UNIVERSITÉ DE GRENOBLE

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE GRENOBLE

Spécialité : **Mathématiques et Informatique**

Arrêté ministérial : 7 août 2006

Présentée par

Thomas Edgar Josef Mensink

Thèse dirigée par **Cordelia Schmid**
et codirigée par **Jakob Verbeek** et **Gabriela Csurka**

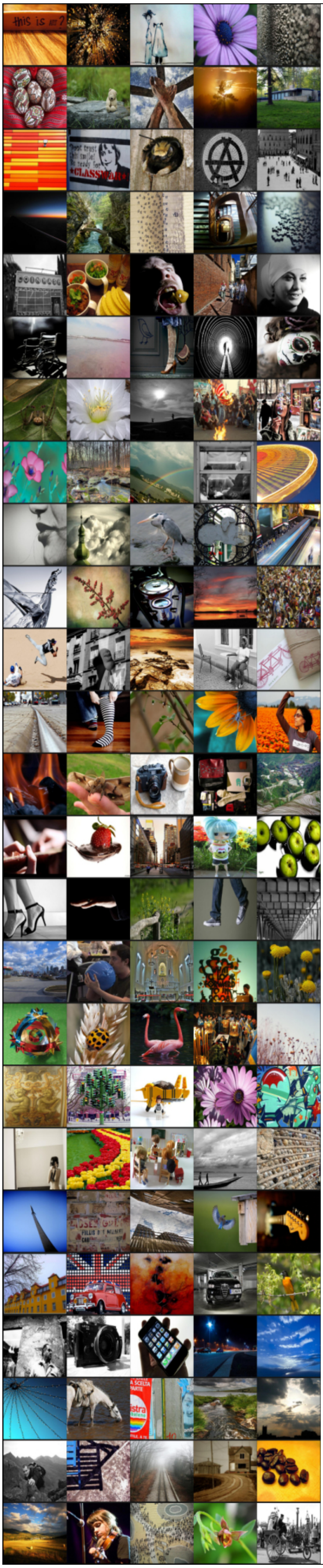
préparée au sein
LEAR – INRIA Grenoble, et
TVPA – Xerox Research Centre Europe
dans **l'École Doctorale Mathématiques, Sciences et Technologies de l'Information, Informatique**

Apprentissage de Modèles pour la Classification et la Recherche d'Images

Learning Image Classification and Retrieval Models

Thèse soutenue publiquement le **26 octobre 2012**,
devant le jury composé de :

Prof. Frédéric Jurie
Université de Caen Bass-Normandie, Président
Prof. Christoph Lampert
Institute of Science and Technology Austria, Rapporteur
Dr. Barbara Caputo
Idiap Research Institute, Rapporteur
Dr. Cordelia Schmid
INRIA Grenoble, Directeur de thèse
Dr. Jakob Verbeek
INRIA Grenoble, Co-Directeur de thèse
Dr. Gabriela Csurka
Xerox Research Centre Europe, Co-Directeur de thèse



Learning Image Classification and Retrieval Models

Thomas Mensink

The research described in this thesis was carried out at the LEAR team of INRIA-Grenoble and the TVPA team of Xerox Research Centre Europe

This work was supported by a CIFRE convention of the ANRT and the French Ministry of Higher Education and Research.

Abstract

We are currently experiencing an exceptional growth of visual data, for example, millions of photos are shared daily on social-networks. Image understanding methods aim to facilitate access to this visual data in a semantically meaningful manner. In this dissertation, we define several detailed goals which are of interest for the image understanding tasks of image classification and retrieval, which we address in three main chapters.

First, we aim to exploit the multi-modal nature of many databases, wherein documents consists of images with a form of textual description. In order to do so we define similarities between the visual content of one document and the textual description of another document. These similarities are computed in two steps, first we find the visually similar neighbors in the multi-modal database, and then use the textual descriptions of these neighbors to define a similarity to the textual description of any document.

Second, we introduce a series of structured image classification models, which explicitly encode pairwise label interactions. These models are more expressive than independent label predictors, and lead to more accurate predictions. Especially in an interactive prediction scenario where a user provides the value of some of the image labels. Such an interactive scenario offers an interesting trade-off between accuracy and manual labeling effort. We explore structured models for multi-label image classification, for attribute-based image classification, and for optimizing for specific ranking measures.

Finally, we explore k-nearest neighbors and nearest-class mean classifiers for large-scale image classification. We propose efficient metric learning methods to improve classification performance, and use these methods to learn on a data set of more than one million training images from one thousand classes. Since both classification methods allow for the incorporation of classes not seen during training at near-zero cost, we study their generalization performances. We show that the nearest-class mean classification method can generalize from one thousand to ten thousand classes at negligible cost, and still perform competitively with the state-of-the-art.

Keywords

Image classification • Image retrieval • Structured prediction • Zero-shot learning • Interactive label prediction • Metric learning • Large-scale classification.

Résumé

Nous assistons actuellement à une explosion de la quantité des données visuelles. Par exemple, plusieurs millions de photos sont partagées quotidiennement sur les réseaux sociaux. Les méthodes d'interprétation d'images vise à faciliter l'accès à ces données visuelles, d'une manière sémantiquement compréhensible. Dans ce manuscrit, nous définissons certains buts détaillés qui sont intéressants pour les taches d'interprétation d'images, telles que la classification ou la recherche d'images, que nous considérons dans les trois chapitres principaux.

Tout d'abord, nous visons l'exploitation de la nature multimodale de nombreuses bases de données, pour lesquelles les documents sont composés d'images et de descriptions textuelles. Dans ce but, nous définissons des similarités entre le contenu visuel d'un document, et la description textuelle d'un autre document. Ces similarités sont calculées en deux étapes, tout d'abord nous trouvons les voisins visuellement similaires dans la base multimodale, puis nous utilisons les descriptions textuelles de ces voisins afin de définir une similarité avec la description textuelle de n'importe quel document.

Ensuite, nous présentons une série de modèles structurés pour la classification d'images, qui encodent explicitement les interactions binaires entre les étiquettes (ou labels). Ces modèles sont plus expressifs que des prédicateurs d'étiquette indépendants, et aboutissent à des prédictions plus fiables, en particulier dans un scenario de prédiction interactive, où les utilisateurs fournissent les valeurs de certaines des étiquettes d'images. Un scenario interactif comme celui-ci offre un compromis intéressant entre la précision, et l'effort d'annotation manuelle requis. Nous explorons les modèles structurés pour la classification multi-étiquette d'images, pour la classification d'image basée sur les attributs, et pour l'optimisation de certaines mesures de rang spécifiques.

Enfin, nous explorons les classifieurs par k plus proches voisins, et les classifieurs par plus proche moyenne, pour la classification d'images à grande échelle. Nous proposons des méthodes d'apprentissage de métrique efficaces pour améliorer les performances de classification, et appliquons ces méthodes à une base de plus d'un million d'images d'apprentissage, et d'un millier de classes. Comme les deux méthodes de classification permettent d'incorporer des classes non vues pendant l'apprentissage à un coût presque nul, nous avons également étudié leur performance pour la généralisation. Nous montrons que la classification par plus proche moyenne généralise à partir d'un millier de classes, sur dix mille classes à un coût négligeable, et les performances obtenus sont comparables à l'état de l'art.

Mot clefs : Classification d'image • recherche d'image • prédiction de structure • apprentissage sans exemple • prédiction interactive d'étiquette • apprentissage de métriques • classification à grande échelle.

Acknowledgements

Over the last few years, I have been lucky to work at two great research labs with fantastic supervisors and colleagues all the while living in Grenoble and its beautiful mountains. I can hardly think of a better place to do research.

First of all, I want to thank my supervisors Jakob and Gabriela, for their continuous support and for creating the PhD position for me. Jakob is the best supervisor I could have hoped for, with his contagious enthusiasm, drive for perfection and constant stream of research ideas. He was always ready to discuss, (re-)explain, help with writing, or with coding and nasty bug-hunting. We also share a great passion for the mountains, which was an endless subject of discussion, and I will happily remember the several ski-rando and hiking trips before going to work.

Gabriela has always given me a lot of encouragement, which has been very valuable for me. She has always given me detailed comments on ideas and results. Moreover I'm grateful for her help dealing with the Xerox side of my project.

I am also grateful to Florent and Cordelia for their collaboration and support as co-authors and managers. Further, I would like to thank Tiberio for his hospitality and the time we worked together on Quadratic Assignment Problems. It was a great experience to be able to visit NICTA in Canberra, Australia. Also thanks to all the great people I met there.

Besides my supervisors, I'm also grateful to many fellow students and colleagues who have come and gone during the last years at LEAR and Xerox. Special thanks to Josip, Julian, and Sara who each read parts of my thesis, and helped in ordering my chaotic thoughts into clear writing, Also thanks to Diane and Adrien for their continuous help with the French language and translating letters, titles and abstracts. The "Rapport de thèse" has been translated entirely by Diane — thanks!

Living in Grenoble has been made extremely pleasant thanks to my flatmates from "Maison du Bonheur" and many friends from HP38. Finally, I'd like to thank my friends and family from the Netherlands whose tireless and unconditional support has been invaluable over the last few years. Thanks to all!

Contents

1

Introduction

"*Data! data! data!*" he cried impatiently.
"*I can't make bricks without clay.*"
— Sherlock Holmes, 1892

The Adventure of the Copper Beeches
Sir Arthur Conan Doyle

Photos and videos are taken, shared, watched and searched constantly, this is a result of readily available high-quality image capturing devices, such as digital cameras and mobile phones, combined with the social acceptance of social-networks and photo-sharing websites. The omnipresence of photos and videos on the internet becomes clear when considering the following:

- The photo-sharing website Flickr hosts over 6 billion photos.
- Users upload 300 million photos per day to the social-network site Facebook.
- The video website Youtube shows over 3 billion hours of videos every month.
- Each week more than 400 hours of programs are converted by the British Broadcasting Corporation (BBC) for their internet service iPlayer.

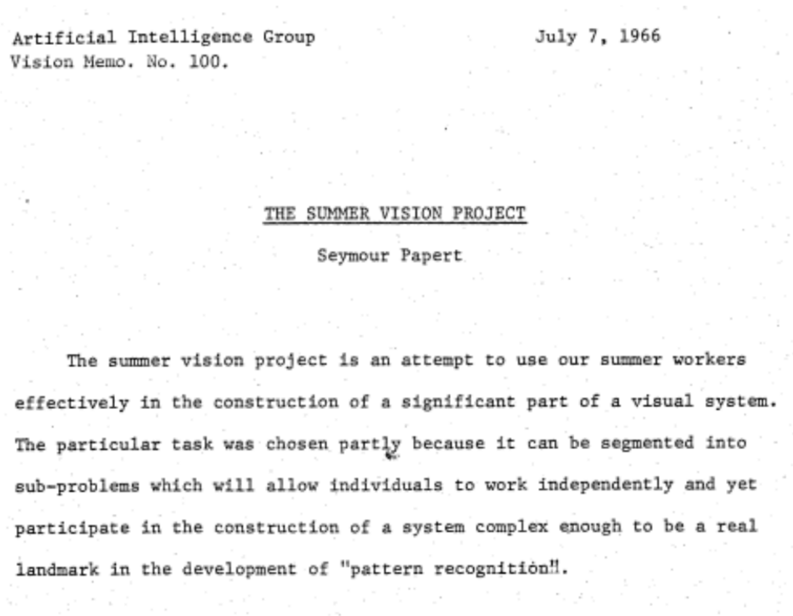
Not only have those internet based services seen an explosion of digital imagery, but also the size of personal photo-collections is increasing. Gartner estimates that due to camera equipped smartphones and tablets, the average storage need per household will grow to over 3 terabytes by 2016.

This visual data is often accompanied by a form of description or meta-data. These descriptions range from automatically generated information, like EXIF-data or GPS-locations, to rich, high-level annotations provided by users, like captions, tags, or face-annotations. Some visual data is even embedded in a hierarchy of knowledge, such as images appearing on Wikipedia, or embedded in a social-network, like Facebook.

Image understanding methods can unlock value in visual data by allowing photos and videos to be findable, searchable, explorable and usable, in a user-oriented and semantically meaningful way based on their visual content. The combination of the described visual data, its additional meta-data, and advances in machine learning, has the potential to lead to new paradigms in image understanding. The methods should generalize over the intrinsic variations in appearances of scenes, objects and people, due to differences in pose, changes in illumination, and object occlusion.

To illustrate the possibilities, we list a few applications using image understanding. First, in robotics, automatic recognition of objects could be used to assign the robot the task of grasping a specific object from a table, while moving around the furniture in the room. Second, in video retrieval, automatic action recognition would allow specific movie scenes to be found based on the actions performed by the actors. Third, in photo book creation, a user could be assisted in composing a photo book from a collection of photos, taking into account the diversity, memorability and aesthetic quality of the photos.

The given examples of applications and the explosion of visual data, illustrate our interest of understanding imagery, on a large-scale for web services, and on a smaller-scale to help users by organizing their personal photo collections. In this dissertation we focus on image understanding, and more specifically the subject matter is supervised learning of statistical models for image classification and retrieval.



Artificial Intelligence Group July 7, 1966
Vision Memo. No. 100.

THE SUMMER VISION PROJECT

Seymour Papert

The summer vision project is an attempt to use our summer workers effectively in the construction of a significant part of a visual system. The particular task was chosen partly because it can be segmented into sub-problems which will allow individuals to work independently and yet participate in the construction of a system complex enough to be a real landmark in the development of "pattern recognition".

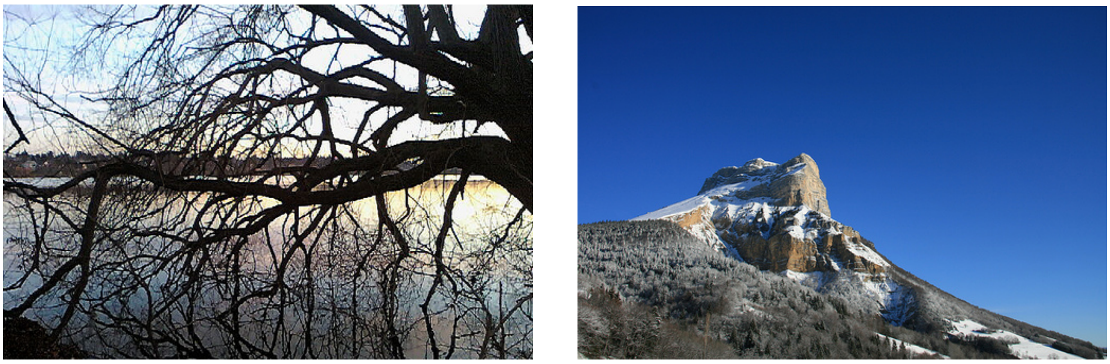
Figure 1.1 *The announcement of the "**The summer vision project**" by Seymour Papert, at MIT in 1966. The goal of the project was to develop, over the course of a summer, a visual recognition system. The system should have been able to segment an image into objects and background, and to identify objects from a dictionary of known objects.*

1.1 Context

In the early days of artificial intelligence, it was believed that solving the computer vision problem would be relatively straightforward. This is illustrated by the "*The summer vision project*", announced by Seymour Papert, at MIT in 1966, see Figure 1.1. The goal of the project was to build a significant part of the visual system by a cohort of students, and more specifically to map the camera input to a description in terms of objects and background. This artificial vision system could be used as an input for *high-level cognitive* tasks such as reasoning and planning. It was thus required to mimic human intelligence or as a component for intelligent robots.

From then on, vision and image understanding have been studied from different perspectives. For example, from the field of cognitive psychology, aiming at understanding human perception, from the field of physics, focusing on modeling physical properties of light emission and surface reflections in images, and from the field of computer science, striving for automatic systems for various vision tasks, like 3D reconstruction and object recognition. The binding factor of these research interests is the use of computers to model and test different theories and methods.

A few decades later we can only conclude that complete understanding of the human-visual system is still elusive, although we know some of its principles. Also that the automatic image understanding of a two years old child remains an unreachable goal



Loved the colour of the sky and water against the dark tree reflections.

A long way to go to the top... Just above my house, close to Grenoble, in front of Belledonne mountains.

Figure 1.2 *Examples of image captions generated by humans for two images from the Stony Brooks University Captioned Photo Dataset, which contains images and captions obtained from Flickr (Ordonez et al., 2011).*

in computer vision research (Szeliski, 2011). Albeit, there are also quite a few success stories, *e.g.* optical character recognition (OCR), face detection in consumer cameras, and multi-camera 3D scene reconstruction.

The difficulty is partly explained by the fact that scene understanding and object recognition are inverse problems, in which we try to recover an understanding of the world given a single or multiple images. It is surprising that humans do this so effortlessly with a very high accuracy, despite the complexity of the task. The visual world in its full complexity is difficult to model, because of the enormous amount of different concepts, the infinite possibilities to project a 3-dimensional scene onto a 2-dimensional image plane, and due to complex scenes with different levels of occlusions and different lighting conditions. Furthermore, there exist a high intrinsic variability in the appearances of objects of the same class. This is even strengthened by visual ambiguity, when two different concepts have a similar appearance, and also by semantical ambiguity, when a single concept has multiple meanings.

Statistical methods allow for the learning of the parameters of a model from large-scale data sets. Once estimated, these models allow for making predictions for previously unseen images. Learning the parameters of these models requires a training set of supervised examples. For image classification these examples could consist of images together with a class label. In addition, an image representation is required, which should be rich enough to encode the relevant visual information of an image. Which visual information is relevant depends by a large amount on the task at hand. We review image representations used for image classification and retrieval in Chapter 2.

1.2 Goals

In this dissertation we address the problem of the understanding of visual content in a semantically meaningful way. The ultimate goal would be to create a system which can generate a human-like description of an image, describing its scene, the objects, and the relations between the objects, see Figure 1.2 for two examples of images with descriptive captions. Ideally, the content of the image should also be placed in context with the non-visible content, *e.g*. the distance relation between the mountain and the city Grenoble, in the right image of Figure 1.2, is not visible in the image. Such a system should be able to relate the low-level image representation to a high-level interpretation of the image by a description of its composition. The lack of coincidence between the low-level features and the high-level interpretation is known as the *semantical gap* (Smeulders et al., 2000).

In this dissertation we specifically focus on supervised learning of statistical models for image classification and retrieval. Both of these tasks aim to relate low-level image features to a semantical similarity with a class label, or with another image, taking advantage of a data set with labeled examples. We briefly define these tasks below.

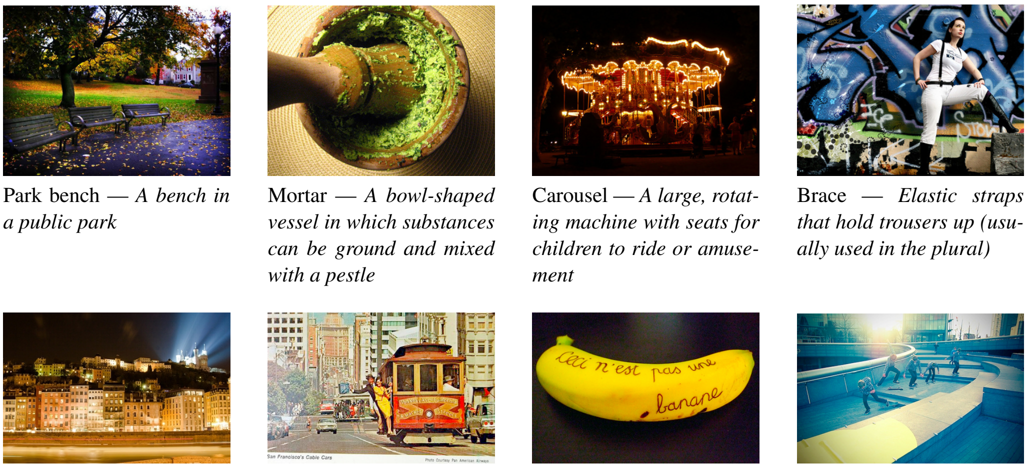
Image classification The goal of image classification is to predict the relevance of one or more semantic concepts from a given vocabulary. We distinguish between multi-class classification, where images are associated with a single label, and multi-label classification, where an image can be related to more than one label. The diversity of concepts used for image classification tasks is illustrated in Figure 1.3. Multi-label image classification is also known as image annotation or image labeling.

Image retrieval The goal of image retrieval is to find the most relevant images in a data set given a query. There are two common paradigms used in image retrieval: in *query-by-text* the query consists of one or more keywords, while for *query-by-example* the query is an example image, see Figure 1.4 for two example queries.

