ongoing work on the SCNet-LBAM architecture.

We have not yet obtained convergence of the LBAM method yet. Figure 4-10 shows objective loss per epoch for different learning rates, and illustrates the objectives convergences we have observed so far: after reduction at the beginning, the training loss stabilizes (and, for the case of learning rate of $5 \cdot 10^{-3}$, increases) at a relative high value (above 0.9).

Figure 4-11 compares the region matching of SCNet-LBAM with SCNet-AG by epoch. We plot the AuC for PCR evaluated over train, test and validation sets separately. We obtain the expected result of SCNet-AG, *i.e.*, for a converging objective function, we obtain improved region matching for the training images set (in blue) and also for a different set of images (red and green). For SCNet-LBAM, we observe that even when its objective is decreasing, region matching is not being improved (for epochs between 20 and 40). In particular, for epochs after 45, we see that smaller objectives do not imply better region matching for test and validation sets (in red and green).

From the training of our three models of Section 4.4, we observed that our architecture is sensitive to changes in the learning rate. Indeed, SCNet-AG and SCNet-AG+ converge under different parameters. We are considering new values of learning rate to solve this problem. Another possible solution is the implementation of a new loss function, as Fig. 4-10 and 4-11 show a drift between the training criteria (minimizing loss function) and testing criteria (maximizing PCR). We do not exclude the possibility of a problem in our back-propagation implementation, despite having not yet encountered the bug that could justify these results. This work will be continued and we intent to submit it as a journal version of our ICCV paper.

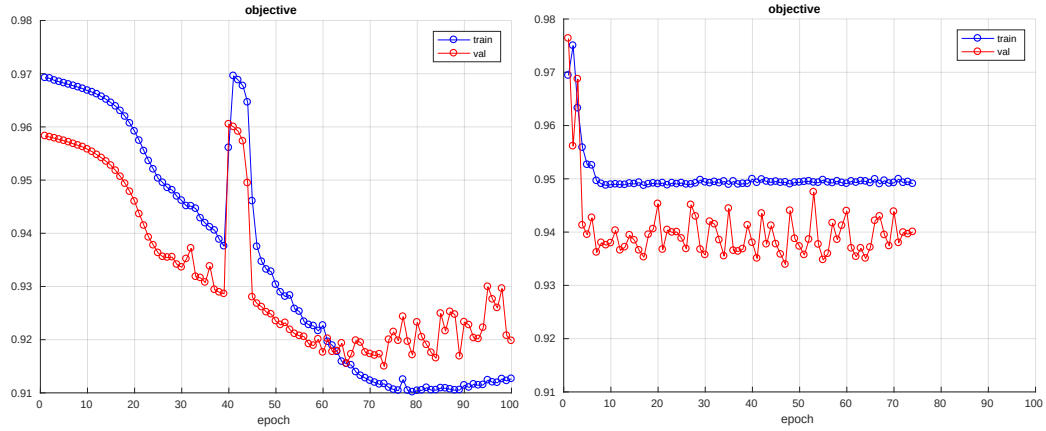


Figure 4-10: Objective by epoch for SCNet-LBAM for different learning rates: $5 \cdot 10^{-3}$ at left, $5 \cdot 10^{-4}$ at right. In both scenarios, we do not observe the expected convergence.