These two image understanding tasks are intimately related. On the one hand, the relevant terms assigned to an image by an image classification system could be used in a query-by-text image retrieval system. On the other hand, image classification could be seen as a query-by-example task, where the query image is labeled with the concepts of its visual nearest neighbors.

In this dissertation we address five specific goals to improve techniques for these two image understanding tasks.

1. **Exploiting multi-modal data** The availability of large scale multi-modal visual databases, which consist of, for example, images along with textual descriptions, or videos along with subtitles, indicates the need for techniques that effectively combine the different modalities present in these documents. The great potential



Park bench — *A bench in a public park*

Mortar — *A bowl-shaped vessel in which substances can be ground and mixed with a pestle*

Carousel — *A large, rotating machine with seats for children to ride or amusement*

Brace — *Elastic straps that hold trousers up (usually used in the plural)*

Building Sights; Citylife; No Visual Season; Outdoor; Sky; Night; Architecture; Street; Church; Visual Arts; Natural; Technical; Cute.

Summer; Outdoor; Plants; Trees; Day; Sunny; Neutral Illumination; Partly Blurred; Small Group; Vehicle; Car; Teenager; Adult; Old person.

No Visual Season; No Visual Place; No Visual Time; Neutral Illumination; No Blur; No Persons; Food; Painting; Artificial; Natural; Cute.

Family Friends; Citylife; Sports; No Visual Season; Outdoor; Day; Sunny; Big Group; Vehicle; Overall Quality; Visual Arts; Cute; Skateboard; Teenager.

Figure 1.3 *Illustrating the diversity of used concepts in image classification, showing example images from the (*top-row*) ImageNet ILSVRC'10 data set (Berg et al., 2010), together with the relevant class name and class description from a set of 1,000 different classes, and (*bottom-row*) ImageClef VCDT'10 data set (Nowak and Huiskes, 2010) together with some of the relevant concepts from a set of 93 labels.*

of exploiting the relation between the different modalities can be understood by viewing the different modalities in a document as forms of mutual, albeit noisy, supervision. Ideally, these techniques allow the exploitation of the multi-modal nature of a training set, even though at test time only a single modality might be available.

2. **Modeling the structure of the output** Many image classification methods do not explicitly model dependencies among the class labels, although often the class labels are (implicitly) embedded in a structure. For example, animal names could be represented by the taxonomy of Linnaeus, or the presence of a certain object may promote the presence of another class that is positively correlated.

   Similarly, the performance measure used to evaluate a system often implies a certain structure. For example, the popular error measure of *average precision* is computed over the list of images ranked according to their relevance for a specific label, where the rank of irrelevant images determines the weight of the incurred penalty. Ideally, both the structure in the output labels, as well as the structure implied by the performance measure should be taken into account during learning.

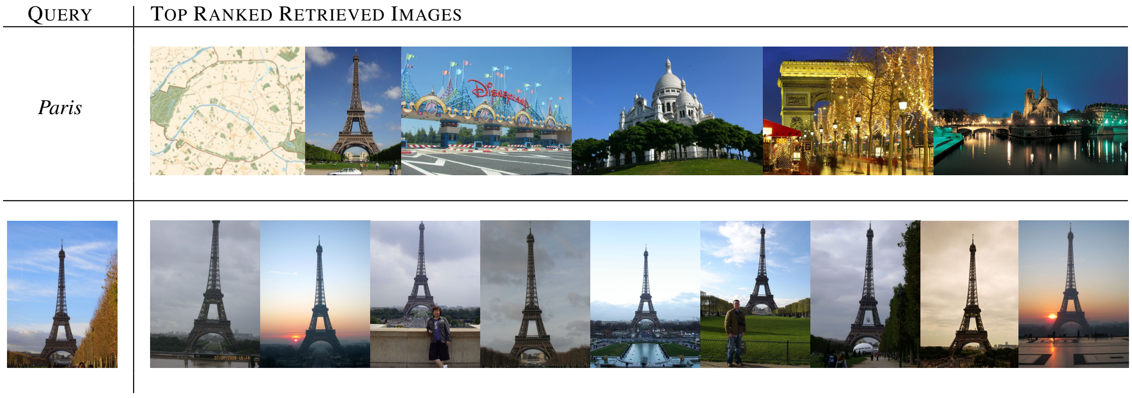


Figure 1.4 *Examples of image retrieval systems,* (top row)*, showing top ranked results from Google's image search using the textual query* "Paris"*, and* (bottom row) *showing top ranked results from the similar image retrieval system BigImbaz*[1]*, using the query image shown on the left.*

3. **Leverage user interaction** User interaction at test time offers an interesting way to enhance the performance of classification algorithms. That users are willing to provide some level of additional information with their images, becomes clear from the popularity of face-naming and geo-tagging of photos on Facebook and in Google's Picasa, to name a few. The amount of human interaction provides an interesting trade-off between accuracy of predictions, and the amount of documents labeled within a specific period of time.

4. **Classifying unseen classes** Real-world data sets are often open ended, where images from new classes are frequently added to the data set. The challenge is to create a classification system which can classify images from classes not seen during training. We explore two popular strategies which enable classification of unseen classes. First, attribute based-classification, which uses predictions of general attributes learned on a training set and an attribute-to-class specification to classify an image into an unseen class. Second, distance based classifiers, such as k-NN, can immediately incorporate new labeled images to determine the closest samples from the data set for a test image.

5. **Scaling to large data sets** The current availability of large scale data sets, such as the ImageNet data set which currently contains 14M images from 21K classes, indicates the need for efficient methods to learn classifiers. Ideally, we would like our learning algorithm to be efficient in several aspects: the computational cost of training and testing, the storage need for the image representations, the number of training images required for generalization, and the human involvement, *e.g.* for the ground-truth labeling of training images.

[1] Demo available at `http://bigimbaz.inrialpes.fr/`

1.3 Contributions

The goals of this dissertation are structured into three main chapters. Here, we briefly describe the main contributions of each chapters presented in this manuscript.

Chapter 3 In this chapter, we consider image retrieval and multi-label image classification using multi-modal data sets, containing images with keywords and captions. In retrieval systems, a successful approach to exploit the multi-modality is by using transmedia relevance feedback models, which given a query image, uses the textual descriptions of the visual nearest neighbors of the query image, as extended textual query. We introduce a parametrization to learn the settings of these transmedia relevance models from training data, for image retrieval. Also, we extend TagProp (Guillaumin et al., 2009a) to exploit these transmedia relevance distances for multi-label image classification. The work in this chapter was partly published in (Mensink et al., 2010b).

Chapter 4 In this chapter, we consider the tasks of image annotation and attribute-based image classification. For these tasks we introduce a series of structured models to exploit the correlation between class labels, and to leverage user interaction.

We use tree-structured graphical models to model the dependencies among the image labels and attributes. The main advantage of these models is that they benefit more from user interaction, compared to independent classifiers.

We also consider the learning of models more specific to ranking measures used for evaluation. Unfortunately, the optimization of several ranking-based performance measures, is intimately related to the NP-hard quadratic assignment problem. We identify a polynomially-solvable subclass, based on c-star structured models that still enables the modeling of a substantial number of pairwise label interactions.

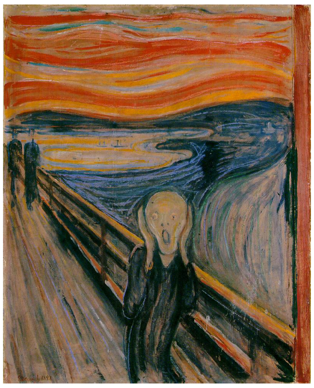
This work was published in (Mensink et al., 2011a, 2012b), and in (Mensink et al., 2011b).

Chapter 5 In this chapter, we consider k-nearest neighbors and nearest class mean classifiers for large-scale image classification. While these methods allow the generalization to unseen classes, they tend to have an unsatisfactory performance when using the Euclidean distance. Therefore, we propose several methods to learn distance metrics, which are shared across all classes, and perform well on unseen classes. The proposed methods are efficient and allow for learning on the ILSVRC'10 data set, which contains over 1.2M images of 1,000 classes, while using image representations of 64K dimensions. This work was published in (Mensink et al., 2012a), and extended in (Mensink et al., Submitted2012).

Before presenting our contributions in Chapter 3, 4 and 5, we first review image representations and machine learning techniques used for image classification and retrieval in Chapter 2. Finally we present our conclusions and discussions in Chapter 6.

2

Image Representations and Image Classification Methods



“*There is something about Munch’s ‘The Scream’ or Constable’s ‘Wivenoe Park’ that no words can convey. It has to be seen. The same holds for of a picture of the Kalahari Desert, a dividing cell, or the facial expression of an actor playing King Lear. It is beyond words. Try to imagine an editor taking in pictures without seeing them or a radiologist deciding on a verbal description. Pictures have to be seen and searched as pictures: by objects, by style, by purpose.*”, Smeulders, Worring, Santini, Gupta, and Jain (2000).

Contents

In this chapter we discuss image representations and machine learning techniques for image classification and retrieval. While this chapter does not consist of core contributions of this thesis, it provides the necessary background in image representations and machine learning concepts used in the subsequent chapters.

2.1 Image representations for classification and retrieval

In this section we discuss image representations used for image classification and image retrieval, with a special emphasis on the Bag-of-Visual-Words (BoV) representation (Sivic and Zisserman, 2003; Csurka et al., 2004) and the Fisher Vector (FV) representation (Perronnin and Dance, 2007; Perronnin et al., 2010b). While image representations are task specific, we focus on these representations as they are efficient and have shown to be among the state-of-the-art representations used for image classification and retrieval since their introduction. They also have proved to be rather generic and were applied in various other image understanding tasks, using a large variety of image classes and types. Even though we limit ourselves by discussing only the BoV and FV approaches the overview given below is far from complete. Due to the success of these approaches and the sheer amount of papers describing new features, modifications and improvements, it is almost impossible to present an exhaustive overview. We therefore focus on a general overview and highlight the most relevant related work for the rest of the manuscript.

2.1.1 Bag-of-Visual-Words

The Bag-of-Visual-Words (BoV, also known as Bag-of-Features and Bag-of-Keypoints) is one of the most successful and popular approaches to describe images for image retrieval and classification. It has been introduced by Sivic and Zisserman (2003) for image

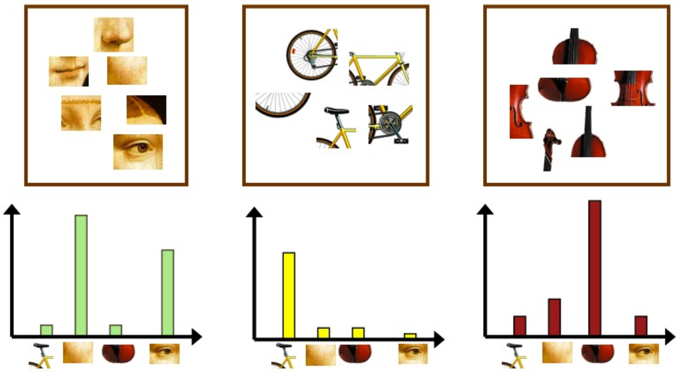


Figure 2.1 *An illustration of the "Bag-of-Visual-Words" representation: an image is considered as an unordered set of patches or regions (*top row*), each patch is assigned to one of the 4 visual-words, and the histogram of "word counts" (*bottom row*) is used to represent the image. Three different images are shown, depicting a face (*left*), a bike (*middle*), and a violin (*right*). Image courtesy of Li Fei-Fei.*

retrieval, and by Csurka et al. (2004) for image classification. Systems using this framework for image classification have consistently performed among the best in the successive PASCAL VOC evaluations (Everingham et al., 2007, 2008, 2009, 2010, 2011).

The BoV approach is inspired by research for text retrieval and classification. A textual document can be efficiently described using the Vector-Space-Model (Salton et al., 1975; Salton, 1989), *i.e.* a vector containing the counts how often a word is present in a document. Because this vector representation neglects the structure of a document, it is known as a "Bag-of-Words" representation, emphasizing the fact that it is invariant of the word order. This Bag-of-Words representation has been successful in text retrieval and classification applications, see *e.g.* (Lewis, 1998; Joachims, 1998).

While text has discrete entities, words, separated by spaces and punctuation which can be counted easily, vision does not have a direct equivalent. A digital image is in the continuous domain, independent whether it is represented by gray-values of its pixels, or by more elaborate features, like SIFT (Lowe, 2004). Therefore, to count "visual-words", first a dictionary (codebook) has to be constructed. Such a codebook does not exist as in natural language[1]. Then, each sample from an unordered set of samples taken from an image, is assigned to its closest match in the codebook, increasing the word-count of the corresponding visual-word. This yields a fixed sized representation of the image, consisting of a histogram of the counts of the occurrence of each visual word in the image. These histograms can be used for image comparisons, retrieval and classification. See Figure 2.1 for a schematic illustration.

[1] Although, also in the natural language domain dictionaries have to be constructed for document retrieval and classification. To reduce dictionary sizes techniques like stemming and stop-word removal are applied. Recent research indicates that fewer processing, yielding higher dimensional (sparse) features, enables better retrieval performance, see *e.g.* (Bai et al., 2010)

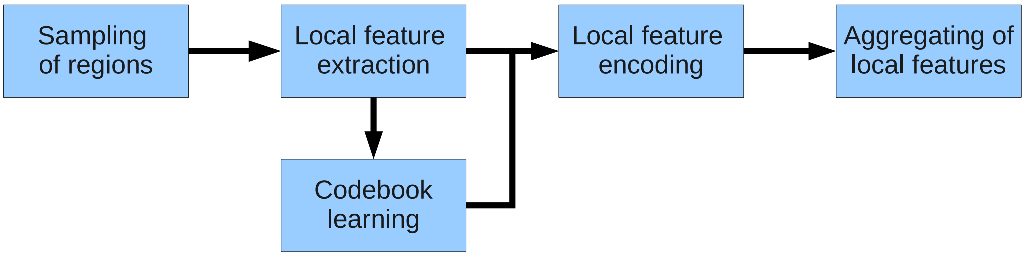


Figure 2.2 *The standard pipeline used in many BoV approaches, see text for details.*

The standard pipeline for most BoV approaches, illustrated in Figure 2.2, consists of the following steps: detection of interest points or regions, local feature extraction, codebook training, feature encoding, and feature aggregation. While many image features are introduced as a combination of a detector (*where to sample*), and a local descriptor (*what to extract*), *e.g*. the SIFT-feature (Lowe, 2004) and the SURF-feature (Bay et al., 2008), we explicitly separate those two steps. Below, we describe the pipeline in more detail.

Sampling of interest points Local features are computed around a set of selected points in the image, therefore the points which are selected influence the final image representation. We describe two different types of approaches to obtain such a set of points.

The first approach is based on key-points detection, where intrinsic image features are used to select a set of points which are of specific interest. These methods are developed for stereo-matching, where the goal is to find matches between similar key-points on the same object in two different, partly overlapping images. Interest points can be obtained for example, by using corners and edges (Harris and Stephens, 1988) or blobs (Lindeberg, 1998). The Harris-Laplace detector (Mikolajczyk and Schmid, 2004), combines the Harris corner detector with automatic scale and shape adaption to obtain a scale and affine invariant key-point detector. The main advantage of these methods is that the same image-region can be found in two images despite variations in viewpoint or scale.

The second approach is based on dense sampling of points from the image (Winn et al., 2005). All points on a regular dense grid over different scales are used as a key-points. The main reasons to use dense sampling is to avoid early removal of potentially interesting points. Intuitively, if the image is described by points from a dense grid over all possible locations, the whole image can be reconstructed from the set of selected points, and therefore less information is lost. Dense sampling has become the de-facto standard in image classification (Van de Sande et al., 2010; Perronnin et al., 2010b); experimental results suggest that the performance increases with the number of regions sampled from images (Nowak et al., 2006; Chatfield et al., 2011).

Local feature descriptors On each of the sampled points, a local feature descriptor is extracted, a description of this pixel and its neighboring pixels. Local features can be as

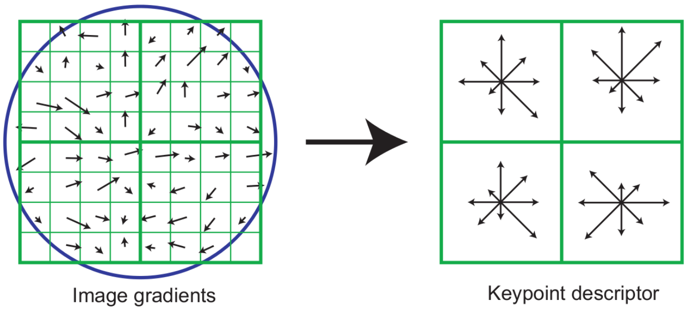


Figure 2.3 *Illustration of the SIFT descriptor, the gradient orientation and magnitude are computed in in a region around the sampled point (*left*), these are accumulated over 4x4 subregions into orientation histograms (*right*). This figure shows a 2x2 descriptor with 8 orientation bins per cell, usually a 4x4 descriptor is used, resulting in a 128 dimensional vector. Image courtesy of Lowe (2004).*

simple as the intensity or RGB values, but usually some more elaborate descriptor is used, which has some level of invariance against illumination change or geometric distortions.

One of the most popular local features is the SIFT descriptor introduced by Lowe (2004), illustrated in Figure 2.3. The SIFT descriptor describes a sampled point, by a histogram of image gradients. The gradients are computed over the intensity levels of the image, and aggregated in several spatial bins around the sampled point, using both the magnitude and the orientation. As a result of using gradients, the descriptor is invariant for additive and multiplicative intensity changes. Also, due to the use of spatial binning and local averages of gradients, the descriptor is robust to some level of geometric distortion and noise. Experimentally the SIFT descriptor has been shown to outperform other descriptors in several tasks (Mikolajczyk and Schmid, 2005).

Despite the popularity of the SIFT descriptor, it is not the only local feature which is used. Some other popular choices include:

- Histogram of Oriented Gradients (HOG, Dalal and Triggs, 2005), which are closely related to the SIFT descriptor, but use a different normalization.
- Color SIFT, which computes SIFT descriptors on gradients of color channels, resulting in *e.g*. Opponent SIFT and rgSIFT (Van de Sande et al., 2010).
- Robust local color histograms which describe the local color by a set of color descriptors which is robust for different photometric changes, such as the Robust Hue descriptor (Van de Weijer and Schmid, 2006).
- Speeded Up Robust Features (SURF, Bay et al., 2008), which computes gradients and magnitudes in only two orientations (x and y); by using integral images and Haar wavelet responses it is faster than SIFT.

- Local Self Similarity (LSS, Shechtman and Irani, 2007), which describes a sampled point by the sum of squared differences between a narrow patch centered at this point and patches from a wider region.
- Local RGB color statistics (LCS, Perronnin et al., 2010b), which computes the mean value and standard deviation of each RGB channel around the sampled point using a 4x4 spatial grid, like the SIFT descriptor.

Many classification systems use a combination of several BoV-histograms, computed over different local features.

Creating a visual dictionary The visual dictionary consists of a series of prototype vectors, which we call visual words. Images will be described by these visual words, therefore the dictionary needs to be rich enough to distinguish different images, and to allow for discriminative image representations. The visual dictionary is often created from a large set of local image descriptors, using an unsupervised clustering approach.

The codebook is frequently obtained by employing k-means clustering (Sivic and Zisserman, 2003; Csurka et al., 2004; Winn et al., 2005), or variants such as hierarchical k-means to reduce the computational complexity (Nistér and Stewénius, 2006), or a Mixture-of-Gaussians (MoG) to obtain a probabilistic interpretation of the clustering (Farquhar et al., 2005; Perronnin and Dance, 2007). Experimentally it has been shown that, classification performance increases with the size of the dictionary (Van Gemert et al., 2010; Chatfield et al., 2011).

Examples of other approaches to learn a codebook are: mean shift clustering to create a codebook which better represents a non-uniform distribution of descriptors (Jurie and Triggs, 2005), or sparse dictionary learning, which learns a codebook to minimize the sparse reconstruction error of local descriptors (Mairal et al., 2008b; Wang et al., 2010).

Supervised methods have also been used for creating a visual dictionary. For example, class labels have been used to learn class-specific codebooks (Farquhar et al., 2005; Perronnin et al., 2006; Khan et al., 2009), or to learn a single codebook to be more discriminative between classes (Winn et al., 2005; Elfiky et al., 2012). Also, methods to learn a codebook together with a classifier have been proposed (Mairal et al., 2008a; Lian et al., 2010; Krapac et al., 2011a).

Encoding local features Each of the local features will now be encoded using the learned codebook. The goal is to represent the original local feature by one or more visual words such that a reconstruction error or expected distortion function is minimized. Since the codebook is in general over-complete (*i.e.* number of codebook entries is larger than number of dimensions of the local descriptors), a constrained encoding is applied to make the problem well defined.

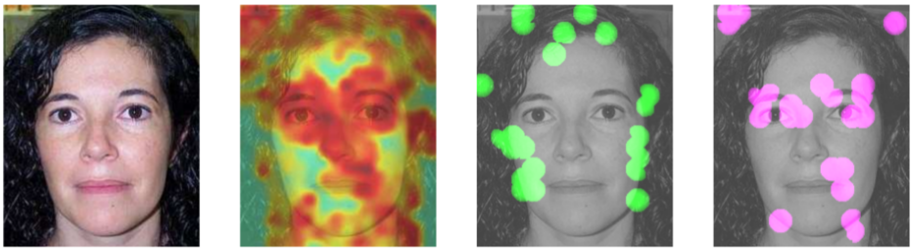


Figure 2.4 *Illustration of the VQ encoding error, (*left*) an image from the Face class in Caltech101, (*middle-left*) the reconstruction error of densely computed SIFT descriptors using a 6,000 word codebook, (red = high error; blue = low error). Note that the most informative patches (eye, nose, etc.) have the highest reconstruction error. The locations of the most frequent descriptors, simple edges, in the database are represented by green marks (*middle-right*). The location of the 8% least frequent descriptors in the database, mostly discriminative facial features, are represented by magenta marks (*right*). Image courtesy of Boiman, Shechtman, and Irani (2008).*

The most frequently used encoding is probably vector quantization (VQ, also known as hard-assignment): a local feature is assigned to its nearest neighbor in the dictionary. This approach is very intuitive, since each local feature is described by a single visual-word from the codebook. However, the encoding by only a single codeword is also the major drawback of the VQ encoding, and gives rise to the following problems. First, the undesired behavior of assigning two slightly different local descriptors to two different visual words. Second, the relatively high reconstruction error of VQ encoding, especially for the few discriminative local features in the image. Since discriminative local features are sporadic, they are most likely to be regarded as noise in the (k-means) clustering algorithm (Boiman et al., 2008). As a result the VQ leads to a high loss of information, as is illustrated in Figure 2.4.

To reduce the reconstruction error, local features could be encoded using soft-assignment, *i.e.* with a weighted combination of visual words from the codebook (Farquhar et al., 2005; Philbin et al., 2008; Van Gemert et al., 2008), this is also known as kernel-codebook-encoding. If the codebook is based on a MoG, the posterior probabilities of each Gaussian can be used as weight in the soft-assignment. A similar idea is used in sparse-coding, where the soft-assignment is used with a sparsity constraint on the reconstruction weights (Mairal et al., 2008b; Yang et al., 2009). Locality-constrained linear coding (LLC, Wang et al., 2010) reconstructs a local descriptor by using a soft-assignment over a few of the nearest visual words from the codebook. The obtained solution is not sparse in the ℓ_0 sense, but sparse in that only a few significant values are assigned to the nearest descriptors in the codebook. The Super-Vector encoding (Zhou et al., 2010) is closely related to the Fisher Vector, which we describe in the next section.

The computational time required for creating an image representation is largely dominated by the encoding of the local features. Independent of the chosen encoding technique, a

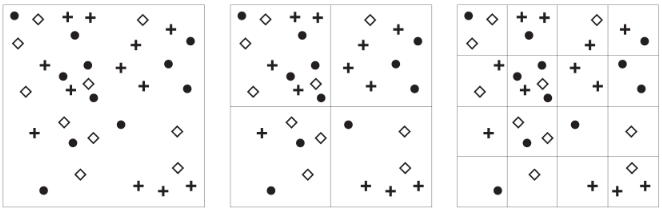


Figure 2.5 *Toy example of constructing a three-level spatial pyramid. The image has three feature types, indicated by circles, diamonds, and crosses. The image is subdivided in different spatial regions at three different levels of the pyramid, yielding a representation of single histogram (*left*), 4 histograms (*middle*) and 16 histograms (*right*). Image courtesy of Lazebnik, Schmid, and Ponce (2006).*

large number of local descriptors have to be matched against a large number of codebook entries, which is computationally costly.

Aggregation of encoded local features When computing the final histogram representation of an image, the encoded local features can be pooled in one of two ways: sum-pooling or max-pooling. In the case of sum-pooling, the local encoded features are additively combined into a histogram, which could be normalized by the number of samples (*i.e.* taking the mean instead of the sum). In the case of max-pooling, each bin of the histogram is assigned the highest value across the local feature encodings, in Yang et al. (2009) max-pooling was used in combination with LLC.

The BoV framework is, by design, invariant to the order of image patches, and thus to the layout of an image. To incorporate weak geometry in the BoV framework spatial pyramids are introduced by (Lazebnik et al., 2006), see Figure 2.5 for an illustration. The idea is to aggregate local features from different image regions. The spatial pyramid consists of several layers, where in layer l the image is divided in $2^l \times 2^l$ cells, each such a cell is described by a separate histogram, and concatenated into a final image representation. The use of spatial pyramids yields a significantly larger image representation: by using a 3 layer spatial pyramid the image representation is a concatenation of 21 histograms.

The winning system of the Pascal VOC 2008 Challenge (Everingham et al., 2008) also includes weak geometry, however it uses a different partitioning of the image: the whole image (1×1), the image quarters (2×2) and three horizontal bars (1×3), concatenating only 8 histograms. Recently, methods to learn spatial models were considered in (Sharma and Jurie, 2011; Bilen et al., 2011), where discriminatively the spatial layout is learned by successively splitting spatial cells. On different data sets, their methods compare favorably to using fixed spatial layout defined by the spatial pyramids of (Lazebnik et al., 2006).

Class-specific aggregation methods have been applied to the BoV framework, for example by using class-specific codebooks (Farquhar et al., 2005; Perronnin et al., 2006), or using a class-specific weighing of local features based on color attention maps (Khan et al., 2009). The size of the final image representation scales linearly with the number of classes, which is a disadvantage when used for a large number of classes.

Aggregation of local features has also been considered using class independent weighting, for example, based on foreground/background modeling to disregard too generic visual words (Zhang et al., 2009), or based on saliency maps to assign a higher weight to local features close to objects (De Campos et al., 2012).

2.1.2 Fisher Vector

In this section we describe the Fisher Vector for image representations, introduced by (Perronnin and Dance, 2007). This section is largely based on the Improved Fisher Vector of Perronnin et al. (2010b). The key idea is based on the Fisher kernel of Jaakkola and Haussler (1999), which is a principled approach to derive a kernel from a generative probabilistic model which can be used in a discriminative classifier. Discriminative classifiers, such as Support Vector Machines (SVM, Vapnik, 1995), have shown excellent results in (image) classification problems. Generative models, on the other hand, are attractive because they describe the model of the data generation process, and provide a way to deal with missing data or variable length data.

The Fisher kernel is based on the gradient of the log-likelihood of a generative probabilistic model with respect to its parameters. The gradient of the log-likelihood describes how the parameters contribute to the process of generating a particular example, and naturally preserves all structural assumptions that the model encodes. Let $X = \{\boldsymbol{x}_i\}_{i=1}^N$ be a set of N observations, in our case these could be a set of any of the local image features described in the previous section. Also lets define a generative model $p(X|\boldsymbol{\theta})$ with parameters $\boldsymbol{\theta}$. Then the *Fisher score*:

$$G_X = \nabla_{\boldsymbol{\theta}} \log p(X|\boldsymbol{\theta}). \tag{2.1}$$

Intuitively, the Fisher score describes the direction in which the parameters should be moved to better fit the data. Also, since the gradient is w.r.t. the parameters of the generative model, the dimensionality of the Fisher score G_X does not depend on the size of $\boldsymbol{x}$ nor on the number of observations N, but has a fixed length. The dimensionality of G_X equals the dimensionality of the parameter vector $\boldsymbol{\theta}$.

The Fisher kernel is given by the normalized inner product of two Fisher score vectors:

$$K(X_i, X_j) = G_{X_i}\, I_{\text{F}}^{-1}\, G_{X_j}^{\top}, \tag{2.2}$$

where I_{F} denotes the *Fisher information* matrix, given by:

$$I_{\text{F}} = \text{E}_X[G_X\, G_X^{\top}], \tag{2.3}$$

where the expectation is with respect to X under the distribution $p(X|\boldsymbol{\theta})$. The normalization by the Fisher information matrix causes the kernel to be invariant under a nonlinear transformation of the parameters of the generative model $\boldsymbol{\theta} \rightarrow \psi(\boldsymbol{\theta})$ (Bishop, 2006). However, only for a few specific simple parametric models the Fisher information matrix can be computed exactly. Therefore, it is often approximated, for example by the identity matrix (Jaakkola and Haussler, 1999), by an analytical approximation (Perronnin and Dance, 2007), or by using the sample average $I_{\text{F}} \approx \frac{1}{N} \sum_n G_{X_n} G_{X_n}^\top$, which corresponds to a whitening of the Fisher scores (Bishop, 2006).

In practice we will use the normalized Fisher score:

$$\mathcal{G}(X) = I_{\text{F}}^{-\frac{1}{2}} G_X, \tag{2.4}$$

as feature vector which we will denote as the *Fisher Vector* (FV). Note that learning a kernel classifier using the Fisher kernel, Eq. (2.2), is equivalent to learning a linear classifier on the FV, Eq. (2.4), which we will discuss in Section 2.2.

Fisher Vectors as image representations In order to use the Fisher kernel framework for image representations, we use a Mixture-of-Gaussians (MoG) as the probabilistic generative model of the local features following (Perronnin and Dance, 2007). We will show how the FV extends the popular bag-of-visual-words (BoV) by going beyond count statistics.

Lets define a MoG model where the probability of the local feature vector $\boldsymbol{x}_i$ is given by:

$$p(\boldsymbol{x}_i) = \sum_{k=1}^{K} \pi_k \, \mathcal{N}(\boldsymbol{x}_i; \boldsymbol{\mu}_k, \Sigma_k), \tag{2.5}$$

where π_k is the mixing weight, $\boldsymbol{\mu}_k$ the mean and Σ_k the covariance matrix of the Gaussian distribution of mixture component $k \in \{1, \dots, K\}$. We assume that the covariance matrix is diagonal, since any distribution can be approximated with an arbitrary precision by a weighted sum of Gaussians with diagonal covariance, and the computational cost when using diagonal covariances is much lower.

The MoG model assumes that the local image features are independent and identically distributed at random (i.i.d. assumption), *i.e.* each feature is drawn at random from the MoG, not taking into account the content of the image from the already drawn image features. For the Fisher Vector, we use the log-likelihood over a set of local image features $X = \{\boldsymbol{x}_i\}_{i=1}^N$, which given this i.i.d. assumption becomes:

$$\mathcal{L}(X) = \sum_{i=1}^{N} \ln p(\boldsymbol{x}_i). \tag{2.6}$$

To emphasis the relation with the BoV model we follow Krapac et al. (2011b) and define the mixing weights via the soft-max function:

$$\pi_k = \frac{\exp \alpha_k}{\sum_{k'} \exp \alpha_{k'}}, \tag{2.7}$$

which by construction, fulfills the constraints on the mixing weights that $\pi_k \geq 0$ and $\sum_k \pi_k = 1$, for any value of $\boldsymbol{\alpha}$.

The MoG defines the visual dictionary, and is thus supposed to describe the content of any image. Therefore the parameters $\boldsymbol{\theta} = \{\alpha_k, \boldsymbol{\mu}_k, \Sigma_k\}_{k=1}^{K}$ of the MoG are learned by Maximum Likelihood estimation, over a large set of local image features.

To compute the Fisher Vector, we use an analytical closed-form approximation of the Fisher information matrix (Perronnin and Dance, 2007), in which case the normalization of the gradient by $I_{\text{F}}^{-1/2}$ is simply a whitening of the dimensions. Let q_{ik} be the responsibility, or posterior probability, of Gaussian k for descriptor $\boldsymbol{x}_i$:

$$q_{ik} = \frac{\pi_k \, \mathcal{N}(\boldsymbol{x}_i; \boldsymbol{\mu}_k, \Sigma_k)}{\sum_{k'} \pi_{k'} \, \mathcal{N}(\boldsymbol{x}_i; \boldsymbol{\mu}_{k'}, \Sigma_{k'})}. \tag{2.8}$$

Then the whitened gradients w.r.t. the mixing weights ($\mathcal{G}_{\alpha_k}^X$), the mean ($\mathcal{G}_{\boldsymbol{\mu}_k}^X$) and covariance ($\mathcal{G}_{\Sigma_k}^X$) of Gaussian k are given by:

$$\mathcal{G}_{\alpha_k}^X = \frac{1}{N\sqrt{\pi_k}} \sum_i (q_{ik} - \pi_k), \tag{2.9}$$

$$\mathcal{G}_{\boldsymbol{\mu}_k}^X = \frac{1}{N\sqrt{\pi_k}} \sum_i q_{ik} \, \Sigma_k^{-\frac{1}{2}} (\boldsymbol{x}_i - \boldsymbol{\mu}_k), \tag{2.10}$$

$$\mathcal{G}_{\Sigma_k}^X = \frac{1}{N\sqrt{2\,\pi_k}} \sum_i q_{ik} \left(\Sigma_i^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}_k)^2 - 1 \right). \tag{2.11}$$

From these Fisher Vectors, we recognize that $\mathcal{G}_{\alpha_k}^X$, Eq. (2.9), is similar to a (soft-assign) BoV-histogram minus the mixing weights π_k. We also observe that the Fisher Vectors on a MoG model go beyond these 0-th order statistics of word counting, by considering the 1-st and 2-nd order statistics (Eqs. (2.10-2.11)).

The final Fisher Vector we use is a concatenation of the whitened gradients for the mean and standard deviation, $\mathcal{G}(X) = [\mathcal{G}_{\boldsymbol{\mu}_k}^X, \mathcal{G}_{\Sigma_k}^X]_{k=1}^{K}$. The gradient w.r.t. the weight parameters adds little additional information, and is therefore omitted. Let D denote the dimensionality of the local descriptors $\boldsymbol{x}_i$, then $\mathcal{G}(X)$ is thus $2KD$-dimensional. This yields a significantly larger representation than using a BoV-histogram, which would be K-dimensional. The FV has the computational advantage that we can use a smaller number of visual words K, since the appearance per visual word is coded more precisely.

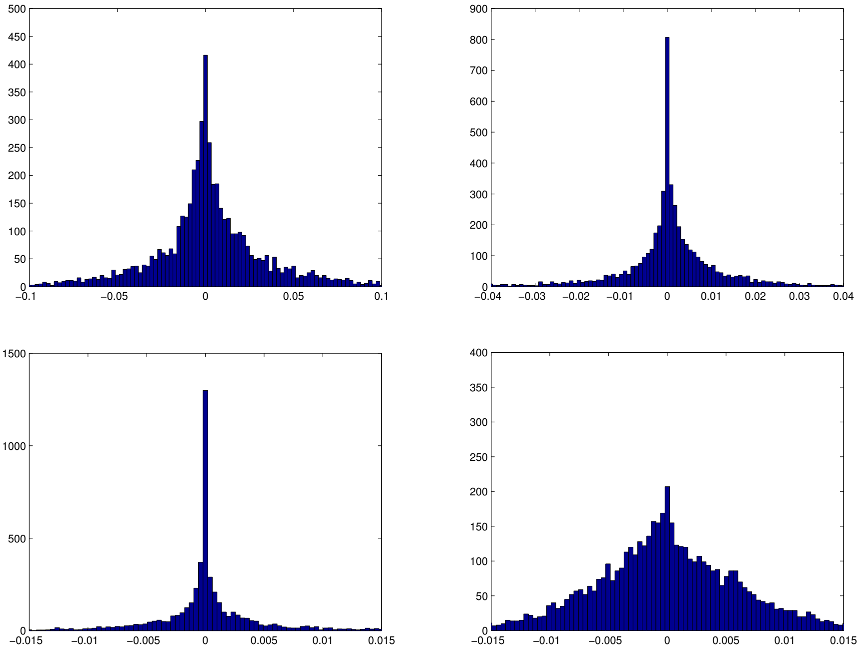


Figure 2.6 *Distribution of the values in the first dimension of the Fisher Vector, showed for 16, 64 and 256 Gaussians (*top left, top right, and bottom left*), without power normalization, and for 256 Gaussians with power normalization (*bottom right*), using $\alpha = .5$; note the different scales. Image taken from (Perronnin et al., 2010b).*

Power normalization The power normalization of the FV is introduced based on an empirical observation: as the number of Gaussians increases, Fisher Vectors become sparser. This effect can be easily explained: as the number of Gaussians increases, fewer descriptors $\boldsymbol{x}_i$ are assigned with a significant weight q_{ik} to each Gaussian. When from a set of descriptors X all descriptors $\boldsymbol{x}_i$ are assigned with near zero weight to a given Gaussian k (*i.e.* $\forall i \; q_{ik} \approx 0$), then also its gradient vectors $\mathcal{G}^X_{\boldsymbol{\mu}_k}$, and $\mathcal{G}^X_{\Sigma_k}$ are close to null, see Eqs. (2.10-2.11). Hence, as the number of Gaussians K increases, the distribution of features in a given dimension becomes more sharply distributed around zero, as illustrated in Figure 2.6.

To “unsparsify” the Fisher Vector, we apply power-normalization:

$$f(z) = \operatorname{sign}(z)|z|^\alpha \tag{2.12}$$

where $0 \leq \alpha \leq 1$ is a parameter of the normalization. The optimal value of α may vary with the number K of Gaussians in the MoG; in practice we use $\alpha = \frac{1}{2}$ for all values of K, which equals to a generalized square rooting of the vector. In Figure 2.6 the effect of this normalization is shown for a MoG with $K = 256$.

The power-normalization can also be justified from different viewpoints. First, assuming that the number of local samples assigned to a specific Gaussian k follows a Poisson dis-

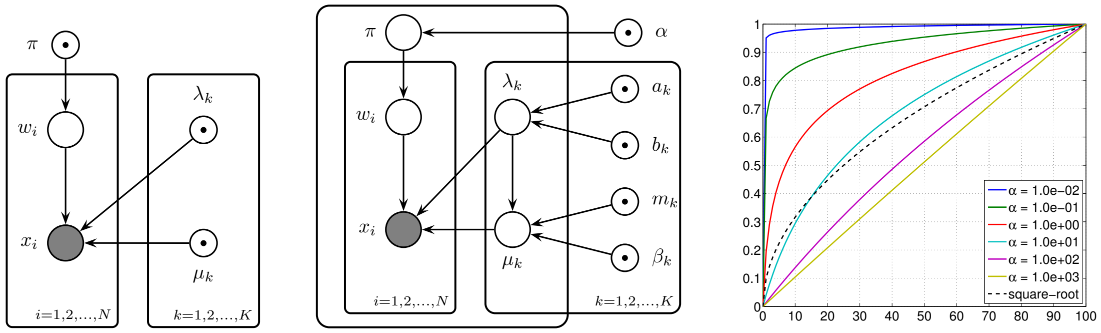


Figure 2.7 *Graphical representation of the i.i.d. MoG model (*left*) and the non-i.i.d. latent MoG model (*middle*). The index i runs over the N local features in an image, and index k over visual words, observed variables are shaded, and (hyper-)parameters are marked with a central dot in the node. The influence of the α parameter is shown (*right*), and compared to square-rooting; the values have been rescaled to the range* $[0, 1]$. *Image courtesy of Cinbis, Verbeek, and Schmid (2012).*

tribution, then the sum of these local samples follows a compound Poisson distribution. In the compound Poisson distribution the values of the mean and variance are dependent, the power normalization acts as a variance stabilizing transformation, which corrects for this dependence (Jégou et al., 2012). Second, it reduces the influence of bursty visual elements, which were shown to corrupt the image similarity (Jégou et al., 2009). Burstiness, is the phenomenon that once a specific visual element appears in an image, it tends to reappear more often in that image than predicted by the statistically independent model.