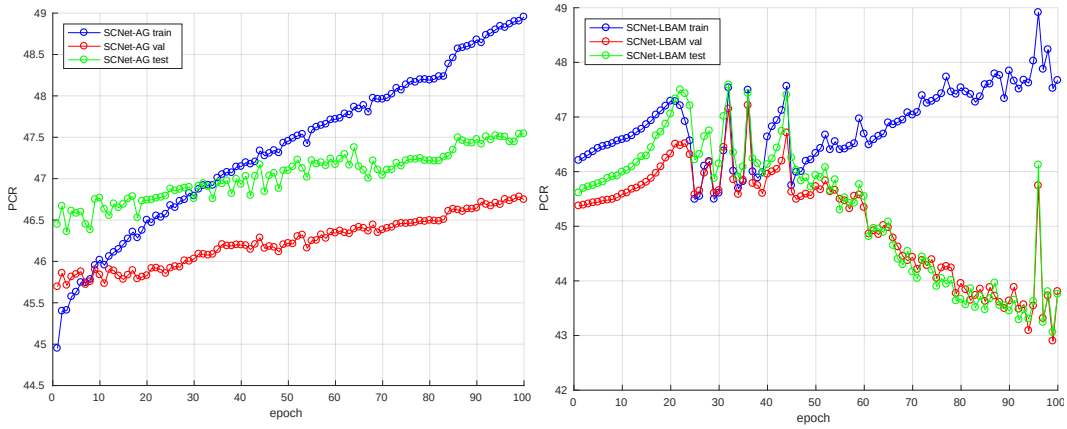


Figure 4-11: Probability of correct region (PCR) plotted by epoch of learning for PHM voting (at left) and LBAM voting (at right, for learning rate of $5 \cdot 10^{-3}$. We plot it for train set (blue), test set (green) and validation set (red).

## 4.7 Conclusion

We have introduced a novel model for learning semantic correspondence, and proposed the corresponding CNN architecture that uses object proposals as matching primitives and learns matching in terms of appearance and geometry. The proposed method substantially outperforms both recent deep learning architectures and previous methods based on hand-crafted features. The result clearly demonstrates the effectiveness of learning geometric matching for semantic correspondence. In the work currently been done, we propose a new model for leverage geometric information between matches and we wish to explore it a improved architecture.

# Chapter 5

# Conclusion

## 5.1 Summary of the thesis

In this work we proposed three new perspectives in three different tasks on image understanding.

### 5.1.1 Fisher vectors for classification

In Chapter 2 we propose experiments and introduce an new formulation of Fisher vectors based upon the assumption we could eliminate background information, which is useful for object-based application such as object classification, and hence obtain a feature representation dependent only on the object category. We start from a statistical formulation present over most of the literature that justifies the elimination of background local descriptors and try to modify our representation in order to better fit this formulation. These modifications consist of:

- Gaussian mixture models that describe only background descriptors instead of full images.
- Projection of Fisher vectors over components that eliminate background information.
- Separation of images in contexts and calculation of one Fisher vector per context for all images, concatenated in a new context-free image representation called contextual Fisher vector, or CFV.

Although the modifications proposed improve the foreground-background ratio for a hypothetical data that follows all assumptions and simple image contexts, these ratio improvements are not observed for more complex datasets that are coherent with the datasets used for classification today. While we are able to create larger foreground-background ratios with a projected Fisher vector, these do not improve performance for object classification.

The work presented in this chapter presents no published material as we judge our negative results are not fitting to most conferences. However, we believe our experiments present an interesting counter-argument to the formulation of Subsection

2.3.2, as we suppose it consists of an *a posteriori* formulation used to justify the normalization of Fisher vectors for improving performance in a classification task. Our experiments suggest that we can not obtain representation free of background information and, furthermore, contextual-specific information can be as useful as foreground information for the construction of an efficient image classification feature. Furthermore, the improved performance obtained by addition of background components to our projected Fisher vector suggests that contextual information is indeed useful for object-based classification.

### 5.1.2 Exemplar classifiers for image retrieval

In Chapter 3, we have addressed the problem of image retrieval using the kernelized square-loss exemplar machines, inspired by exemplar support vector machines, and its efficient implementation. The main novelty of the paper is two-fold: First, using the square loss, which avoids retraining for each additional positive training example and calibrating one of its parameters; second, kernelizing the method while keeping a reasonable memory footprint through the use of low-rank approximations. Similar ideas have of course been used in other contexts in machine learning. Our work is, however, to our knowledge, the first to apply these ideas to examplar-based classifiers, in particular in the context of image retrieval.