A third interpretation is given in Cinbis et al. (2012), where it is shown that the benefit of square-rooting can be explained by replacing the i.i.d. MoG model with a non-i.i.d. model which generates similar image representations. They introduce a non-i.i.d. model by treating the parameters of the MoG as latent variables, and placing conjugate priors on these latent variables, see Figure 2.7 for an illustration. Using these models, integrating over the latent variables, renders all local regions dependent. Their image representation is a Fisher Vector w.r.t. the parameters of the conjugate priors, and their representation naturally involves discounting transformations similar to taking the square-roots. For example, the gradient to the hyper-parameter α_k, controlling the conjugate prior on the weights of the multinomial distribution of the visual words, is given by the digamma function $\psi(\alpha_k + n_k)$, up to additive constants, where n_k is the number of appearances of word k in the image, see Figure 2.7 for an illustration of this transformation for various values of α and n, together with square-rooting.

ℓ_2 **normalization** Here, we will show that the FV approximately discards image independent background information and focuses on image specific content. However the FV depends on the proportion of the image specific content w.r.t. the background information, to remove this dependency we apply ℓ_2 normalization.

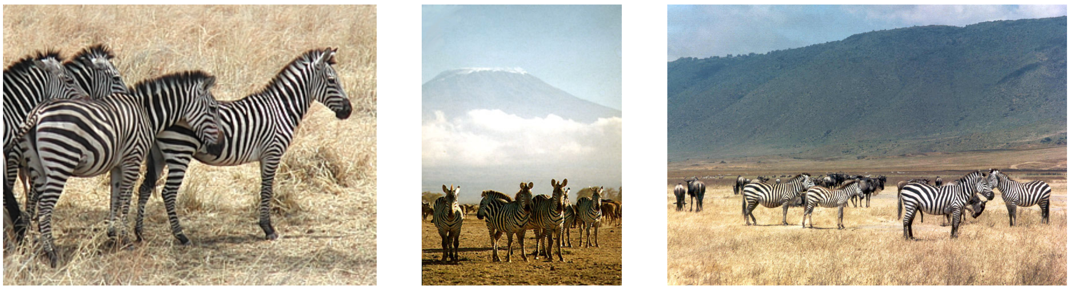


Figure 2.8 *Three photos depicting zebras, but seen in different configurations and settings, even though the Fisher Vector approximately disregards the background information, the proportion of image-specific information differs per photos. Without ℓ_2 normalization, this yields different image signatures, while applying the ℓ_2 normalization makes the signature independent of the proportion of image-specific information, see text for more details. Images from the Animals with Attributes data set (Lampert et al., 2009).*

Given that the FV is computed over a large set of local descriptors $X = \{x_i\}_{i=1}^N$, using an i.i.d. MoG model, the law of large numbers states that the sample average converges to the expected value when N increases. Therefore, we can rewrite the Fisher score of Eq. (2.1) as:

$$G_X = \frac{1}{N} \sum_i \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{x}_i|\boldsymbol{\theta}) \approx \nabla_{\boldsymbol{\theta}} \mathrm{E}_{x \sim p}[\log p(\boldsymbol{x}_i|\boldsymbol{\theta})]. \tag{2.13}$$

Now, lets assume that the feature generation process p can be decomposed into a mixture:

$$p(\boldsymbol{x}) = \omega\, q(\boldsymbol{x}) + (1-\omega) p(\boldsymbol{x}|\boldsymbol{\theta}), \tag{2.14}$$

where $q(\boldsymbol{x})$ is an image specific distribution, $p(\boldsymbol{x}|\boldsymbol{\theta})$ is the image-independent MoG model of Eq. (2.5), and $\omega \in [0,1]$ represents a mixing weight. Then we can rewrite Eq. (2.13) to:

$$G_X \approx \omega\, \nabla_{\boldsymbol{\theta}} \mathrm{E}_{x \sim q}[\log p(\boldsymbol{x}_i|\boldsymbol{\theta})] + (1-\omega)\, \nabla_{\boldsymbol{\theta}} \mathrm{E}_{x \sim p_\theta}[\log p(\boldsymbol{x}_i|\boldsymbol{\theta})], \tag{2.15}$$

where p_θ denotes the MoG model. The parameters $\boldsymbol{\theta}$ of the MoG are estimated to maximize $\mathrm{E}_{x \sim p_\theta}[\log p(\boldsymbol{x}_i|\boldsymbol{\theta})]$, therefore the gradient has to fulfill $\nabla_{\boldsymbol{\theta}} \mathrm{E}_{x \sim p_\theta}[\log p(\boldsymbol{x}_i|\boldsymbol{\theta})] \approx 0$. Consequently, the Fisher score can be approximated by:

$$G_X \approx \omega\, \nabla_{\boldsymbol{\theta}} \mathrm{E}_{x \sim q}[\log p(\boldsymbol{x}_i|\boldsymbol{\theta})], \tag{2.16}$$

which considers only the image-specific information.

However, we observe that the signature still depends on the proportion of image-specific information ω. As a result, two images containing the same object but different amounts of background information due to scale, viewpoint changes or small variations in environment, will have different signatures. Small objects with a small ω value will be especially difficult to detect, see Figure 2.8 for some illustrative examples.

To remove the dependence on ω, we ℓ_2 normalize the vector G_X, or similar the vector $\mathcal{G}_X$, which is strictly equivalent of replacing the Fisher kernel of Eq. (2.2), by:

$$\tilde{K}(X_i, X_j) = \frac{K(X_i, X_j)}{\sqrt{K(X_i, X_i)K(X_j, X_j)}}, \tag{2.17}$$

While the ℓ_2 norm of the Fisher Vector could contain class-specific information, experimentally it has been shown that this normalization leads to a significant increase of performance (Perronnin et al., 2010b).

Fisher Vector compression The described Fisher Vector framework yields very high dimensional image representations, typically 64K-1M dimensional FV are used. In order to reduce storage cost, and to improve computational throughput, we apply Product Quantization (PQ, Gray and Neuhoff, 1998). PQ is a lossy data compression technique, introduced by Jégou et al. (2011) for large scale nearest neighbor search using SIFT vectors. PQ has been applied on FV for image classification and have shown state-of-the-art performance (Sánchez and Perronnin, 2011).

Product quantization is based on vector quantization, where the idea is to represent a vector by a single codeword, like the hard assignment which is used to encode the local features in the BoV framework. However, instead of representing the whole high-dimensional feature vector by a single codeword, PQ splits the vector into a set of M sub-vectors, each of the sub-vectors has the size $G = D/M$, where D is the original feature size. Clearly, for $G = D$ we obtain vector quantization, while for $G = 1$ we obtain scalar quantization. The parameter G allows to trade off the storage cost and the reconstruction error.

In practice we use $G = 8$ dimensions per sub-vector, and apply vector-quantization on each sub-vector independently. Per sub-quantizer we use 256 centroids, such that each sub-vector can be represented with a single byte, indexing the corresponding centroid. This allows to reduce the storage cost of the high dimensional FV by a factor of about 32. For example, the 64K dimensional FVs of 1.2M images would require 320GB, while by employing PQ this is reduced to about 10GB.

Furthermore we can take into account the sparsity structure of FVs. While, on average, the FV is dense (more than 50% non-zeros), the zeros are not randomly distributed. Indeed, if no patch is assigned to the k-th Gaussian, then the gradient signal for this component is zero. Hence, the FV is block sparse and instead of encoding the sparsity on a per-dimension basis, it can be encoded on a per Gaussian basis.

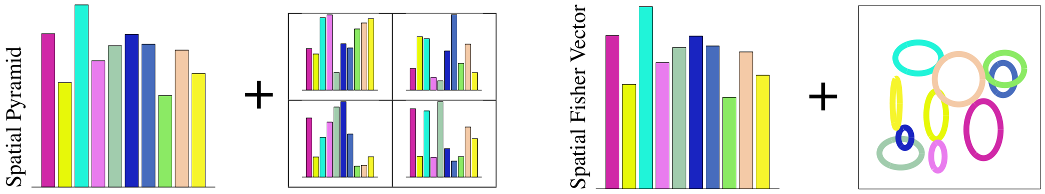


Figure 2.9 *Illustration of the Spatial Fisher Vector, in contrast to the spatial pyramid (*left*) which concatenates visual word histograms of the spatial cells, the spatial Fisher Vector models spatial layout by the mean and variance of the occurrences of each visual word. Images courtesy of Krapac, Verbeek, and Jurie (2011b).*

When learning classification models, usually the training set is large and each training image needs to be accessed multiple times during learning. The test set on the other hand, is often much smaller and each image needs only the be accessed once to predict its class. Therefore we apply PQ encoding only on the training set, and use the uncompressed image representations for the test images. Such an asymmetric approach has shown to yield a (slightly) better performance (Jégou et al., 2011).

Relation to other methods and extensions The described FV framework relates to other methods which extend the standard BoV framework. For example, the VLAD descriptor (Jégou et al., 2010) was introduced as an extension of the BoV framework, and it can be seen as a simplified non-probabilistic version of the FV: it uses k-means codebook training, VQ encoding, and takes only the derivative w.r.t. the mean. The VLAD descriptor is, due to the VQ encoding, a highly sparse representation which has been successfully applied to very large scale image retrieval.

The Super-Vector encoding (Zhou et al., 2010), is derived from a function analysis of the classification function. The obtained encoding is similar to the VLAD and FV descriptor, the Super-Vector for an image patch $\boldsymbol{x}_i$ is defined as:

$$F_{\boldsymbol{x}_i}^\top = [q_{ik}\ s,\ q_{ik}\ (\boldsymbol{x}_i - \boldsymbol{\mu}_k)^\top]_{k=1}^K, \tag{2.18}$$

where $\boldsymbol{\mu}_k$ is the descriptor of code word k, q_{ik} is the posterior probability of code word k for patch i, as defined for the FV in Eq. (2.8), and where s is a balancing constant set by cross-validation. The image representation of a set of features X, is $F_X = \sum_i F_{\boldsymbol{x}_i}$.

Compared with the Fisher Vector, the Super-Vector uses only the first order statistics and includes the mixing weights. It has a $K(D+1)$ dimensional representation.

Finally, we note the recent work of Krapac et al. (2011b) where a Spatial Fisher Vector is introduced. This Fisher Vector encodes per visual word also the spatial mean and variance of image regions associated with the visual word, see Figure 2.9 for an illustration.

Their generative probabilistic model, which combines a single spatial Gaussian per visual word with a MoG visual appearance model is defined as:

$$p(\tilde{\boldsymbol{x}}) = \sum_k \pi_k \, \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_k, \Sigma_k) \, \mathcal{N}(\boldsymbol{l}; \boldsymbol{m}_k, S_k) \tag{2.19}$$

where $\tilde{\boldsymbol{x}} = (\boldsymbol{x}, \boldsymbol{l})$ is the tuple of the local (SIFT) descriptor $\boldsymbol{x}$ and its location $\boldsymbol{l}$. They use the gradients w.r.t. $\boldsymbol{\theta} = \{\pi_k, \mu_k, \Sigma_k, \boldsymbol{m}_k, S_k\}_{k=1}^{K}$, yielding a $K(1+2D+2d)$ dimensional vector, where d is the dimensionality of the location, and D is the dimensionality of the local features. Their experimental results show that their method performs about equal to the spatial pyramids (Lazebnik et al., 2006) approach, but with image representations which are about four times more compact.

2.1.3 Comparison of image representations

In a recent study of Chatfield et al. (2011), several popular image representations and encodings have been compared, including the discussed BoV framework and the FV framework. While almost all authors compare their method on the PASCAL VOC 2007 benchmark (Everingham et al., 2007) and/or the Caltech-256 benchmark (Griffin et al., 2007), performance is difficult to asses since small differences in local features, sampling approaches and parameter tuning are not taken into account. This also results in difficulties to reproduce the results reported by the authors of new representations.

Chatfield *et al.*, have re-implemented[2] all encoding techniques they compare, allowing for a fair evaluation of the different methods. In some cases, their obtained results differ significantly from the ones presented in the original publications, which emphasizes the importance of such a thorough evaluation.

In Table 2.1, we show the performance in mean average precision (MAP) on the PASCAL VOC 2007 classification task. Since our goal is to compare the different encodings, and not the influence of the codebook size and sampling strategy, we only show a single entry per method, and thus a subset of the results reported in Chatfield et al. (2011). The methods we compare all use a dense sampling strategy (every 3 pixels), extract 128 dimensional SIFT features, and use spatial pyramids (Lazebnik et al., 2006). The codebook size for the BoV methods is 25K, for the Super-Vector it is 1,024, and for the FV 256 Gaussians are used. For the FV framework the SIFT features are reduced to 80 dimensions using PCA. Due to the different encoding methods the sizes of the image representations are different, each spatial cell is described with a 25K (BoV), 129K (Super-Vector), or 40K (FV) feature vector. For the kernels used by the classifiers, we report the results using the kernel (linear or χ^2) proposed in the original publication, or the current standard (like χ^2 for BoV-VQ).

[2] Available at http://www.robots.ox.ac.uk/~vgg/software/enceval_toolkit.

Method	Kernel	Vocabulary Size	MAP
Fisher Vector	Linear	256	61.7
Super-Vector	Linear	1024	58.2
BoV-LLC	Linear	25K	57.3
BoV-KCB	χ^2	25K	56.3
BoV-VQ	χ^2	25K	55.3

Table 2.1 *Overview of different image representations and their performance on the Pascal VOC 2007 data set.All methods use the same densely extracted SIFT features. Table courtesy of Chatfield, Lempitsky, Vedaldi, and Zisserman (2011).*

From the results in Table 2.1 we conclude that any of the advanced encodings outperform the baseline BoV-VQ approach, which obtains 55.3% MAP. In this evaluation, the FV framework obtains the best performance of 61.7% MAP, an increase of more than 6 absolute percent points in MAP compared to BoV-VQ. Given these results, we select the FV as the image representation of our choice in the rest of this dissertation.

2.2 Image classification methods

In this section we review machine learning techniques relevant for image classification. First, we discuss parametric classification and compare logistic-regression models and support vector machines (SVMs), for binary, multi-class and structured prediction tasks. Then, we describe k-nearest neighbor (k-NN) classification, which is a non-parametric method. Since the performance of k-NN classification depends critically on the distance metric, we describe two metric learning approaches: LMNN (Weinberger et al., 2006) and TagProp (Guillaumin et al., 2009a). Finally, we compare the performance of SVM classifiers and TagProp on the ImageClef'10 data set.

2.2.1 Linear classification and extensions

In this section we discuss linear classification methods in a supervised learning setting, that is we assume to have a training set consisting of n images with its corresponding ground truth label: $\{\boldsymbol{x}_1, y_1, \ldots \boldsymbol{x}_n, y_n\}$. In contrast to the previous section, $\boldsymbol{x}_i$ describes the whole image i (for example with a FV or a BoV-histogram), and not a local feature in the image[3]. We first discuss binary classification, and then generalize to multi-class and structured-output classification.

Binary linear classification The goal of binary classification is to learn a model that can predict whether a specific concept is present, or relevant, for a given image, *e.g.* it answers the question: "*Is there a bicycle present in the image?*". For this purpose, we construct a discriminant function:

$$f(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x}, \tag{2.20}$$

which predicts the object as being present in the image when $f(\boldsymbol{x}) \geq 0$, and not present otherwise. For the clarity of the presentation we assume that $\boldsymbol{w}$ and $\boldsymbol{x}$ are *augmented* vectors, *i.e.* they are extended to accommodate a bias term such that $\boldsymbol{w}^\top \boldsymbol{x} = \tilde{\boldsymbol{w}}^\top \tilde{\boldsymbol{x}} + b$, with $\boldsymbol{w} = [b, \tilde{\boldsymbol{w}}^\top]^\top$, and $\boldsymbol{x} = [1, \tilde{\boldsymbol{x}}^\top]^\top$.

When the class-labels are defined as $y = \{+1, -1\}$, where $+1$ denotes the presence of the object in an image, then correct classification is achieved when $\big(y\, f(\boldsymbol{x})\big) > 0$. The objective is to find the parameter vector $\boldsymbol{w}$ which minimizes the *classification error* on the training set:

$$\sum_{i=1}^{n} [\![\big(y_i\, f(\boldsymbol{x}_i)\big) \leq 0]\!], \tag{2.21}$$

[3] Since the discussed techniques in this chapter are generic, image and input are used interchangeably.

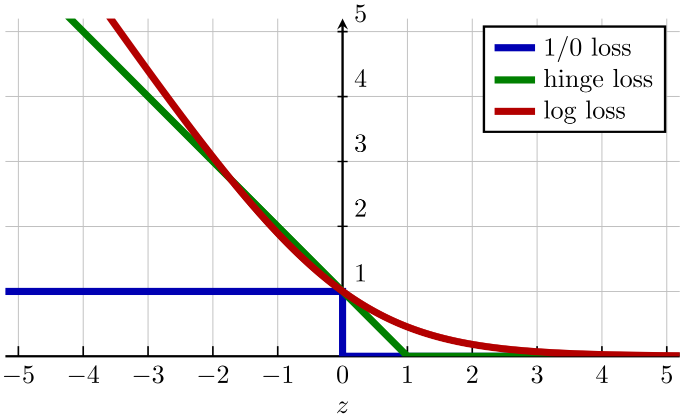


Figure 2.10 *Illustration of the loss for different values of z, shown for different loss functions, in (*blue*) the zero-one loss or classification error, in (*red*) the log loss, and in (*green*) the hinge loss. See text for details.*

where we use Iversons bracket notation: $[\![z]\!] = 1$ if the condition z is true and zero otherwise. The classification error for an image i is also known as the *zero-one loss*, since its value is either zero or one. Unfortunately, it is difficult to minimize the zero-one loss, since it is non-convex and non-differentiable in the parameters $\boldsymbol{w}$.

Therefore, in order to learn the parameter vector $\boldsymbol{w}$, we make use of an approximate, convex loss function $\ell(\boldsymbol{x}_i, y_i, \boldsymbol{w})$ which measures the discrepancy between the ground-truth label y_i and the prediction arising from using $\boldsymbol{w}$ for an input i. Two popular approximations of the zero-one loss are the *log loss* and the *hinge loss*, which are both an upper-bound on the zero-one loss, see Figure 2.10 for an illustration.

As the learning objective we use regularized empirical loss minimization:

$$\min_{\boldsymbol{w}} F(\boldsymbol{w}) = \frac{1}{n} \sum_{i=1}^{n} \ell(\boldsymbol{x}_i, y_i, \boldsymbol{w}) + \lambda \, R(\boldsymbol{w}), \tag{2.22}$$

where R is a regularization function, and λ is a trade-off parameter between the loss term and the regularizer. Since the loss function ℓ is an upper-bound on the classification error, λ weighs the number of training errors with the model complexity. The regularizer has two important functions, it can overcome over-fitting of the data by penalizing the model complexity, and in the case that multiple instantiations of $\boldsymbol{w}$ lead to the same minimum loss, *e.g.* when the data is linear separable, it makes the objective well defined and selects the $\boldsymbol{w}$ with lowest regularization cost.

Next we describe *logistic regression* which uses the log loss, and *Support Vector Machines* (SVMs, Vapnik, 1995) which uses the hinge loss.

Logistic regression Logistic regression is a discriminative probabilistic model for classification. We define the posterior probability for label $y = +1$ given an input $\boldsymbol{x}$ as:

$$p(y = +1|\boldsymbol{x}) = \sigma(\boldsymbol{w}^\top \boldsymbol{x}), \tag{2.23}$$

where $\sigma(z) = \big(1 + \exp(-z)\big)^{-1}$ is the sigmoid function, which is also known as the logistic function, hence the name logistic regression. The probability for label $y = -1$ is defined as: $1 - p(y = +1|\boldsymbol{x}) = \sigma(-\boldsymbol{w}^\top \boldsymbol{x})$.

The loss function ℓ is defined as the negative log-likelihood on correct prediction:

$$\ell_{\mathrm{L}}(\boldsymbol{x}_i, y_i, \boldsymbol{w}) = -\log p(y_i|\boldsymbol{x}_i, \boldsymbol{w}) = -\log \sigma(y_i\, \boldsymbol{w}^\top \boldsymbol{x}_i). \tag{2.24}$$

In Figure 2.10 the log loss $\ell_{\mathrm{L}}(z) = -\log \sigma(z)/\log 2$ is shown, to tightly upper-bound the zero-one loss at the point $(0, 1)$.

We can interpret Eq. (2.22) as a *maximum-a-posteriori* estimation, by using the negative log-likelihood of a prior distribution on $\boldsymbol{w}$ as regularizer. For logistic regression we use a Gaussian prior, with zero mean and unit covariance. The negative log-likelihood of this prior equals the ℓ_2 norm, up to an additive constant c, of the parameter vector $\boldsymbol{w}$:

$$R(\boldsymbol{w}) = -\log \mathcal{N}(\boldsymbol{w}; \mathbf{0}, \mathbf{I}) = \frac{1}{2}||\boldsymbol{w}||_2^2 + c. \tag{2.25}$$

To minimize the regularized empirical loss, we use gradient based methods. Straightforward derivation of the loss function and regularizer provides the following gradients:

$$\begin{aligned} \nabla_{\boldsymbol{w}}\, \ell_{\mathrm{L}}(\boldsymbol{x}_i, y_i, \boldsymbol{w}) &= -y_i\, \sigma(-y_i\, \boldsymbol{w}^\top \boldsymbol{x}_i)\, \boldsymbol{x}_i, & (2.26)\\ &= \big(p(y = +1|\boldsymbol{x}_i, \boldsymbol{w}) - [\![y_i = +1]\!]\big)\, \boldsymbol{x}_i, & (2.27)\\ \nabla_{\boldsymbol{w}}\, \lambda\, R &= \lambda\, \boldsymbol{w}. & (2.28) \end{aligned}$$

Support Vector Machines The aim of SVMs (Vapnik, 1995) is to find the separating hyperplane that has the largest distance to the nearest training data points of both classes. Therefore, SVMs are also known as large-margin classifiers. The key insight is that a larger margin leads to better generalization performance, and hence, to a higher performance on the test set. This insight is reflected in the hinge loss function:

$$\ell_{\mathrm{H}}(\boldsymbol{x}_i, y_i, \boldsymbol{w}) = \big[1 - y_i\, \boldsymbol{w}^\top \boldsymbol{x}_i\big]_+, \tag{2.29}$$

where $[z]_+ = \max\{0, z\}$ is the hinge function, and the margin is enforced by the offset, *i.e.* to obtain zero loss the scoring should be $y_i\, \boldsymbol{w}^\top \boldsymbol{x}_i \geq 1$. In Figure 2.10, the hinge loss $\ell_{\mathrm{H}}(z) = [1 - z]_+$ is shown.

The subgradient of the hinge loss is given by:

$$\nabla_{\boldsymbol{w}}\, \ell_{\mathrm{H}}(\boldsymbol{x}_i, y_i, \boldsymbol{w}) = -y_i\, [\![\ell_{\mathrm{H}}(\boldsymbol{x}_i, y_i, \boldsymbol{w}) > 0]\!]\, \boldsymbol{x}_i. \tag{2.30}$$

As regularizer for SVMs we also use the ℓ_2 norm of the parameter vector $\boldsymbol{w}$, similar to logistic regression, $R(\boldsymbol{w}) = \frac{1}{2}||\boldsymbol{w}||_2^2$. Other regularizers with different properties, like the sparsity inducing ℓ_1 norm, could also be used, leading to different optimization problems.

The SVM classifier does not output a probability, but it assigns class $y = \mathrm{sign}(\boldsymbol{w}^\top \boldsymbol{x})$ to an image. To obtain probabilistic calibrated outcomes, the method known as *Platt-scaling* (Platt, 1999) is frequently used. The idea is to use a small validation set to learn the parameters α and β of the logistic regression function $p(y|\boldsymbol{x}) = \sigma(\alpha\, \boldsymbol{w}^\top \boldsymbol{x} + \beta)$.

Extension to non-linear classification To increase the discriminative power of both models, we can use a non-linear transformation of the features $\boldsymbol{x} \rightarrow \phi(\boldsymbol{x})$. When using such a non-linear parametrization, the discussed models remain linear in the parameters $\boldsymbol{w}$, but can construct a non-linear decision boundary in the original input $\boldsymbol{x}$.

We can avoid computing the transformation $\phi(\boldsymbol{x})$ explicitly by applying the “kernel trick”. The *Representer Theorem* (Schölkopf and Smola, 2002) states that the optimal parameter vector $\boldsymbol{w}$ is a weighted sum of the features of the training images:

$$\boldsymbol{w} = \sum_{m=1}^{n} \alpha_m\, \phi(\boldsymbol{x}_m), \tag{2.31}$$

where n is the number of training samples, and α_m denotes the weight for training image m. The kernel-trick is essentially to apply this theorem and rewrite:

$$\boldsymbol{w}^\top \phi(\boldsymbol{x}) = \sum_m \alpha_m\, \phi(\boldsymbol{x}_m)^\top \phi(\boldsymbol{x}) = \sum_m \alpha_m\, k(\boldsymbol{x}_m, \boldsymbol{x}), \tag{2.32}$$

where k denotes a kernel-function.

According to Mercer’s Theorem any positive definite kernel function[4] equals a dot-product in some (possibly infinite dimensional) feature space (Vapnik, 1995; Schölkopf and Smola, 2002). Therefore we can use any valid similarity function k, such as the Gaussian kernel, $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\frac{1}{2\sigma^2}||\boldsymbol{x}_i - \boldsymbol{x}_j||)$, which feature vector $\phi(\boldsymbol{x})$ has infinite dimensionality, or the χ^2-kernel, $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sum_d \frac{2\, x_{id}\, x_{jd}}{x_{id} + x_{jd}}$, which is popular for comparing

[4] A function k is called a positive definite kernel function if:

- k is symmetric, *i.e.* $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = k(\boldsymbol{x}_j, \boldsymbol{x}_i)$ for all $\boldsymbol{x}_i, \boldsymbol{x}_j$, and
- for any set of n points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$, the kernel-matrix K, with $K_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, is positive semi-definite, *i.e.* for all vectors $\boldsymbol{a} \in \mathbb{R}^n$ holds that $\sum_{i,j=1}^{n} a_i a_j K_{ij} \geq 0$.

BoV-histograms. Different kernels can encode different image similarities, a task specific weighted sum of several kernels can be obtained by *e.g.* multiple-kernel-learning (Sonnenburg et al., 2006).

The ℓ_2 regularizer on the parameter vector $\boldsymbol{w}$ can also be written in terms of the kernel:

$$R(\boldsymbol{w}) = \frac{1}{2}||\boldsymbol{w}||_2^2 = \frac{1}{2}\sum_{i,j} \alpha_i\alpha_j\, k(\boldsymbol{x}_i, \boldsymbol{x}_j), = \frac{1}{2}\boldsymbol{\alpha}^\top K\boldsymbol{\alpha}, \tag{2.33}$$

where K is the $n \times n$ kernel-matrix, and $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_n]^\top$. The gradients of the log loss, the hinge loss and the regularizer w.r.t. the parameter vector $\boldsymbol{\alpha}$ are given by:

$$\nabla_{\boldsymbol{\alpha}}\, \ell_{\mathrm{L}}(\boldsymbol{x}_i, y_i, \boldsymbol{\alpha}) = \big(p(y = +1|\boldsymbol{x}_i, \boldsymbol{w}) - [\![y_i = +1]\!]\big)\, \boldsymbol{k}_i, \tag{2.34}$$

$$\nabla_{\boldsymbol{\alpha}}\, \ell_{\mathrm{H}}(\boldsymbol{x}_i, y_i, \boldsymbol{\alpha}) = -y_i\, [\![\ell_{\mathrm{H}}(\boldsymbol{x}_i, y_i, \boldsymbol{\alpha}) > 0]\!]\, \boldsymbol{k}_i, \tag{2.35}$$

$$\nabla_{\boldsymbol{\alpha}}\, \lambda\, R = \lambda\, K\boldsymbol{\alpha}, \tag{2.36}$$

where $\boldsymbol{k}_i$ is the i-th column of the kernel matrix K.

An advantage of using kernels is that they can allow for efficient computation of dot-products in high or infinite dimensional feature spaces. Moreover, the explicit feature map $\phi(\cdot)$ is not required to be known, as long as the kernel function $k(\cdot, \cdot)$ can be evaluated. However, pre-computing and storing the $n \times n$ kernel-matrix is impractical for large data sets. Therefore, research has recently focused on using explicit embeddings $\phi(\cdot)$ that correspond to a kernel function, and to approximate appropriate kernel functions (Maji et al., 2008; Maji and Berg, 2009; Vedaldi and Zisserman, 2011).

Multi-class classification So far, we have discussed the binary classification problem, where the input belongs to one out of two classes. Here, we generalize to the task of multi-class classification where the labels are defined as $y \in \{1, \ldots, C\}$, and the goal is to assign an input $\boldsymbol{x}$ to the correct class.

In the probabilistic framework the generalization to multi-class logistic regression is straightforward, by replacing the Bernoulli distribution with a multinomial distribution. We define the probability of class c for an input $\boldsymbol{x}$ as the soft-max over the linear functions:

$$p(c|\boldsymbol{x}) = \frac{\exp\left(\boldsymbol{w}_c^\top \boldsymbol{x}\right)}{\sum_{c'} \exp\left(\boldsymbol{w}_{c'}^\top \boldsymbol{x}\right)}. \tag{2.37}$$

The natural loss function is still the negative log-likelihood of predicting the correct class $\ell_{\mathrm{L}}(\boldsymbol{x}_i, y_i, \boldsymbol{w}) = -\log p(y_i|\boldsymbol{x}_i)$, where $\boldsymbol{w} = [\boldsymbol{w}_1, \ldots, \boldsymbol{w}_C]$.

For SVMs the generalization from the binary case to the multi-class classification setting is less straightforward. However, from the observation that the hinge loss is an upper-bound on the binary zero/one loss, there are two obvious generalizations.

First, Weston and Watkins (1999) have introduced an upper-bound of the multi-class zero/one loss, given by:

$$\ell_{\text{WW}}(\boldsymbol{x}_i, y_i, \boldsymbol{w}) = \sum_{c \neq y_i} \left[1 - \boldsymbol{w}_{y_i}^\top \boldsymbol{x}_i + \boldsymbol{w}_c^\top \boldsymbol{x}_i\right]_+ . \tag{2.38}$$

This loss compares the ground-truth class to all other classes, and each pair which is not separated by at least a margin contributes to the loss term. This is clearly an upper bound on the multi-class zero/one loss, because at least one pair will incur a penalty when the correct class does not obtain the highest score. The bound is tight when $\ell_{\text{WW}} = 0$.

Second, Crammer and Singer (2001) have introduced a different upper-bound:

$$\ell_{\text{CS}}(\boldsymbol{x}_i, y_i, \boldsymbol{w}) = \left[1 - \boldsymbol{w}_{y_i}^\top \boldsymbol{x}_i + \max_{c \neq y_i} \boldsymbol{w}_c^\top \boldsymbol{x}_i\right]_+ . \tag{2.39}$$

The idea behind this loss is that when the highest scoring class is at least a margin away from the correct class, all other classes are as well. Therefore only the highest scoring other class is considered in the loss function. This approach provides a tighter bound on the zero/one loss compared to ℓ_{WW}. Both formulations reduce to binary SVMs for $C = 2$.

While both these SVM losses are theoretically sound, in practice multi-class classification problems are solved using a combination of several binary classifiers, following:

- *One-versus-rest* (OVR), which learns C binary classifiers independently; each classifier c is trained to discriminate class c versus all other classes, and input i is assigned to the classifier with the highest score.
- *One-versus-one* (OVO), which learns $C(C-1)$ binary classifiers; each classifier discriminates between 2 classes, and an input i is assigned using a majority vote.

A disadvantage of these approaches is that due to the decoupled learning of the vectors $\boldsymbol{w}_c$, the score functions are, in principle, not comparable and might lead to ambiguity in the output scores. In practice, OVR is frequently used because it is easier to implement, has better scaling properties, and often outperforms the mentioned multi-class losses, see *e.g.* (Perronnin et al., 2012).

Structured-output prediction In this section we consider the case where we want to predict simultaneously the outcome of L binary labels. This is, for example, relevant for multi-label classification and foreground/background segmentation of images.

Using the classification methods introduced so far, we have two options: we could use the multi-class approach and consider each possible output as a separate class; or we could predict each label independently using L binary classifiers. The problem with the first approach is that we obtain an exponential number of 2^L classes, and thus we have to estimate an exponential number of parameters from the data. The problem with the second approach is that it ignores any interdependence between the output labels.

In structured-output prediction the aim is to model dependencies among the output labels, but without using an exponential number of parameters. In order to do so, we generalize the multi-class algorithms, to use a more general energy or compatibility function. We define the energy function E, between an input $\boldsymbol{x}$ and output $\boldsymbol{y}$ as:

$$E(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w}) = \langle \boldsymbol{w}, \Phi(\boldsymbol{x}, \boldsymbol{y}) \rangle, \tag{2.40}$$

where $\Phi(\boldsymbol{x}, \boldsymbol{y})$ represents a combined feature representation of inputs $\boldsymbol{x} \in \mathcal{X}$ and outputs $\boldsymbol{y} \in \mathcal{Y}$, and $\boldsymbol{w}$ is the parameter vector to be estimated. Since this energy function is linear in $\boldsymbol{w}$, the objectives will remain convex, and can be learned using gradient based methods.

The multi-class logistic regression model of Eq. (2.37), is generalized to the Gibbs distribution[5]. The probability of an output $\boldsymbol{y}$, given an input $\boldsymbol{x}$, is defined by:

$$p(\boldsymbol{y}|\boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x})} \exp\big(E(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w})\big), \tag{2.41}$$

where $Z(\boldsymbol{x})$ is an image-dependent normalizing term known as the *partition function*:

$$Z(\boldsymbol{x}) = \sum_{\boldsymbol{y}' \in \mathcal{Y}} \exp\big(E(\boldsymbol{x}, \boldsymbol{y}', \boldsymbol{w})\big). \tag{2.42}$$

To train the parameter vector $\boldsymbol{w}$, we resort once more on minimizing the negative log-likelihood of the ground truth output $\boldsymbol{y}_i$ for input i, $\ell_{\text{L}}(\boldsymbol{x}_i, \boldsymbol{y}_i, \boldsymbol{w}) = -\log p(\boldsymbol{y}_i|\boldsymbol{x}_i, \boldsymbol{w})$.

For the SVM framework, we use the margin-rescaling structured SVM (Tsochantaridis et al., 2005; Taskar et al., 2003), which generalizes ℓ_{CS} of Eq. (2.39), to:

$$\ell_{\text{MR}}(\boldsymbol{x}_i, \boldsymbol{y}_i, \boldsymbol{w}) = \left[\max_{\boldsymbol{y} \in \mathcal{Y}_{\backslash \boldsymbol{y}_i}} \Big(\Delta(\boldsymbol{y}, \boldsymbol{y}_i) + E(\boldsymbol{x}_i, \boldsymbol{y}, \boldsymbol{w})\Big) - E(\boldsymbol{x}_i, \boldsymbol{y}_i, \boldsymbol{w})\right]_+, \tag{2.43}$$

where $\mathcal{Y}_{\backslash \boldsymbol{y}_i}$ denotes the set of all possible outputs except $\boldsymbol{y}_i$, and $\Delta(\boldsymbol{y}, \boldsymbol{y}_i)$ quantifies the incurred cost for predicting $\boldsymbol{y}$, while the ground-truth output is $\boldsymbol{y}_i$. The zero-one loss, for example, is defined by $\Delta(\boldsymbol{y}_i, \boldsymbol{y}_i) = 0$, and $\Delta(\boldsymbol{y}, \boldsymbol{y}_i) = 1$, for all $\boldsymbol{y} \neq \boldsymbol{y}_i$. Learning this model requires frequent evaluation of the *loss-augmented prediction* step:

$$\max_{\boldsymbol{y} \in \mathcal{Y}_{\backslash \boldsymbol{y}_i}} \left(\Delta(\boldsymbol{y}, \boldsymbol{y}_i) + E(\boldsymbol{x}_i, \boldsymbol{y}, \boldsymbol{w})\right). \tag{2.44}$$

[5] While the Gibbs distribution is often defined using the negative energy, we use the positive energy to make a clear relation to the SVM framework and the multi-class algorithm.

After learning, both models assign the output which highest energy to an input $\boldsymbol{x}$:

$$\boldsymbol{y}^* = \operatorname*{argmax}_{\boldsymbol{y} \in \mathcal{Y}} E(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w}). \tag{2.45}$$

To allow for efficient structured-output estimation, we have to impose constraints on the energy function E, since in itself it contains both:

- Multi-class classification, by setting $E(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w}) = \boldsymbol{w}_{\boldsymbol{y}}^\top \boldsymbol{x}$.
- Independent label classification, by setting $E(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w}) = \sum_{l=1}^{L} y_l \, \boldsymbol{w}_l^\top \boldsymbol{x}$.

An intermediate model encodes some form of structure in the output labels, *e.g.* by modeling pairwise label interactions in a tree-structure. By restricting interaction terms between the labels, inference in these models, *e.g.* computing the partition function or the loss-augmented prediction, can be performed by standard message passing algorithms, such as belief propagation (Bishop, 2006).

The loss-augmented prediction of Eq. (2.44) has, in general, the same complexity of computing the partition function of Eq. (2.42). However, there exist classes of functions for which the max operator can be more efficiently solved than the sum operator. The most notable example is the class of sub-modular functions imposed on the interaction terms. Sub-modular functions are used, for example, in image segmentation, where the output is a binary foreground/background segmentation map, and each pixel is connected to its neighbors. While computing the exact partition function for these models is NP hard, for sub-modular pairwise terms the (loss-augmented) prediction can be efficiently computed.

Structured loss functions Until now we have assumed that the performance measure of interest is the zero/one loss, where only the correct classification leads to a zero loss, and any other output obtains a loss of 1. However this is not always an appropriate loss for the given task. For example in the multi-label case, predicting all but one of the labels correctly, should not obtain the same loss as predicting all labels incorrect. Another example is when the final performance is measured by the average precision over a ranked list of output labels for a specific input.

Using the structured SVM approach, it is relatively straightforward to learn for a specific loss function. Just by defining a suitable cost function Δ, which is imposed to satisfy:

$$\Delta(\boldsymbol{y}, \boldsymbol{y}) = 0 \qquad \text{and} \qquad \Delta(\boldsymbol{y}, \boldsymbol{y}') \geq 0, \text{for } \boldsymbol{y} \neq \boldsymbol{y}'.$$

For learning it is desirable that the loss-augmented prediction of Eq. (2.44) is efficiently computable. Therefore it is often imposed that the cost function and the energy function have the same decomposition over the labels in $\boldsymbol{y}$. Structured SVMs have, for example, been used to optimize document ranking with average precision loss (Yue et al., 2007), and to optimize image classification using a ranking loss (Weston et al., 2011).

In the probabilistic framework a more general loss can be accommodated by, instead of minimizing for the log-loss ℓ_{L}, minimizing the expected cost:

$$\ell_{\mathrm{EC}}(\boldsymbol{x}_i, \boldsymbol{y}_i, \boldsymbol{w}) = \sum_{y \in \mathcal{Y}} \Delta(\boldsymbol{y}, \boldsymbol{y}_i)\; p(\boldsymbol{y}|\boldsymbol{x}_i, \boldsymbol{w}), \tag{2.46}$$

where Δ denotes a suitable cost function.

2.2.2 Non-parametric nearest neighbor classification

In the previous section we have discussed a series of classification models parameterized by a vector $\boldsymbol{w}$. This parameter vector could be obtained from a training set using logistic regression or SVMs. In this section we discuss non-parametric nearest neighbor classification methods, which directly use the training images to classify a new input.

The k-Nearest Neighbor (k-NN) method classifies an input based on a majority vote among the k closest samples from the training set. k-NN is a local learning approach, since the classification function is only approximated among these nearest neighbors. It is also called a *lazy method* since there is no training stage involved, only at test time is the input compared to all samples from the training set. This shows the strong resemblance between k-NN classification and image retrieval models. k-NN classificiation is essentially a retrieval task of finding the k relevant samples for the query input.

While the k-NN algorithm is amongst the simplest of all machine learning algorithms it has some favorable properties.

- It has strong theoretical guarantees, in the limit of infinite data it approaches the Bayes optimal error rate, for a specific value of k (Cover and Hart, 1967).
- It is a non-parametric model, which does not assume any particular distribution on the training data, and where k balances between the smoothness (large k) and the capacity (small k) of the local decision boundary.
- It allows to incorporate new images and new classes by just adding them to the database.

A common choice is to use the Euclidean (ℓ_2) distance to define the nearest neighbors. While the performance of k-NN, using the ℓ_2 distance, is guaranteed to approach the Bayes optimal error in the limit of infinite data (Cover and Hart, 1967), the performance depends critically on the chosen distance function when a finite amount of data is used.