We have obtained significant improvements over the base features we tested and outperformed similar encoders on different datasets.

Finally, our method constructs a generic feature encoding and therefore can be used in many other computer vision problems, such as object classification and scene recognition.

### 5.1.3 Learning proposal flow for semantic correspondence

In Chapter 4, we have introduced a novel model for learning semantic correspondence, and proposed the corresponding CNN architecture that uses object proposals as matching primitives and learns matching in terms of appearance and geometry. Our architecture allows for a simultaneous learning of the appearance representation of each object proposal and the geometric voting layer used to measure the compatibility of the relative positions of pairs of matches.

The proposed method substantially outperforms both recent deep learning architectures and previous methods based on hand-crafted features. The result clearly demonstrates the effectiveness of learning geometric matching for both region correspondence and semantic correspondence. The robustness of our method is demonstrated by evaluating the transferability of the learned weights over many datasets.

In the work currently been developed, we propose a new model for leverage geometric information between matches and we wish to explore it a improved architecture.

## 5.2 Contributions and future work

For Chapter 2, our contributions are presented as follows:

- We propose a new framework of application of Fisher vectors to context-independent tasks, by aiming to eliminate the background information.
- We introduce a measurement of the foreground-background ratio of information present in a fisher vector and propose a projection that maximizes it.
- We present results that contradicts the claims of our method and shed light on the statistical formulation used to justify the normalization of Fisher vectors.

For Chapter 3, the contributions are threefold:

- We introduce a kernelized variant of SLEM that enjoys similar computational advantages and improves retrieval performances.
- We propose a low-rank factorization of the kernel matrix for computational and storage efficiency.
- Our experiments show improved results on a variety of base features and we obtain state-of-the-art results for Inria Holidays dataset.

As future work, we plan to work on a convolutional implementation similar to [1] so its parameters can be learned in a supervised manner. The use of other kernel functions is worth investigating. The polynomial kernel performs similarly to the Gaussian kernel, even though the Hilbert space obtained from the Gaussian kernel has infinite dimensions and the Hilbert space obtained from the polynomial kernel does not. The spatial pyramid kernel can be used to improve representations based on local descriptors and it offers the better boosts when compared with its baseline. Experimentation of kernels that allow different base features are worth pursuing.

For Chapter 4, our contributions can be summarized as:

- We introduce a simple and efficient model for learning to match regions using both appearance and geometry.
- We propose a convolutional neural network, SCNet, to learn semantic correspondence with region proposals.
- We achieve state-of-the-art results on several benchmarks, clearly demonstrating the advantage of learning both appearance and geometric terms.

For the future, we want to finish implementation of the SCNet-LBAM architecture, demonstrate the evolution of the performance for region correspondence and apply it to semantic matching in order to compare to our existing method. Eventually, we can improve our voting layer by implementing an hierarchical method of geometric consistency, following the work of Yang *et al.* [103]. Additionally, we wish to improve the efficiency of our architecture calculating the object proposals with a region proposal network (RPN) [76].

# Bibliography


[1] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2016.

[2] R. Arandjelović and A. Zisserman. Three things everyone should know to improve object retrieval. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2012.

[3] P. Arbelaez, J. Pont-Tuset, J. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2014.

[4] Y. Aytar and A. Zisserman. Enhancing exemplar svms using part level transfer regularization. In *British Machine Vision Conference*, 2012.

[5] Y. Aytar and A. Zisserman. Part level transfer regularization for enhancing exemplar svms. In *Journal of Computer Vision and Image Understanding*, 2015.

[6] A. Babenko and V. Lempitisky. Aggregating deep convolutional features for image retrieval. In *Proc. European Conf. Comp. Vision*, 2015.

[7] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitisky. Neural codes for image retrieval. In *Proc. European Conf. Comp. Vision*, 2014.

[8] F. Bach and M. Jordan. Kernel independent component analysis. In *Journal of Machine Learning Research*, 2002.