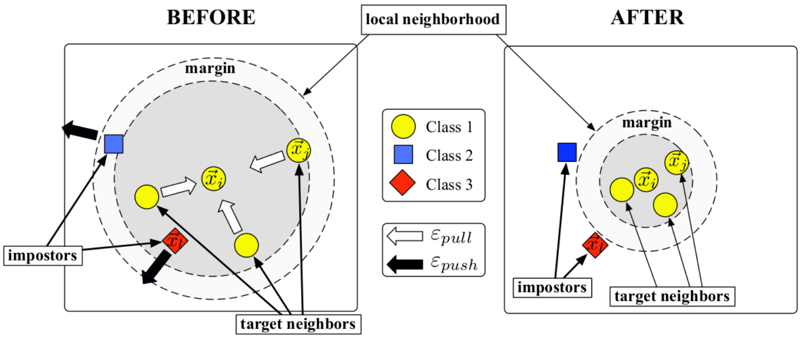


Figure 2.11 *Schematic illustration of LMNN, (*left*) the Euclidean space where target neighbors and impostors are equally close to a query, (*right*) the optimized distance so that the target neighbors lie within a small radius around the query, and the impostors lie outside this radius by some margin. Image courtesy of Weinberger and Saul (2009).*

Metric learning methods aim at learning a task-specific metric to compare two images. There is a large body of literature on metric learning, for different tasks and for different types of inputs, such as images, text documents and DNA-sequences.

In this section we focus on metric learning methods which are designed for image classification problems. We describe two metric learning approaches: LMNN (Weinberger et al., 2006) and TagProp (Guillaumin et al., 2009a). We will use TagProp in Chapter 3 and LMNN in Chapter 5.

Large margin nearest neighbor classification This method, LMNN, is introduced in (Weinberger et al., 2006; Weinberger and Saul, 2009) and learns a Mahalanobis distance optimized for k-NN classification. The Mahalanobis distance between input i and j is given by:

$$d_M(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i - \boldsymbol{x}_j)^\top M(\boldsymbol{x}_i - \boldsymbol{x}_j), \tag{2.47}$$

where M is the $D \times D$ positive semidefinite Mahalanobis matrix, and D the dimensionality of the representation of the input $\boldsymbol{x}_i$.

LMNN tries to ensure that for each image i, a predefined set of target neighbors from the same class are closer than any samples from other classes by a large margin. See Figure 2.11 for an intuition about this method.

The margin criterion of LMNN leads to a convex optimization based on the hinge loss:

$$\ell_{\text{LMNN}}(\boldsymbol{x}_q, P_q, L_q, M) = \sum_{p \in P_q} \sum_{n \in N_q} \left[1 + d_M(\boldsymbol{x}_q, \boldsymbol{x}_p) - d_M(\boldsymbol{x}_q, \boldsymbol{x}_n)\right]_+, \tag{2.48}$$

where P_q is the set of selected target-neighbors for input q, which are pulled towards each other, and N_q is the set of non-target neighbors which are pushed away. For each training input q, its target neighbors P_q and non-target neighbors N_q have to be defined in advance.

TagProp This method, introduced in (Guillaumin et al., 2009a), is a weighted k-NN method designed for multi-label classification. It propagates the labels from the training set, according to a learned weighing function, to a test image.

TagProp uses a probabilistic formulation, where a Bernoulli distribution models the relevance of label $l \in \{1, \ldots, L\}$ for input $\boldsymbol{x}$, which is defined as:

$$p(y_l = +1|\boldsymbol{x}) = \sum_{j \in N} p(y_l = +1|j)\, p(j|\boldsymbol{x}), \tag{2.49}$$

where N denotes the set of all training images, and $p(y_l = -1|\boldsymbol{x}) = 1 - p(y_l = +1|\boldsymbol{x})$. The probability of training sample j predicting label $y_l = +1$, is given by:

$$p(y_l = +1|j) = [\![y_{jl} = +1]\!](1 - 2\epsilon) + \epsilon, \tag{2.50}$$

where $y_{jl} \in \{-1, +1\}$ denotes the absence or presence of label l in the ground-truth annotation of image j, and where ϵ is used to avoid zero prediction probabilities.

The probability of sample j predicting the tags for input $\boldsymbol{x}_i$, is defined as:

$$p(j|\boldsymbol{x}_i) = \frac{\exp -d_{\boldsymbol{\theta}}(\boldsymbol{x}_i, \boldsymbol{x}_j)}{\sum_{j'} \exp -d_{\boldsymbol{\theta}}(\boldsymbol{x}_i, \boldsymbol{x}_{j'})}, \tag{2.51}$$

where $\boldsymbol{\theta}$ is a parameter vector controlling this probability function. This probability can be based on the rank, *e.g.* by $d_{\boldsymbol{\theta}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \theta_k$, where k is the rank of sample j w.r.t. query input i, or by distance, *e.g.* using a weighted combination of base distances between samples i and j, $d_{\boldsymbol{\theta}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{\theta}^\top \boldsymbol{d}_{ij}$, where the distances are collected in vector $\boldsymbol{d}_{ij}$. The performance difference between the rank and distance based formulations is experimentally shown to be small in (Verbeek et al., 2010).

To estimate the parameter vector $\boldsymbol{\theta}$ we maximize the log-likelihood of the predictions of training annotations:

$$\mathcal{L} = \sum_{i,l} \log p(y_{il}|\boldsymbol{x}_i), \tag{2.52}$$

using its gradient:

$$\nabla_\theta\, \mathcal{L} = \sum_i \sum_{j \in N_{\setminus i}} \big(p(j|\boldsymbol{x}_i) - \rho_{ij}\big)\, \nabla_\theta\, d_\theta(\boldsymbol{x}_i, \boldsymbol{x}_j), \tag{2.53}$$

where $N_{\setminus i}$ denotes the set of all training samples, except sample i, and ρ_{ij} denotes the weighted average over all labels l of the posterior probability of neighbor j for image i given the ground-truth label:

$$\rho_{ij} = \sum_l \frac{p(y_{il}|j)p(j|\boldsymbol{x}_i)}{p(y_{il}|\boldsymbol{x}_i)}. \tag{2.54}$$

To reduce the computational cost of training the models, we do not compute all pairwise distances $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$, but rather use a large set of k neighbors for each training sample i. The probability of the remaining samples $p(j|\boldsymbol{x}_i)$ is set to zero. After selecting the k neighbors for each image, TagProp scales linearly with the number of training images.

For a label to receive a high probability, it needs to be present among most of the neighbors of an input with a significant weight. This is, however, unlikely to be the case for rare labels. To boost the probability of rare labels in the annotation vocabulary, word-specific logistic discriminant models could be used:

$$p(y_l = +1|\boldsymbol{x}) = \sigma(\alpha_l \, p_l(\boldsymbol{x}) + \beta_l), \tag{2.55}$$

where $p_l(\boldsymbol{x})$ is the score function for label l given input $\boldsymbol{x}$ as defined by Eq. (2.49). This word-specific sigmoid model adds 2 parameters for each word, and in practice the parameters $\boldsymbol{\theta}$ and $[\alpha_l, \beta_l]_{l=1}^{L}$ are estimated in an alternating fashion.

2.2.3 Benchmark: SVM versus TagProp

In this section we compare SVM and TagProp on the data set of the ImageClef 2010 Visual Concept Detection and Annotation Task (Nowak and Huiskes, 2010). The data set consists of 8,000 training images and 10,000 test images, each image is provided with their Flickr-tags, and manually annotated with 93 concepts. For an illustration of the data set and the different concepts see Figure 1.3. We will also use this data set in Chapter 4, where we will describe it in more detail.

We discuss the results obtained using FV image representations, extracted on densely sampled SIFT and LCS descriptors, which are projected to 64 dimension with PCA. We use 256 visual words, spatial layouts (1×1,2×2, 1×3), and power and ℓ_2 normalization as described in detail in Section 2.1. For both types of local features, the dimensionality of this representation is 256K.

We also consider a multi-modal approach by including a textual feature based on the presence and absence of the ± 700 most frequent Flickr-Tags. We apply ℓ_2 normalization on this binary feature vector, and obtain $\boldsymbol{t}_i$ as a textual feature vector. For TagProp we define textual nearest neighbors based on $\langle \boldsymbol{t}_i, \boldsymbol{t}_j \rangle$.

The performance is measured by using mean Average Precision (MAP), which computes the mean over the average precision of the image ranking per label. We also use its inverse iMAP, which computes the mean over the average precision of the label ranking per image.

For the SVM experiments we have used linear SVMs, using early fusion of features if multiple are used, and cross-validated the weight λ of the regularization term.

	MAP			**iMAP**		
Features	SVM	k-NN	TagProp	SVM	k-NN	TagProp
LCS	**32.8**	29.9	30.8	71.8	71.3	**72.4**
SIFT	**35.5**	33.4	34.5	73.8	72.9	**74.4**
Text	32.6	33.3	**33.8**	65.8	69.0	**71.0**
LCS+SIFT	**38.9**	35.6	36.4	**76.0**	73.4	75.3
Text+LCS+SIFT	**45.5**	42.6	43.7	**77.6**	73.9	77.2

Table 2.2 *Overview of the performance of SVMs and TagProp on the ImageClef'10 data set, using the same image and text representations. For each feature and each evaluation measure the best performing method is indicated in bold.*

For TagProp, we have employed the rank-based version, using $K = 1,000$ nearest neighbors, and applying the word-specific modulations. Since we only have to estimate a limited number of $\pm 1,000$ parameters, we do not use a regularization term. To obtain K nearest neighbors from d different similarity measures, we select from each similarity measure the $K_d = \lceil K/d \rceil$ neighbors.

Results In Table 2.2 we show an overview of the performance obtained by using SVM, k-NN and TagProp for different features and combinations of features. For each feature and each evaluation measure the best performing method is indicated in bold. The SVM and TagProp methods using the combination of Text, SIFT and LCS features, are the two best performing submissions to the ImageClef'10 challenge (Nowak and Huiskes, 2010), for more details see (Mensink et al., 2010a).

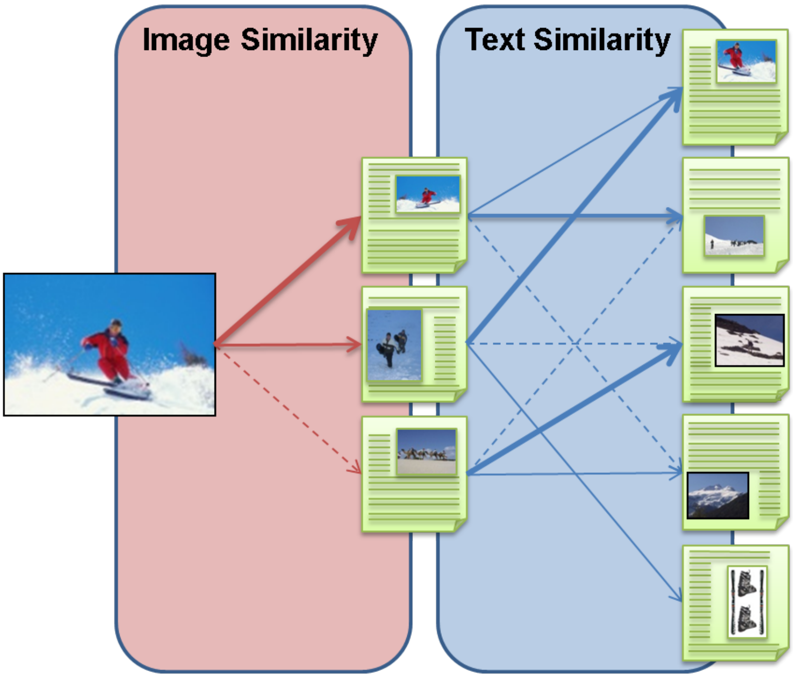
From the results, we first observe that using a combination of several features always improves results, for all three methods, and for both the performance measures. Specifically when using a combination of the heterogeneous visual and textual features, the performance significantly increases.

Second we observe that the performance of the three classification methods is rather similar. Especially when compared to the performance difference of using a single feature versus combining features. Also, when compared to the performance differences of different encodings strategies, as is shown in Section 2.1.2, Table 2.1. However the learned weighting of TagProp performs consistently better than k-NN, most clearly on the iMAP measure.

From these observations we conclude, that both SVMs and TagProp are competitive performing image classification methods. Each having its own advantages, while SVMs outperform TagProp using the combinations of features for this data set, and allow for fast parallel training of accurate classifiers, TagProp learns only a small set of class-independent weights. This allows, for example, for generalizing to new classes, and TagProp is, due to the small number of parameters to learn, less prune to over fitting when more noisy annotations are used (Verbeek et al., 2010).

3

Weighted Transmedia Relevance Feedback for Multi-Modal Retrieval



Retrieval using weighted transmedia relevance feedback. The visual query is used to select a few documents based on their visual similarities. Then, the textual parts of these documents are used as a query to rank all documents.

Contents

In this chapter, we address the goal of exploiting multi-modal image databases, where each image is accompanied by different types of meta-data, such as an image-caption, a set of keywords, or location information. We consider two tasks. First, multi-modal image retrieval, where a query consists of an image with a textual description, and where the goal is to rank the images from the database on their relevance for the query. Second, we consider image auto-annotation, where the goal is to assign the relevant labels to a query image. For this task we use TagProp and therefore propagate the labels from the query's nearest neighbors in the multi-modal data set. The query itself, however, consists of only an image. To take advantage of the multi-modality of a database we use transmedia relevance feedback models. These models assign similarity scores between two documents based on generalized pseudo-relevance feedback models. Parts of this chapter have been published in (Mensink et al., 2010b).

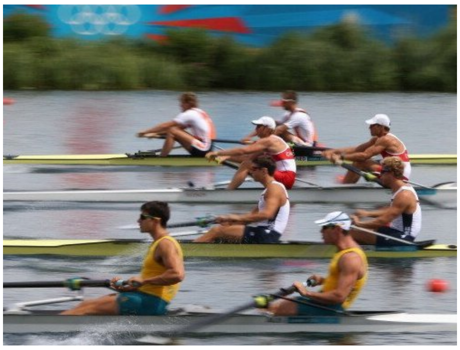
3.1 Introduction

Multi-modal data arise throughout the internet, for example on social media websites showing images and videos with comments, location and keywords (*e.g.* Flickr, Facebook and Youtube) and on news websites publishing articles with photographs (*e.g.* BBC News or Yahoo! News)[1]. The textual meta-data often describes, to some extent, the visual content of the images, although they are not specifically created for this purpose, see the two examples in Figure 3.1.

The nature of these multi-modal databases indicates the need for methods that effectively combine the different modalities present in documents, to access these data using *e.g.* clustering, classification, and retrieval techniques. The great potential of exploiting the relation between the different modalities can be understood by viewing the different modalities in a document as forms of mutual, albeit noisy, supervision. Which, in addition, is

[1] The respective urls are: www.flickr.com, www.facebook.com, www.youtube.com, news.bbc.co.uk, and news.yahoo.com

President Barack Obama greets people on the tarmac after arriving at John F. Kennedy International Airport, Monday, July 30, 2012, in New York.
AP Photo — Jason DeCrow



The Netherlands, United States, Australia and Canada compete in the Men's Pair Heat 2 on Day 1 of the London 2012 Olympic Games at Eton Dorney on July 28, 2012 in Windsor, England.
Photo by Streeter Lecka — Getty Images

Figure 3.1 *Examples of news photographs, illustrating the fact that captions partly describe the visual content of the images. The images are taken from Yahoo! News website.*

usually straightforward to obtain, while it is often expensive to create a large manually annotated database. Recently, this idea has been leveraged to learn recognition models from little or no manual supervision, for example, for face recognition based on image captions (Berg et al., 2004; Guillaumin et al., 2012), accurate object recognition models learned from noisy search engine results (Schroff et al., 2007; Krapac et al., 2010), and action recognition in video using script alignment in combination with text classification to obtain noisy annotation of video clips (Laptev et al., 2008).

Most of the current retrieval systems, however, use only a single modality for retrieval, following one of two paradigms.

- **Query-by-text** is used by most internet image search engines, where a user provides a textual description of the target images, and the text in the meta-data of the images is used to rank the images.
- **Query-by-example** is used for similar image search, where the user provides an example image and visual similarity is used to rank the database images according to their relevance; this approach has recently shown to be successful for image auto-annotation (Makadia et al., 2008; Guillaumin et al., 2009a).

Multi-modal document retrieval differs from this line of work in the sense that it exploits the multi-modal aspect of the database. Also the query might be composed of multiple modalities, *e.g.* a query could consist of text along with an image.

As opposed to classical late or early fusion methods to combine multiple modalities, we use transmedia relevance feedback models. This is an intermediate level fusion method, that models the correlations between the different modalities based on mono-modal similarity measures. It generalizes the pseudo-relevance feedback principle, by changing modality at the query expansion step (Chang and Chen, 2006; Clinchant et al., 2007). For example, for the visual part of a query, visual similarities are used to select the set of most similar documents to this query. In the query expansion step the textual modality of these documents are used to rank all documents in the database. These methods have shown good performance in multi-modal retrieval tasks, *e.g.* at the ImageClef Retrieval evaluations (Müller et al., 2010). These systems are also able to handle documents or queries with only a single modality, and therefore they are also suitable for tasks such as image annotation.

Transmedia relevance models, like pseudo-relevance feedback models, require a few parameters to be set, such as the number of documents used in the query expansion step, and the weighing of the documents retrieved from the expanded query. In this chapter we introduce two parametrizations for transmedia relevance models, and we learn these parameters automatically from training data. The advantage of these learned parametrized models is that we can combine a larger number of retrieval functions.

For multi-modal image retrieval we use a combination of several mono-modal distances and transmedia distances. We compare two models to learn the parameters of the late fusion weights and the transmedia distances from training queries. Each training query has a set of documents labeled by their relevance. Since we encounter a large difference in the distribution of retrieval scores for relevant and non-relevant documents across different training queries, we introduce a method to correct for these inter-query variations in order to learn better parameters.

For image auto-annotation, we extend TagProp, see Section 2.2.2, with the proposed transmedia relevance models. This allows TagProp to exploit the textual relations between documents in the training set, even though the query consists of only an image. Our experiments on multi-modal document retrieval and image auto-annotation, show that our models outperform current state-of-the-art results.

The chapter is organized as follows. In Section 3.2 we provide an overview of related work on multi-modal document retrieval and multi-label image classification. In Section 3.3 we introduce two different parametrizations for transmedia relevance models. In Section 3.4 we present our approaches to learn these parameters on retrieval models, and introduce the query correction terms. In Section 3.5 we describe the extension of TagProp with the transmedia relevance models for image annotation. In Section 3.6 we present experimental results. We present our conclusions in Section 3.7.

3.2 Related work

In this section we discuss the related work for multi-modal image retrieval and image auto-annotation methods which are the most relevant to our work.

Multi-modal document retrieval Learning to rank is an active area of research, which is partly due to its relevance for improving web search engines and due to the availability of large data sets, *e.g.* the LETOR text-retrieval data set (Qin et al., 2010). Most work focuses either on mono-modal text or image retrieval, however the proposed approaches are independent of which modality is used, for an extensive overview see (Li, 2011). Different approaches that have been proposed for retrieval based on keywords include, using a probabilistic cost function (Burges et al., 2005), SVM-based methods (Joachims, 2005; Frome et al., 2007), and online passive-aggressive discriminative learning (Grangier and Bengio, 2008). Recently large-scale (image) data sets are used for query-by-text, see *e.g.* (Checkik et al., 2010; Weston et al., 2010) and for query-by-example image retrieval (Jégou et al., 2010; Perronnin et al., 2010a).

In multi-modal document retrieval the dominant approach is, also, to use only one of the available modalities, either the textual or the visual similarity is used. However, methods that combine different modalities can improve retrieval performance on multi-modal databases, *e.g.* (Müller et al., 2010; Ah-Pine et al., 2010). The methods combining several modalities can be roughly divided into ones that use late fusion or early fusion models.

Late fusion models are attractive due to their simplicity. For each modality a separate retrieval function is defined and the final score for a document is the weighted sum of these scores. The combination might be obtained by a simple averaging of ranking scores from models learned on the different modalities, or by learning an additional fusion model (*e.g.* a linear combination, or a more complex function) that depends on the mono-modal scores. An advantage of these models is that they combine mono-modal similarities for which exist well engineered retrieval systems, and which are well studied; see for a recent overview of text retrieval *e.g.* (Manning et al., 2008), and for content based image retrieval *e.g.* (Datta et al., 2008). Nevertheless, despite the simplicity of late fusion models, they have the disadvantage that they are unable to exploit the correlations between the different modalities, since each modality is treated independently. Furthermore, they can only handle query and document pairs which have the same modality, *i.e.* they can not assign a relevance score to a visual document for a textual query.

Early fusion models attempt to exploit the correlations between the different modalities by finding a joint representation. The joint representation should allow for the heterogeneity of the different modalities, due to variations in their level of semantic meaning (words vs. low level image features), and due to different dimensionalities. An example of early fusion is the concatenation of histograms, which we used in Section 2.2.2 to combine textual and visual data. Another example of early fusion are topic models, which can be used to learn a joint distribution over image features and words, and have been used for image annotation (Barnard et al., 2003).

Different authors have found mixed results when comparing early and late fusion methods for image and video classification tasks (Snoek et al., 2005; Kludas et al., 2007); the best strategy seems to vary across the different classification tasks. Similarly, for multi-modal document retrieval late fusion is the most used approach, however, early fusion techniques are (sometimes) found to perform better depending on the data set and the queries (Depeursinge and Müller, 2010).

Transmedia relevance feedback Intermediate fusion models have been recently introduced to alleviate the drawbacks of early and late fusion, by exploiting the correlations between the different modalities, while using mono-modal similarities. These models are also known as transmedia or cross-media relevance models, since they are based on pseudo-relevance feedback, but in the query expansion step the modalities are switched.

The relevance feedback model was originally developed in the context of text retrieval (Salton and Buckley, 1990). It is a query expansion method where a user selects relevant documents from an initial ranking to extend the query. This extended query contains more relevant keywords, and therefore helps to improve retrieval performance.

Pseudo-relevance feedback models automate relevance feedback; instead of relying on user feedback, the top k retrieved documents are assumed to be relevant and used to enrich the original query. For example, in text retrieval the most frequent words of the top k documents are added to the original query and used to obtain the final document ranking. This often improves retrieval performance since user queries tend to be too short to precisely capture the user intention, while the extended query is likely to contain words related to the original query terms, and therefore a more robust matching is obtained. Pseudo-relevance models have also been successfully used in image retrieval systems (Chum et al., 2007).

These models have been generalized for multi-modal data to transmedia relevance models, where the modality is exchanged in the query expansion step (Chang and Chen, 2006; Maillot et al., 2006; Clinchant et al., 2007; Ah-Pine et al., 2010). For example, in the first retrieval step a visual similarity is used, while for the second step a textual similarity is used, see for a schematic illustration the cover page of this chapter. Transmedia models are useful for databases where at least some of the documents contain both visual and textual data.

The advantage of these models over early fusion approaches is that both the query expansion and the document ranking are based on single modalities, although in combination multiple modalities are exploited. This allows the use of well engineered mono-modal retrieval systems, *e.g.* specifically developed for text or image retrieval. These approaches go beyond late fusion of mono-modal retrieval functions, since they exploit the correspondence between the visual and textual content. These correspondences are implicitly described by the multi-modal representation of the documents in the database.

Another advantage of transmedia models is that they allow the handling of multi-modal and mono-modal queries in a coherent and principled way. For multi-modal queries the different modalities in the data set might bring complementary information. While for mono-modal queries, the multi-modal data set allows to retrieve, for example, pure-text documents that are relevant for a pure-image query. In this manuscript we focus on data sets consisting of images along with a textual representation. The queries are either multi-modal, for the retrieval task, or only visual, for the multi-label classification task.

Image auto-annotation The goal of auto-annotation is to predict relevant keywords from a finite vocabulary for a given image. One solution is to learn a separate binary classifier for each potential keyword. The current state of the art for image classification is to use discriminative classifiers, these could be based on *e.g.* SVMs (Cusano et al., 2004; Sánchez and Perronnin, 2011), boosting methods (Hertz et al., 2004), or ranking losses (Grangier and Bengio, 2008; Weston et al., 2011). However these approaches might become costly when large and dynamic image sets are used, and especially in the case that the data set contains a large number of classes.

An alternative solution is using local learning techniques, where a test image is annotated using the labels from similar images in the training set. Given the increasing amount of training data that is currently available, these techniques become a simple yet powerful alternative to the discussed discriminative models. Local learning techniques have been used, for example for methods based on label diffusion over a similarity graph of labeled and unlabeled images (Liu et al., 2009; Pan et al., 2004), or for learning discriminative models in the neighborhood of test images (Zhang et al., 2006).

Recently, these nearest neighbor based methods have shown excellent performance for image auto-annotation (Makadia et al., 2008; Guillaumin et al., 2009a; Deng et al., 2010). Also, they can outperform discriminative trained classifiers when only noisy training labels are available, such as Flickr tags (Verbeek et al., 2010). In contrast to these models, where the neighbors are found purely on visual information, we integrate transmedia models in TagProp (Guillaumin et al., 2009a, see also Section 2.2.2). Which allows the neighbors of a query image to be defined using also the textual data in the data set.

In the previous chapter we have also discussed image annotation in a multi-modal setting. In Section 2.2.2, we compared the performance of TagProp with SVMs, in a setting where for each training and test image also the Flickr tags were provided. We have observed that using both modalities significantly improved the results. The setting in this chapter is different, since we assume to have a multi-modal training database, but during test time only the image is available as input.

Similarity	Text Query	Image Query
Direct	$s_t(q,d)$	$s_v(q,d)$
Pseudo-relevance	$s_{tt}(q,d)$	$s_{vv}(q,d)$
Transmedia	$s_{tv}(q,d)$	$s_{vt}(q,d)$

Table 3.1 *Overview of possible similarities when using visual and textual modalities.*

3.3 Parametrized transmedia relevance feedback

In this section we introduce our parametrized models for transmedia relevance feedback. Our parametrized models are inspired on the model of Ah-Pine et al. (2008), which defines the transmedia similarity s_{ab} between a query q and a document d as:

$$s_{ab}(q,d) = \sum_{i=1}^{k} s_a(q,d_i)\ s_b(d_i,d), \tag{3.1}$$

where a and b denote the used modalities, s_a and s_b are mono-modal similarities, d_i denotes the i-th most similar document to q according to s_a, and k is the number of query-expansion neighbors. The multiplication of the scores is used to give higher importance to documents that are similar according to both similarities. In our experiments we use visual and textual modalities, an overview of the direct, pseudo-relevance, and transmedia similarities, which we use is given in Table 3.1. For now, we assume that for each modality a similarity measure is available, in Section 3.6 we describe the exact visual and textual similarities we use.

We will now present two parametrized alternatives for Eq. (3.1), which have the advantage that the parameters can be learned from training data for the task at hand.

In the first parametrization, we associate a fixed, rank-based weight with each of the k most similar documents. Using γ_i as the weight for the i-th neighbor, we define:

$$s_{ab}(q,d) = \sum_{i=1}^{k} \gamma_i\ s_a(q,d_i)\ s_b(d_i,d). \tag{3.2}$$

Clearly, this model includes the equal weighting of the first k neighbors (Eq. (3.1)), as a special case when $\gamma_i = 1$ for all neighbors. Since we expect a positive relation between the neighbor similarities and the final similarity, we can impose non-negativity constraints on the coefficients γ_i of the linear combination. Further, since the neighbors are ordered on their distance, we expect that the weight of neighbor $i+1$ should not exceed that of neighbor i, *i.e.* $\gamma_i \geq \gamma_{i+1}$. Both constraints allow for better generalization, and they define a convex set of feasible γ_i values.

The second model we use satisfies the non-negativity and ordering constraints by construction. We use the soft-max function on the first similarity $s_a(q, d')$ to define the following weighting over the neighboring documents:

$$s_{ab}(q, d) = \sum_{i=1}^{k} \tilde{s}_a(q, d_i)\ s_b(d_i, d), \tag{3.3}$$

where

$$\tilde{s}_a(q, d_i) = \frac{\exp(\gamma\ s_a(q, d_i))}{\sum_{j=1}^{k} \exp(\gamma\ s_a(q, d_j))}. \tag{3.4}$$

This model has only a single parameter γ to set, which determines the rate of the exponential decay. The weights assigned to neighbors vary smoothly with the similarity, which might be advantageous. Consider for example two documents d_i and d_j that are very similar to a query q, *e.g.* $s_a(d_i, q) = s_a(d_j, q) + \epsilon$ for a small ϵ. In this example the rank-based weights of the two documents can change substantially depending on γ_i, γ_j, and the sign of ϵ. On the contrary, in the soft-max model the weights of the two documents will remain close as desired.

In the following sections we will use these transmedia relevance similarities for multimodal document retrieval and multi-label image classification.

3.4 Learning score functions for multi-modal retrieval

In the previous section we have introduced parametrized versions of transmedia relevance feedback models. In this section we define score functions to combine several distance functions, which we use to rank the documents of the database for a specific query. We consider two learning objectives to learn the parameters of the retrieval functions, and we introduce query-dependent scaling parameters to allow for better generalization.

We define the retrieval function f as a linear combination of several similarities:

$$f(q, d) = \boldsymbol{w}^\top \boldsymbol{x}_{qd}. \tag{3.5}$$

where $\boldsymbol{x}_{qd}$ denotes the vector of similarities between query q and document d, and $\boldsymbol{w}$ is a parameter vector that controls the late fusion of these similarities. In what follows, we use f_{qd} as a short-hand for $f(q, d)$.

Note that $\boldsymbol{x}_{qd}$ might also be controlled by a set of parameters $\boldsymbol{\gamma}$, depending on the used similarity functions. We learn all parameters $\boldsymbol{\theta} = [\boldsymbol{w}; \boldsymbol{\gamma}]$ using training data which consists of queries q with corresponding lists of relevant and non-relevant documents, denoted R_q and N_q respectively for a query q. In practice, they might be automatically obtained from *e.g.* click-through logs, although the labeling would be noisy in that case. We consider two different classification models to learn these parameters.

Relevance Classification (RC) This model defines the retrieval task as a binary classification problem where query-document pairs have to be classified as either relevant or non-relevant. The RC model has also been explored for text retrieval in Lewis (2001). We use $y_{qd} \in \{-1, +1\}$ to denote the non-relevant or relevant label of a query-document pair, and define the class probabilities using the logistic discriminant model as:

$$p(y_{qd} = +1) = \sigma(f(q,d)) = \sigma(\boldsymbol{w}^\top \boldsymbol{x}_{qd} + w_0), \tag{3.6}$$

where σ is the sigmoid function, and w_0 is a bias term which does not affect the ranking.

The objective is to maximize the log-likelihood of correct classification of all query-document pairs:

$$\begin{aligned} \mathcal{L}_{RC} &= \sum_q \sum_{d \in R_q} \ln p(y_{qd} = \text{+}1) + \sum_{d \in N_q} \ln p(y_{qd} = \text{-}1), \\ &= \sum_q \sum_{d \in (R_q \cup N_q)} \ln \sigma(y_{qd}\, f_{qd}). \end{aligned} \tag{3.7}$$

The objective function is concave in $\boldsymbol{w}$ and w_0, and can be optimized for example using gradient-based methods. The derivative of $\mathcal{L}_{RC}$ w.r.t. θ is given by:

$$\frac{\partial \mathcal{L}_{RC}}{\partial \theta} = -\sum_q \sum_{d \in (R_q \cup N_q)} \sigma(-y_{qd}\, f_{qd}) \frac{\partial f_{qd}}{\partial \theta}. \tag{3.8}$$

Comparative Classification (CC) Learning a classifier to predict document relevance might not be optimal if the goal is to perform ranking on the score function. Instead, we can learn a score function based on pair-wise comparisons, that tries to ensure that each relevant document has a larger score than each non-relevant document. To this end, we define a classification problem over pairs consisting of a relevant document $d \in R_q$ and a non-relevant one $d' \in N_q$. Using the relevance labels y_{qd} and $y_{qd'}$ as before, the goal is to predict which of the two documents is the relevant one and which is the non-relevant one:

$$p(y_{qd} > y_{qd'}) = \sigma(f(d,q) - f(d',q)). \tag{3.9}$$

The objective in this case is to maximize the log-probability of correct classification of all pairs for each query:

$$\mathcal{L}_{CC} = \sum_q \sum_{d \in R_q} \sum_{d' \in N_q} \ln p(y_{qd} > y_{qd'}). \tag{3.10}$$

As before, the model is concave in $\boldsymbol{w}$ and w_0, and can be optimized using gradient-based methods. The derivative of $\mathcal{L}_{CC}$ w.r.t. the parameters θ is given by:

$$\frac{\partial \mathcal{L}_{CC}}{\partial \theta} = -\sum_{q} \sum_{d \in R_q} \sum_{d' \in N_q} \sigma(f_{qd'} - f_{qd}) \left(\frac{\partial f_{qd}}{\partial \theta} - \frac{\partial f_{qd'}}{\partial \theta} \right). \tag{3.11}$$

A similar model has been used in text retrieval using the hinge loss (Joachims, 2005).

In the context of text retrieval, it has been shown that combining these models (RC and CC) could be beneficial, since they encode complementary information (Zheng et al., 2008). However, learning the trade off between the two losses is non-trivial, and requires *e.g.* cross-validation on the used data sets, therefore we do not consider this combination. To the best of our knowledge the two models were not compared before to learn transmedia relevance feedback models.

Correcting for inter-query variations The above objective functions to learn the parameters are defined by summing log-likelihood functions for different queries. In practice we encounter large differences in the distribution of similarities for relevant and non-relevant documents across different queries. Not only does the mean similarity value between the query and the relevant documents changes significantly, also the variance of these similarities varies significantly across queries.

For the ranking performance this does not pose any problem, since ranking is invariant to additive and (positive) multiplicative variations of the scoring function. The objective functions defined above, however, are not invariant to these transformations. The problem is that the objective function optimizes macro-precision —a single cut-off point used for all queries— while we are interested in maximizing micro-precision where a different cut-off point is used per query (Perronnin et al., 2009).

To render the objective functions invariant to additive and multiplicative constants per query, we redefine the retrieval function of Eq. (3.5) as:

$$f(q,d) = \alpha_q(\boldsymbol{w}^\top \boldsymbol{x}_{qd}) + \beta_q, \tag{3.12}$$

where α_q and β_q are free parameters in the optimization. Using Eq. (3.12) as scoring function makes the learning objectives $\mathcal{L}_{RC}$ and $\mathcal{L}_{CC}$ bi-concave, given $\boldsymbol{w}$ they are concave in $\boldsymbol{\alpha}$, and $\boldsymbol{\beta}$, and given $\boldsymbol{\alpha}$ they are concave in $\boldsymbol{w}$ and $\boldsymbol{\beta}$. It is important to note that we do not need α_t and β_t for a test query t, since the ranking is not affected by these multiplicative and additive query specific variables.

The idea of using a correction bias (β_q) was introduced independently from our work for text retrieval in (Ailon, 2009). Ailon calls it the '*intercept*' with the intuition to allow for different relevance criterion for different queries. In our experiments we observe that not

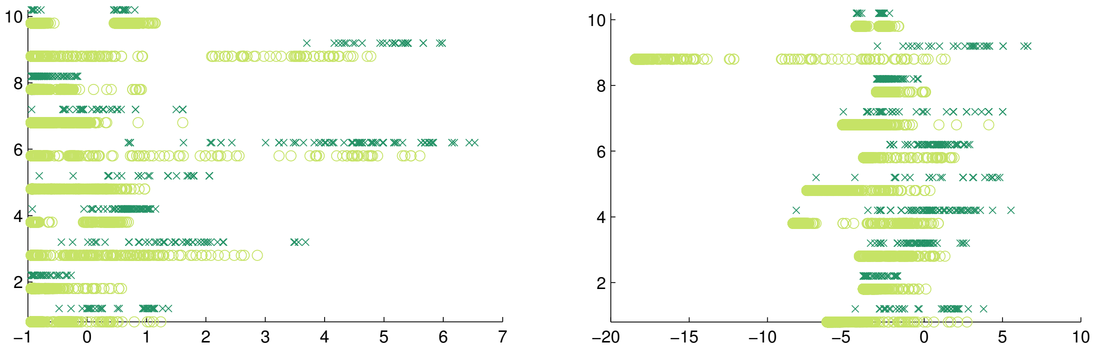


Figure 3.2 *Illustration of the effect of using query dependent correction terms for the RC model. For ten queries (organized vertically) we show the distribution of $f(q, d)$ for relevant documents ($\times$) and non-relevant ones ($\circ$). Without using correction terms we obtain* $\boldsymbol{w} = [3.1, 2.7]^\top$ (left), *while including them yields* $\boldsymbol{w} = [5.1, 9.1]^\top$ (right).

only the bias is query dependent, but also the scaling of the similarity scores, which is corrected for by α_q.

In Figure 3.2 we illustrate the effect of learning the α_q and β_q parameters in practice, in a setting where we combine a textual mono-modal similarity, with an image-to-text transmedia similarity. It shows that without these terms it is difficult to find a single cut-off on the score to discriminate relevant and non-relevant documents. When the correction terms are included, the retrieval scores $f(q, d)$ for relevant and non-relevant documents across different queries become more comparable. Importantly, note that the learned values for late fusion weights $\boldsymbol{w}$ are qualitatively different: without the correction terms more weight is given to the first similarity measure, while the situation is reversed when correction terms are used.

Implementation of the learning algorithms In this section we describe the implementation of the learning algorithms, the gradients used for optimization and how we learn for the parameters of the transmedia models.

The learning algorithm maximizes the objective function ($\mathcal{L}_{RC}$ or $\mathcal{L}_{CC}$), using gradient ascent, over a set of training queries with associated sets of relevant and non-relevant documents. See Algorithm 1 for an overview of the learning procedure.

When using the rank-based formulation of the transmedia feedback models from Eq. (3.2), the transmedia similarity $s_{ab}(q, d)$ is a linear function of γ_i. Therefore we can absorb γ_i into $\boldsymbol{w}$ and the combined distance $s_a(q, d_i)s_b(d_i, d)$ into the vector $\boldsymbol{x}_{qd}$. Lets assume that we combine two similarities, and that $\boldsymbol{w} = [w_a,\ w_{ab}]$ and $\boldsymbol{x}_{qd} = [s_a(q, d),\ s_{ab}(q, d)]$.

```
while not converged do
    maximise $\mathcal{L}$ w.r.t. $\gamma$ (when using soft-max Eq. (3.3));
    maximise $\mathcal{L}$ w.r.t. $\boldsymbol{w}$ (and $\{\beta_q\}$ for $\mathcal{L}_{RC}$);
    maximise $\mathcal{L}$ w.r.t. $\{\alpha_q\}$ (and $\{\beta_q\}$ for $\mathcal{L}_{RC}$);
    calculated log-likelihood with current parameters;
    check for convergence;
end
```

Algorithm 1: Iterative learning of ranking functions for $\theta = \{\boldsymbol{w}, \gamma, \{\alpha_q\}, \{\beta_q\}\}$.

Absorbing the linear transmedia similarity s_{ab} into these vectors, means that $\boldsymbol{w}_{ab}$ and $\boldsymbol{x}_{ab}$ are redefined to:

$$\boldsymbol{w}_{ab} = [\gamma_1, \ldots, \gamma_k], \tag{3.13}$$

$$\boldsymbol{x}_{ab} = [s_a(q, d_1)s_b(d_1, d), \ldots, s_a(q, d_k)s_b(d_k, d)]. \tag{3.14}$$

Thus, we can directly learn a linear weighting of neighbors for transmedia feedback, while also combining several mono-modal or multi-modal similarity measures.

When using the soft-max weighting for transmedia feedback as in Eq. (3.3), we iteratively optimize over γ for fixed $\boldsymbol{w}$, and w_0, and then over $\boldsymbol{w}$ and w_0 for fixed γ. The optimization over γ is not convex, and we use an approximate second order gradient ascent method.

If we include the correction terms α_q and β_q, the learning objective functions remain as before. Except that we will now maximize not only over the generic linear combination $\boldsymbol{w}$, but also over the query specific auxiliary variables $\{\alpha_q, \beta_q\}$. Since the score function is now bi-linear in the parameters, we optimize it in alternation over $\boldsymbol{w}$ and the α_q, which is a convex problem in both cases. The β_q parameters are optimized jointly with both $\boldsymbol{w}$ and the α_q. The bias terms β_q are only used in the RC model, since they cancel out in the CC model.

For the optimization of $\mathcal{L}_{CC}$ and $\mathcal{L}_{RC}$ we use gradient ascent algorithms, and we have given the gradients of these models in Eq. (3.8) and Eq. (3.11) respectively. However, we still need the derivatives of f_{qd} w.r.t. the parameters θ. The derivatives for $\boldsymbol{w}, \alpha_q$, and β_q are trivial to derive. The derivative for γ_{ab} which controls the soft-max of s_{ab}, is given by:

$$\frac{\partial f_{qd}}{\partial \gamma_{ab}} = w_{ab} \sum_q \alpha_q \sum_{i=1}^{k} \tilde{s}_a(q, d_i)s_b(d_i, d) \left(s_a(q, d_i) - \sum_{j=1}^{k} \tilde{s}_a(q, d_j)s_a(q, d_j) \right), \tag{3.15}$$

where w_{ab} denotes the corresponding entry in $\boldsymbol{w}$.