[9] F. Bach and M. Jordan. Predictive low-rank decomposition for kernel methods. In *Proc. Int. Conf. on Machine Learning*, 2005.

[10] Cagdas Bilen, Joaquin Zepeda, and Patrick Pérez. Learning sparsity inducing analysis operators for discriminative similarity metrics. In *Signal Processing with Adaptive Sparse Structured Representations*, 2015.

[11] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *International Conference on Computational Statistics*, 2010.

[12] S. Boughorbel, J. P. Tarel, and N. Boujemaa. Generalized histogram intersection kernel for image recognition. In *Int. J. of Comp. Vision*, 2010.

[13] Y-L. Boureau, F. Bach, Y. LeCun, and J. Ponce. Learning mid-level features for recognition. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2010.

[14] H. Bristow, J. Valmadre, and S. Lucey. Dense semantic correspondence where every pixel is a classifier. In *Proc. Int. Conf. Comp. Vision*, 2015.

[15] G. C. Cawley and N. L. C. Talbot. Efficient cross-validation of kernel fisher discriminant classifiers. *Pattern Recognition*, 36(11):2585–2592, 2003.

[16] R. Chandrasekaran and A. Tamir. Open questions concerning weiszfeld's algorithm for the fermat-weber location problem. In *Mathematical Programming*, volume 44, pages 293–295, 1989.

[17] M. Cho, S. Kwak, C. Schmid, and J. Ponce. Unsupervised object discovery and localization in the wild: Part-based matching with bottom-up region proposals. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2015.

[18] C.B. Choy, J. Gwak, S. Savarese, and M. Chandraker. Universal correspondence network. In *Proc. Neural Info. Proc. Systems*, 2016.

[19] R. G. Cinbis, J. Verbeek, and C. Schmid. Image categorization using fisher kernerls of non-iid image models. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2012.

[20] S. Clinchant, G. Csurka, F. Perronnin, and J. M. Renders. Xrce's participation to imageval. In *ImageEval Workshop at CVIR*, 2007.

[21] G. Csurka, C. Dance, L. Fan J. Williamowski, and C. Bray. Visual categorization with bag of keypoints. In *ECCV SLCV Workshop*, 2004.

[22] N. Dalal and B. Triggs. Histogram of oriented gradients for human detection. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2005.

[23] Jonathan Delhumeau, Philippe Henri Gosselin, Hervé Jégou, and Patrick Pérez. Revisiting the vlad image representation. *ACM International Conference on Multimedia*, 2013.

[24] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2009.

[25] M Everingham, L Van Gool, CKI Williams, J Winn, and A Zisserman. The pascal visual object classes challenge 2007 (voc2007) results.

[26] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, volume Workshop on Generative-Model Based Vision., 2004.

[27] Li Fei-Fei, Rob Fergus, and Pietro Perona. One-shot learning of object categories. *TPAMI*, 28(4):594–611, 2006.

[28] S. Fine and K Scheinberg. Efficient SVM training using low-rank kernel representations. *JMLR*, 2:243–264, 2001.

[29] Philipp Fischer, Alexey Dosovitskiy, Eddy Ilg, Philip Häusser, Caner Hazırbaş, Vladimir Golkov, Patrick Van der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2015.

[30] M. Gharbi, T. Malisiewicz, S. Paris, and F. Durand. A Gaussian approximation of feature space for fast image similarity. Technical Report CSAIL-TR-2012-032, MIT, 2012.

[31] Ross Girshick. Fast R-CNN. In *Proc. Int. Conf. Comp. Vision*, 2015.

[32] G. H. Golub and C. F. Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.

[33] Albert Gordo, Jon Almazán, Jerome Revaud, and Diane Larlus. Deep image retrieval: Learning global representations for image search. In *Proc. European Conf. Comp. Vision*, 2016.

[34] K. Grauman and T. Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *Proc. Int. Conf. Comp. Vision*, 2005.

[35] B. Ham, M. Cho, C. Schmid, and J. Ponce. Proposal flow. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2016.

[36] K. Han, R. S. Rezende, B. Ham, K. Y. K. Wong, M. Cho, C. Schmid, and J. Ponce. Scnet: Learning semantic correspondence. In *Proc. Int. Conf. Comp. Vision*, 2017.

[37] Xufeng Han, Thomas Leung, Yangqing Jia, Rahul Sukthankar, and Alexander C Berg. MatchNet: Unifying feature and metric learning for patch-based matching. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2015.

[38] B. Hariharan, J. Malik, and D. Ramanan. Discriminative decorrelation for clustering and classification. In *Proc. European Conf. Comp. Vision*, pages 459–472. Springer, 2012.

[39] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning*. Springer, 2009. 2nd edition.

[40] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *European Conference on Computer Vision*, 2014.

[41] B. K. P. Horn and B. G. Schunck. Determining optical flow: A retrospective. *Artificial Intelligence*, 59(1):81–87, 1993.

[42] J. Hosang, R. Benenson, P. Dollár, and B. Schiele. What makes for effective detection proposals? *TPAMI*, 2015.

[43] Junhwa Hur, Hwasup Lim, Changsoo Park, and Sang Chul Ahn. Generalized deformable spatial pyramid: Geometry-preserving dense correspondence estimation. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2015.

[44] T. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In *Proc. Neural Info. Proc. Systems*, 1998.

[45] H. Jégou and O. Chum. Negative evidences and co-occurrences in image retrieval: The benefit of pca and whitening. In *Proc. European Conf. Comp. Vision*, 2012.

[46] H. Jegou, M. Douze, and C. Schmid. Hamming embedding and weak geometry consistency for large scale image search. In *Proc. European Conf. Comp. Vision*, 2008.

[47] H. Jégou and A. Zisserman. Triangulation embedding and democratic aggregation for image search. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2014.

[48] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[49] Yannis Kalantidis, Clayton Mellina, and Simon Osindero. Cross-dimensional weighting for aggregated deep convolutional features. In *Proc. European Conf. Comp. Vision*, 2016.

[50] Angjoo Kanazawa, David W Jacobs, and Manmohan Chandraker. WarpNet: Weakly supervised matching for single-view reconstruction. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2016.

[51] V. Kantorov and I. Laptev. Efficient feature extraction, encoding and classification for action recognition. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2014.

[52] Jaechul Kim, Ce Liu, Fei Sha, and Kristen Grauman. Deformable spatial pyramid matching for fast dense correspondences. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2013.

[53] Sang Wun Kim, Dongbo Min, Bumsub Ham, and Kwanghoon Sohn. Dasc: Dense adaptative self-correlation descriptor for multi-modal and multi-spectral correspondence. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2015.

[54] Seungryong Kim, Dongbo Min, Bumsub Ham, Sangryul Jeon, Stephen Lin, and Kwanghoon Sohn. Fcss: Fully convolutional self-similarity for dense semantic correspondence. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2017.

[55] T. Kobayashi. Three viewpoints toward exemplar-svm. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2015.

[56] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. Neural Info. Proc. Systems*, 2012.

[57] S. Labeznik, C. Schmid, and J. Ponce. Beyond bag of features: Spatial pyramid matching for recognizing natual scenes categories. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2006.

[58] Ce Liu, Jenny Yuen, and Antonio Torralba. Nonparametric scene parsing via label transfer. *TPAMI*, 33(12):2368–2382, 2011.

[59] Ce Liu, Jenny Yuen, and Antonio Torralba. SIFT flow: Dense correspondence across scenes and its applications. *TPAMI*, 33(5):978–994, 2011.

[60] J. L. Long, N. Zhang, and T. Darrell. Do convnets learn correspondence? In *Proc. Neural Info. Proc. Systems*, 2014.

[61] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. of Comp. Vision*, 60(2):91–110, 2004.

[62] T. Malisiewicz, A. Gupta, and A. A. Efros. Ensemble of exemplar-svms for object detection and beyond. In *Proc. Int. Conf. Comp. Vision*, 2011.

[63] T. Malisiewicz, A. Shrivastava, A. Gupta, and A. A. Efros. Exemplar-svms for visual object detection, label transfer and image retrieval. In *Proc. Int. Conf. on Machine Learning*, 2012.

[64] Santiago Manen, Matthieu Guillaumin, and Luc Van Gool. Prime object proposals with randomized Prim's algorithm. In *Proc. Int. Conf. Comp. Vision*, 2013.

[65] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and vision computing*, 22(10):761–767, 2004.

[66] Naila Murray and Florent Perronnin. Generalized max pooling. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2014.

[67] M. Okutomi and T. Kanade. A multiple-baseline stereo. *TPAMI*, 15(4):353–363, 1993.

[68] M. Ozuysal, V. Lepetit, and P.Fua. Pose estimation for category specific multiview object localization. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, Miami, FL, June 2009.

[69] F. Perronnin, C. Dance, G. Csurka, and M. Bressan. Adapted vocabularies for generic visual categorization. In *Proc. European Conf. Comp. Vision*, 2006.

[70] F. Perronnin, J. Sánchez, and T. Mensink. Improving the fisher kernel for large-scale image classification. *European Conference on Computer Vision*, 2010.

[71] Florent Perronnin, Yan Liu, Jorge Sánchez, and Hervé Poirier. Large-scale image retrieval with compressed fisher vector. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2010.

[72] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2007.

[73] James Philbin, Ondřej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2008.

[74] Filip Radenović, Giorgios Tolias, and Ondrej Chum. Cnn image retrieval learns from bow: Unsupervised fine-tuning with hard examples. In *Proc. European Conf. Comp. Vision*, 2016.

[75] Aakanksha Rana, Joaquin Zepeda, and Patrick Perez. Feature learning for the image retrieval task. In *Asian Computer Vision and Pattern Recognition Workshops*, 2014.

[76] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2015.

[77] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid. Deepmatching: Hierarchical deformable dense matching. *ArXiv e-prints*, 2015.

[78] Jerome Revaud, Philippe Weinzaepfel, Zaid Harchaoui, and Cordelia Schmid. EpicFlow: Edge-preserving interpolation of correspondences for optical flow. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2015.

[79] R. S. Rezende, J. Zepeda, J. Ponce, F. Bach, and P. Pérez. Square loss exemplar machines for image retrieval. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2016.

[80] Christoph Rhemann, Asmaa Hosni, Michael Bleyer, Carsten Rother, and Margrit Gelautz. Fast cost-volume filtering for visual correspondence and beyond. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2011.

[81] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek. Image classification with fisher vector: Theory and practice. *Int. J. of Comp. Vision*, 2012.

[82] Eli Schechtman and Michal Irani. Matching local self-similarities accross images and videos. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2007.

[83] B. Schölkopf, R. Herbrich, and A.J. Smola. A genelarized representer theorem. In *Annual Conference on Computational Learning Theory*, pages 416–426, 2001.

[84] B. Schölkopf, A. J. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(3):1299–1319, 1998.

[85] A. Shrivastava, T. Malisiewicz, A. Gupta, and A. A. Efros. Data-driven visual similarity for cross-domain image matching. *ACM Transaction of Graphics (TOG) (Proceedings of ACM SIGGRAPH ASIA)*, 30(6), 2011.

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

[87] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale visual recognition. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2014.

[88] Johan Suykens, T Van Gestel, J De Moor, and J Vandewalle. Least squares support vector machines. In *World Scientific, in press (ISBN 981-238-151-1)*, 2002.

[89] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2015.

[90] Tatsunori Taniai, Sudipta N Sinha, and Yoichi Sato. Joint recovery of dense correspondence and cosegmentation in two images. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2016.

[91] Moria Tau and Tal Hassner. Dense correspondences across scenes and scales. *TPAMI*, 2015.

[92] Engin Tola, Vincent Lepetit, and Pascal Fua. Daisy: An efficient dense descriptor applied to wide-baseline stereo. *IEEE transactions on pattern analysis and machine intelligence*, 32(5):815–830, 2010.

[93] Giorgios Tolias, Ronan Sicre, and Hervé Jégou. Particular object retrieval with integral max-pooling of cnn activations. In *Int. Conf. on Learning Representations*, 2016.

[94] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders. Selective search for object recognition. *Int. J. of Comp. Vision*, 104(2):154–171, 2013.

[95] J. VanGemert, C. Veenman, A. Smeulders, and J. Geusebroek. Visual word ambiguity. In *TPAMI*, 2010.

[96] A. Vedaldi and A. Zisserman. Efficient additive kernels via feature map. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2010.

[97] J. Wang, J. Kang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2010.

[98] G. Washba. *Spline models for observational data.* SIAM, 1990.

[99] Philippe Weinzaepfel, Jerome Revaud, Zaid Harchaoui, and Cordelia Schmid. Deepflow: Large displacement optical flow with deep matching. In *Proc. Int. Conf. Comp. Vision*, 2013.

[100] J. Winn, A. Criminisi, and T. Minka. Object categorization by learned visual dictionary. In *Proc. Int. Conf. Comp. Vision*, 2005.

[101] M. A. Woodbury. *Inverting modified matrices.* Memorandum Rept. 42, Statistical Research Group, Princeton University, Princeton, NJ, 1950, 1950. 4pp.

[102] J. Xiao, J. Hays, K. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2010.

[103] F. Yang, X. Li, H. Cheng, J. Li, and L. Chen. Object-aware dense semantic correspondence. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2017.

[104] Hongsheng Yang, Wen-Yan Lin, and Jiangbo Lu. Daisy filter flow: A generalized discrete approach to dense correspondences. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2014.

[105] J. Yang, K. Yu, Y Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2009.

[106] Yi Yang and Deva Ramanan. Articulated human detection with flexible mixtures of parts. *IEEE TPAMI*, 35(12):2878–2890, 2013.

[107] J. Ye and T. Xiong. Svm versus least squares svm. In *Journal of Machine Learning Research*, 2007.

[108] Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, and Pascal Fua. Lift: Learned invariant feature transform. In *Proc. European Conf. Comp. Vision*, 2016.

[109] S. Zagoruyko and N. Komodakis. Learning to compare image patches via convolutional neural networks. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2015.

[110] Jure Žbontar and Yann LeCun. Computing the stereo matching cost with a convolutional neural network. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2015.

[111] Jure Zbontar and Yann LeCun. Stereo matching by training a convolutional neural network to compare image patches. *Journal of Machine Learning Research*, 17(1-32):2, 2016.

[112] J. Zepeda and P. Pérez. Examplar SVMs as visual feature encoders. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2015.

[113] Tinghui Zhou, Yong Jae Lee, Stella X Yu, and Alyosha A Efros. FlowWeb: Joint image set alignment by weaving consistent, pixel-wise correspondences. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2015.

[114] Tinghui Zhou, Philipp Krähenbühl, Mathieu Aubry, Qixing Huang, and Alexei A Efros. Learning dense correspondence via 3d-guided cycle consistency. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, 2016.

[115] X. Zhou, K. Yu, T. Zhang, and T. Huang. Image classification using super-vector coding. In *Proc. European Conf. Comp. Vision*, 2010.

[116] C Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In *Proc. European Conf. Comp. Vision*, 2014.

## Résumé

Le problème de représentation d'image est au cœur du domaine de vision. Le choix de représentation d'une image change en fonction de la tâche que nous voulons étudier. Un problème de recherche d'image dans des grandes bases de données exige une représentation globale compressée, alors qu'un problème de segmentation séantique nécessite une carte de partitionnement de ses pixels. Les techniques d'apprentissage statisque sont l'outil principal pour la construction de ces représentations. Dans ce manuscrit, nous abordons l'apprentissage des représentations visuels dans trois problèmes différents : la recherche d'image, la correspondance sémantique et classification d'image.

Premièrement, nous étudions la représentation vectorielle de Fisher et sa dépendence sur le modèle de mélange Gaussien employé. Nous introduisons l'utilisation de plusieurs modèles de mélange Gaussien pour différents types d'arrière-plans, i.e. différentes catégories de scène, et analyser la performance de ces représentations pour objet classification et l'impact de la catégorie de scène en tant que variable latente.

Notre seconde approche propose une extension de la représentation l'exemple SVM pipeline. Nous montrons d'abord que, en remplaçant la fonction de perte de la SVM par la perte carrée, on obtient des résultats similaires à une fraction de le coût de calcul. Nous appelons ce modèle la « square-loss exemplar machine », ou SLEM en anglais. Nous introduisons une variante de SLEM à noyaux qui bénéficie des même avantages coputationnelles mais affiche des performances améliorées. Nous présentons des expériences qui établissent la performance et l'efficacité de nos méthodes en utilisant une grande variété de représentations de base et de jeux de données de recherche d'images.

Enfin, nous proposons un réseau neuronal profond pour le problème de l'établissement sémantique correspondance. Nous utilisons des boîtes d'objets en tant qu'éléments de correspondance pour construire une architecture qui apprend simultanément l'apparence et la cohérence géométrique. Nous proposons de nouveaux scores géométriques de cohérence adaptés à l'architecture du réseau de neurones. Notre modèle est entraîné sur des paires d'images obtenues à partir des points-clés d'un jeu de données de référence et évaluées sur plusieurs ensembles de données, surpassant les architectures d'apprentissage en profondeur récentes et méthodes antérieures basées sur des caractéristiques artisanales. Nous terminons la thèse en soulignant nos contributions et en suggérant d'éventuelles directions de recherche futures.



## Mots Clés

Représentation d'image, recherche d'image, modèle à noyaux, réseau de neurone, correspondance de région, correspondance sémantique, vecteurs de Fisher.



## Abstract

The problem of image representation is at the heart of computer vision. The choice of feature extracted of an image changes according to the task we want to study. Large image retrieval databases demand a compressed global vector representing each image, whereas a semantic segmentation problem requires a clustering map of its pixels. The techniques of machine learning are the main tool used for the construction of these representations. In this manuscript, we address the learning of visual features for three distinct problems: Image retrieval, semantic correspondence and image classification.

First, we study the dependency of a Fisher vector representation on the Gaussian mixture model used as its codewords. We introduce the use of multiple Gaussian mixture models for different backgrounds, e.g. different scene categories, and analyze the performance of these representations for object classification and the impact of scene category as a latent variable.

Our second approach proposes an extension to the exemplar SVM feature encoding pipeline. We first show that, by replacing the hinge loss by the square loss in the ESVM cost function, similar results in image retrieval can be obtained at a fraction of the computational cost. We call this model square-loss exemplar machine, or SLEM. Secondly, we introduce a kernelized SLEM variant which benefits from the same computational advantages but displays improved performance. We present experiments that establish the performance and efficiency of our methods using a large array of base feature representations and standard image retrieval datasets.

Finally, we propose a deep neural network for the problem of establishing semantic correspondence. We employ object proposal boxes as elements for matching and construct an architecture that simultaneously learns the appearance representation and geometric consistency. We propose new geometrical consistency scores tailored to the neural network's architecture. Our model is trained on image pairs obtained from keypoints of a benchmark dataset and evaluated on several standard datasets, outperforming both recent deep learning architectures and previous methods based on hand-crafted features.

We conclude the thesis by highlighting our contributions and suggesting possible future research directions.



## Keywords

Image representation, image retrieval, kernel methods, neural networks, region matching, semantic matching, Fisher vectors.