3.5 Image annotation with transmedia distances

In this section we apply the parametrized transmedia relevance models to the problem of image annotation. We extend TagProp to use transmedia relevance models to define nearest neighbors. While the query image to be annotated consists of only visual content, these models allow us to exploit the textual and visual modality of the training set.

We have described TagProp in Section 2.2.2, it is a weighted nearest neighbor approach, to propagate labels from training images to a query image. The probability of label y_l for an image $\boldsymbol{x}_i$ is given by:

$$p(y_l = +1|\boldsymbol{x}_i) = \sum_{j \in N} p(y_l = +1|j)\ p(j|\boldsymbol{x}_i), \tag{3.16}$$

where we define the probability $p(j|\boldsymbol{x}_i)$, based on the distance between i and j as:

$$p(j|\boldsymbol{x}_i) = \frac{\exp(-\boldsymbol{w}^\top \boldsymbol{d}_{ij})}{\sum_{j'} \exp(-\boldsymbol{w}^\top \boldsymbol{d}_{ij'})} \tag{3.17}$$

where the vector $\boldsymbol{w}$ defines the weights for the combination of the different distances between image $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ collected in the vector $\boldsymbol{d}_{ij}$.

To include the idea of transmedia relevance feedback, we define a transmedia distance between the visual query image and the textual concepts of the data set. Inspired by the transmedia scores of Eqs. (3.2-3.3), we explore two visual-to-textual distance measures[2]. We define the linear transmedia distance as:

$$d_{vt}(i,j) = \sum_k \gamma_k\ d_v(i,k)\ d_t(k,j), \tag{3.18}$$

and the soft-max transmedia distance as:

$$\tilde{d}_{vt}(i,j) = \sum_k \tilde{d}_v(i,k)\ d_t(k,j), \tag{3.19}$$

where

$$\tilde{d}_v(i,k) = \frac{\exp(-\gamma\ d_v(i,k))}{\sum_{k'} \exp(-\gamma\ d_v(i,k'))}. \tag{3.20}$$

The difference with the pseudo-relevance models used for image retrieval is that we use distance functions instead of similarity functions. These new transmedia distances can be added to the vector of distances $\boldsymbol{d}_{ij}$ to compute the neighbor weights using Eq. (3.17).

[2] Formally the defined transmedia distances are dissimilarity measures, since they do not necessarily satisfy all conditions of a distance function, *e.g.* the symmetric condition $d(i,j) = d(j,i)$ is not obeyed because the k-NN relation is asymmetric.

```
while not converged do
    minimize log-likelihood $\mathcal{L}$ w.r.t. $\gamma$ parameters
    compute $\tilde{d}_{vt}$ given the $\gamma$ parameters
    minimize $\mathcal{L}$ w.r.t. $\boldsymbol{w}$ using distances $\boldsymbol{d}_{ij}$
    check for convergence of the log-likelihood $\mathcal{L}$.
end
```

Algorithm 2: Optimizing TagProp using soft-max transmedia feedback models.

Learning the parameters of the model Here we describe how to learn the transmedia relevance models using TagProp for image annotation. TagProp maximizes the log-likelihood $\mathcal{L}$ of the predictions of training annotations, using gradient based algorithms. For clarity of presentation we consider a single transmedia component, however extensions to use multiple transmedia components are straightforward.

In the case that we use the linear transmedia distance, Eq. (3.18), each neighbor k from the initial retrieval step can be used as separate distance which is added to $\boldsymbol{d}_{ij}$, and its parameter γ_k is added to the parameter vector $\boldsymbol{w}$. In this way we can use the original TagProp learning algorithm and gradient, just with an extended distance vector per query-image pair.

When we use the soft-max transmedia distance, Eq. (3.19), we optimize iteratively for γ and $\boldsymbol{w}$, as described in Algorithm 2. To compute the gradient for TagProp, given in Eq. (2.53), we need the gradient of the distance function w.r.t. γ_d, which is given by:

$$\nabla_{\gamma_d} \boldsymbol{w}^\top \boldsymbol{d}_{ij} = -w_d \sum_k \tilde{d}_v(i,k) d_t(k,j) \left(d_v(i,k) - \sum_{k'} \tilde{d}_v(i,k') d_v(i,k') \right), \qquad (3.21)$$

where w_d denotes the relevant entry in $\boldsymbol{w}$.

3.6 Experimental results

In this section we describe our experimental evaluation of the proposed transmedia models for retrieval and image annotation. We start with providing details of the data sets, the features and performance measures we have used, then we describe the results for image retrieval, and for image annotation.

3.6.1 Data sets, features, and performance measures

For the experiments we have used two data sets, the IAPR data set (Grubinger et al., 2006), and the Corel-5K data set (Duygulu et al., 2002).

The train station of Oruro in Bolivia
Humans and a dog at a platform with grey stones; a blue and yellow train on the left, the station building on the right.
Oruro - Bolivia

Building; Dog; Stone; Train

Group photo in the citadel of Machu Picchu
Tourists are posing on a green terrace with grey walls and more green terraces behind them.
Machu Picchu - Peru

Terrace; Tourist; Wall

Australian Open - Men's Single Fourth Round
A man in white (Cañas) is hitting a forehand on a green hard court; the net in the background.
Melbourne-Australia

Court; Man; Net

Iguazú Waterfalls
A brown river in the foreground; brown cliffs with dark green trees and numerous waterfalls behind it; a blue sky in the background.
Iguazú - Brazil

Cliff; River; Sky; Tree; Waterfall

Figure 3.3 *Example images from the IAPR data set, for each image its caption, description, location and keywords from a 291 vocabulary are shown.*

The IAPR-TC12 data set[3], has been used for different tasks of the ImageCLEF photo track from 2006 to 2008. The data set contains 20,000 images along with a caption and a textual description, see Figure 3.3 for some examples. We use this data set for our retrieval and annotation experiments.

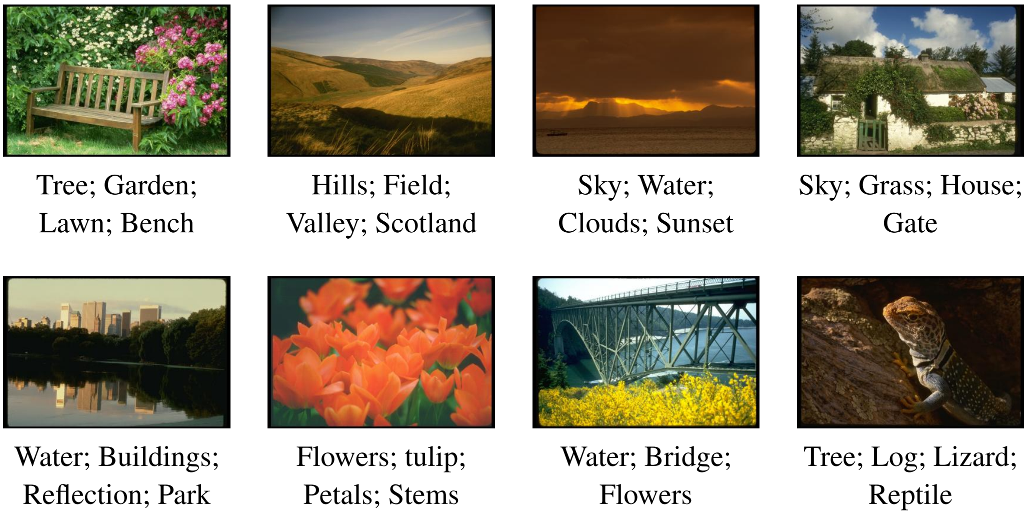
For the retrieval experiments, we use the 60 queries from the ImageCLEF retrieval challenges of 2007 and 2008. Each query consists of a short description and three images, and the goal is to rank the images from the database, some test queries are shown in Figure 3.6. We use the 39 queries from the 2008 challenge as test set, and we use the remaining 21 queries as training set to learn the parameters.

For the annotation experiments, we use the same test and train-split, and the same annotation vocabulary as in (Makadia et al., 2008; Guillaumin et al., 2009a). The vocabulary contains of 291 keywords, and is constructed from the most common nouns of the descriptions obtained using natural language processing techniques.

The Corel 5K data set[4] contains around 5000 images from the Corel CD set, which is a large collection of manually annotated images. Since the introduction, this data set has become an important benchmark for image annotation. The annotations consist of one to five keywords, from a vocabulary of 260 words, and were assigned for the purpose of keyword-based image retrieval. We use this data set only for our annotation experiments, using a fixed set of 499 images as test set, and the rest for training. In Figure 3.4 we show some examples of images with their annotations.

[3] Available at: http://www.imageclef.org/photodata.

[4] Available at http://kobus.ca/research/data/eccv_2002/.



Tree; Garden; Lawn; Bench

Hills; Field; Valley; Scotland

Sky; Water; Clouds; Sunset

Sky; Grass; House; Gate

Water; Buildings; Reflection; Park

Flowers; tulip; Petals; Stems

Water; Bridge; Flowers

Tree; Log; Lizard; Reptile

Figure 3.4 *Example images with their relevant labels from the Corel-5K data set.*

Visual and Textual Features In this section we describe the visual and textual features we use for our experiments. We extract different visual features for the retrieval and annotation experiments, in order to compare the proposed models to the different baseline approaches.

For the image retrieval experiments, we use the visual features described in (Ah-Pine et al., 2008). We extract Fisher Vectors on local SIFT and LCS descriptors, and concatenate them in an early fusion style to obtain a single image signature. The similarity between two images d and d' is defined as 2 minus the negative ℓ_1 distance:

$$s_v(d, d') = 2 - \sum_k |x_d^{(k)} - x_{d'}^{(k)}|. \tag{3.22}$$

For the image annotation experiments we use the same features and distances as used in (Guillaumin et al., 2009a)[5]. These consist of a collection of 15 visual features including, the Gist descriptor (Oliva and Torralba, 2001), color histograms for RGB, LAB and HSV representations, and BoV histograms computed from quantized SIFT and color descriptors. To compute the visual distances from the descriptors we use ℓ_2 as the base metric for Gist, ℓ_1 for color histograms, and χ^2 for the BoV histograms. In general we use these visual features equally weighted and averaged, which we refer to as the Joint Equal Contribution (JEC) distance. It has been shown that the JEC distance outperforms any of the individual base distances (Guillaumin, 2010). Unless specified otherwise, we use JEC as the single visual distance in our annotation experiments.

For the textual similarities, we use two different textual representations, one based on the (extracted) keywords, and the other on the textual descriptions, the latter is only available for the IAPR data set.

[5] Available at http://lear.inrialpes.fr/data.

For the representation of the textual descriptions, we use a probabilistic language modeling approach on pre-processed texts (Ponte and Croft, 1998). The pre-processing includes tokenization, lemmatization, word de-compounding and a standard stop-word removal. The word counts associated with each document are obtained by adding counts from the caption, the location, and the description fields of each document. As a similarity measure between documents d and d' we use the cross-entropy:

$$s_t(d, d') = \sum_w p(w|d) \log p(w|d'). \tag{3.23}$$

To use this similarity in TagProp, we define $d_t(d, d') = 1 - s_t(d, d')$.

The second textual similarity is based on the annotation keywords associated with each image. We define the tag-distance as the intersection-over-union measure over the ground truth annotations of documents d and d':

$$d_t(d, d') = 1 - |Y_d \cap Y_{d'}|/|Y_d \cup Y_{d'}|, \tag{3.24}$$

where $Y_d = \{t|y_{dt} = +1\}$ is the set of relevant keywords for document d. We use this distance only during the annotation experiments.

In this chapter we use the described mono-modal similarities to investigate the influence of learning transmedia similarities. We have chosen these similarities to be comparable to our baseline methods. However the obtained results might be improved when the mono-modal similarities are also parametrized, and the parameters learned for the task at hand.

Performance Measures The performance of the image retrieval system is measured by evaluating the different test-queries and calculating the mean average precision (MAP) and the mean precision-at-k with $k = 20$ (P@20). MAP is obtained by computing for each query the average of the precisions measured after each relevant image is retrieved, and P@20 is obtained by measuring the precision of the top 20 retrieved images.

Image auto-annotation is usually evaluated measuring the keyword based retrieval of the system, for this we also use MAP, and break-even point precision (BEP) over keywords. BEP (or R-precision) measures for each keyword (tag) t the precision among the top n_t images, where n_t is the number of images annotated with this keyword in the ground truth. In order to evaluate the performance of an annotation for a given image, we inverse these measures, and calculate iMAP and iBEP, following (Verbeek et al., 2010). These measures calculate precision over ranked keywords, and average over all images, instead of calculating precision over ranked images and averaging over keywords.

		MAP	P@20
Text	s_t	20.5	27.4
Image	s_v	14.0	29.4
Text-to-image	s_{tv}	13.3	24.4
Image-to-text	s_{vt}	33.5	50.9
Baseline	$s_t + 2\ s_{vt}$	40.1	56.4

Table 3.2 *Baseline performance, shown for the two mono-modal and transmedia similarities, and for the baseline approach.*

3.6.2 Multi-modal image retrieval

In this section we evaluate image retrieval on the IAPR data set. We explore the different methods for learning the parameters, the different versions of transmedia relevance feedback, and compare to the state-of-the-art on this data set.

For efficiency reasons, for each document we only store the similarities to the most similar documents, and set the others to zero. For both the similarities between queries and the database, and for the similarities between documents in the database we keep only the 200 most similar documents. Except for our baseline, where we follow the approach of Ah-Pine et al. (2008), and use the 1,000 most similar documents for the similarity between queries and the database.

Baseline method As a baseline method for our retrieval experiments we use the approach described in Ah-Pine et al. (2008), which was the winning entry of the 2008 ImageClef Retrieval challenge (Arni et al., 2008). This method combines the direct textual similarity based on the language model, Eq. (3.23), with the image-to-text transmedia distance. The transmedia relevance feedback uses equal weighing, *c.f*. Eq. (3.1), with $k = 2$ neighbors, and the late fusion weights are used to give more weight to the image-to-text distance, and are defined as $\boldsymbol{w} = [1, 2]^\top$. We will refer to this method as the baseline.

In Table 3.2, we show the performance of the different direct similarity measures, the transmedia similarities, and the combination used in the baseline method. For the transmedia similarities we use equal weighting of the $k = 2$ query-expansion neighbors. We observe that the image-to-text transmedia similarity outperforms any of the other similarities, but combined with the text similarity obtains a significantly higher performance.

Retrieval models with query correction terms In our first set of experiments we compare the objective functions RC and CC to learn the late fusion weights. For these experiments we use the weight vector $\boldsymbol{w}$ to combine the textual similarity s_t and the transmedia similarity s_{vt}. The transmedia similarity uses equal weighting of each query expansion neighbor, *c.f*. Eq. (3.1), and we use different values for k.

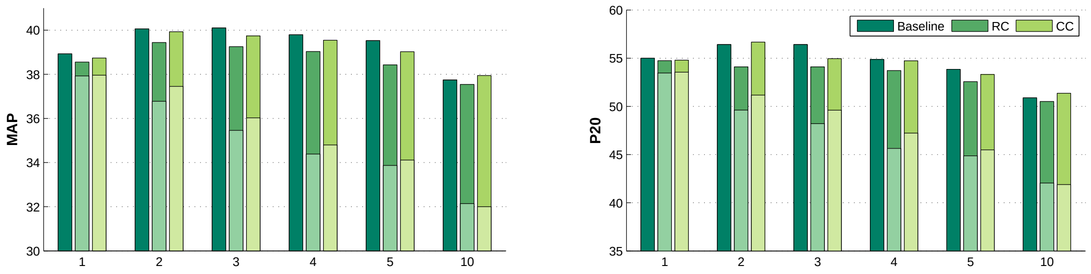


Figure 3.5 *Performance in MAP (left) and P@20 (right) of the baseline and our models, for several values of k. For the learned classification models, we show the results without (lighter bars) and with (darker bars) the query specific correction terms $\{\alpha_q, \beta_q\}$.*

In Figure 3.5 we show the MAP and P@20 scores obtained for the different models, and compare them to the baseline approach using $\boldsymbol{w} = [1,2]^\top$. We observe that for both performance measures, for all values of k, and for both learning objectives, the performance is significantly improved when applying the query correction terms $\{\alpha_q, \beta_q\}$.

Similarly, we observe for all values of k, both with and without the query correction terms, and using both performance measures, the CC objective performs better than, or comparable to, the RC objective. Also, the CC model performs comparably with the baseline system that uses a manually tuned weight vector.

Best results are obtained when using a relatively low value for k; depending on the performance measure and the method choosing either $k = 2$ or $k = 3$ is optimal. This is in line with the results obtained in (Ah-Pine et al., 2010). In the following experiments we evaluate our learned transmedia models to assess whether there is truly nothing to be gained by using larger values of k, or whether the equal weighting of the neighbors is hindering performance.

Learned transmedia relevance models In our second set of experiments we evaluate the transmedia weighting schemes: equal weighting, using neighbor ranks, and using the soft-max function, *c.f*. Eqs. (3.1-3.3). For the rank-based weights we also evaluate the effect of imposing positivity and ordering constraints. We only present results obtained with the CC model including the correction terms, since it was outperforming the RC model in the earlier experiments. The results are summarized in Table 3.3.

We see that, for a larger value of k our parametrized models substantially improve over the equal weighting of query expansion neighbors. For increasing values of k the performance of the equal weighting scheme decreases, while that of our learned soft-max weighting improves and leads to the best overall results.

It is interesting to observe that for the rank-based weighting it is important to impose stricter constraints to improve the performance. In particular for larger values of k, where a larger number of parameters needs to be estimated with a higher risk of overfitting.

	Eq. Weighed		Unconstr.		Positive		Pos Ord		Softmax	
k	MAP	P@20	MAP	P@20	MAP	P@20	MAP	P@20	MAP	P@20
2	**39.9**	**56.7**	38.7	54.7	38.8	54.8	**39.9**	56.6	**39.9**	55.8
5	39.0	53.3	38.9	54.2	39.2	54.2	40.0	55.7	**41.6**	**58.8**
10	38.0	51.4	38.5	54.5	39.9	55.3	40.1	56.1	**42.1**	**59.3**
25	36.2	48.4	35.7	52.1	37.4	52.4	40.0	55.1	**42.1**	**58.3**
50	34.3	47.0	30.4	43.9	32.9	46.4	39.0	53.4	**42.7**	**59.7**

Table 3.3 *Results for different definitions of the transmedia component s_{vt}, combined with s_t by late fusion, using the $\mathcal{L}_{CC}$ model. For each value of k the best results are highlighted bold.*

The soft-max model, does not suffer from this problem, since it only requires a single parameter γ to be learned. The soft-max model outperforms any of the rank-based models for $k > 2$.

In Figure 3.6 we illustrate the performance of the CC model for six queries, learned with query correction terms, and soft-max weighting (with $k = 50$) in the transmedia component.

Combining all six similarities For the experiments so far we have combined only two similarities, the direct text similarity with the image-to-text similarity. Considering a combination of more similarities, *e.g.* all the six similarities which are possible between the visual and textual modalities for this retrieval task, see Table 3.1, makes the manual parameter tuning much more cumbersome: there are six late fusion weights to set, and four values of k to set for the indirect similarities.

In a preliminary experiment we have used all six components in an equal weighted late fusion ranking function. Where we have defined the pseudo-relevance and transmedia similarities using equal weighting, Eq. (3.1), and all use the same value for k the number of query expansion neighbors. For any of the tested value of k this did not lead to any improvement over the baseline of combining just the direct text similarity with the image-to-text similarity, using $\boldsymbol{w} = [1, 2]^\top$.

On the contrary, learning the weight parameters with the CC model, and using soft-max transmedia components, allowed for improvements over the learned two component models, and this for any choice we made for k. For example, for $k = 50$ we obtain an MAP value of 43.1% and 59.9% in P@20, see Table 3.4. Upon inspection we find that the learned weight for both pseudo-relevance components s_{tt} and s_{vv} equals zero, explaining partially that the improvement over the model using only two components is only moderate. For this model, we have to learn 6 late fusion weights $\boldsymbol{w}$ and 4 values of γ for the relevance feedback models, plus two parameters (α_q, β_q) per query.

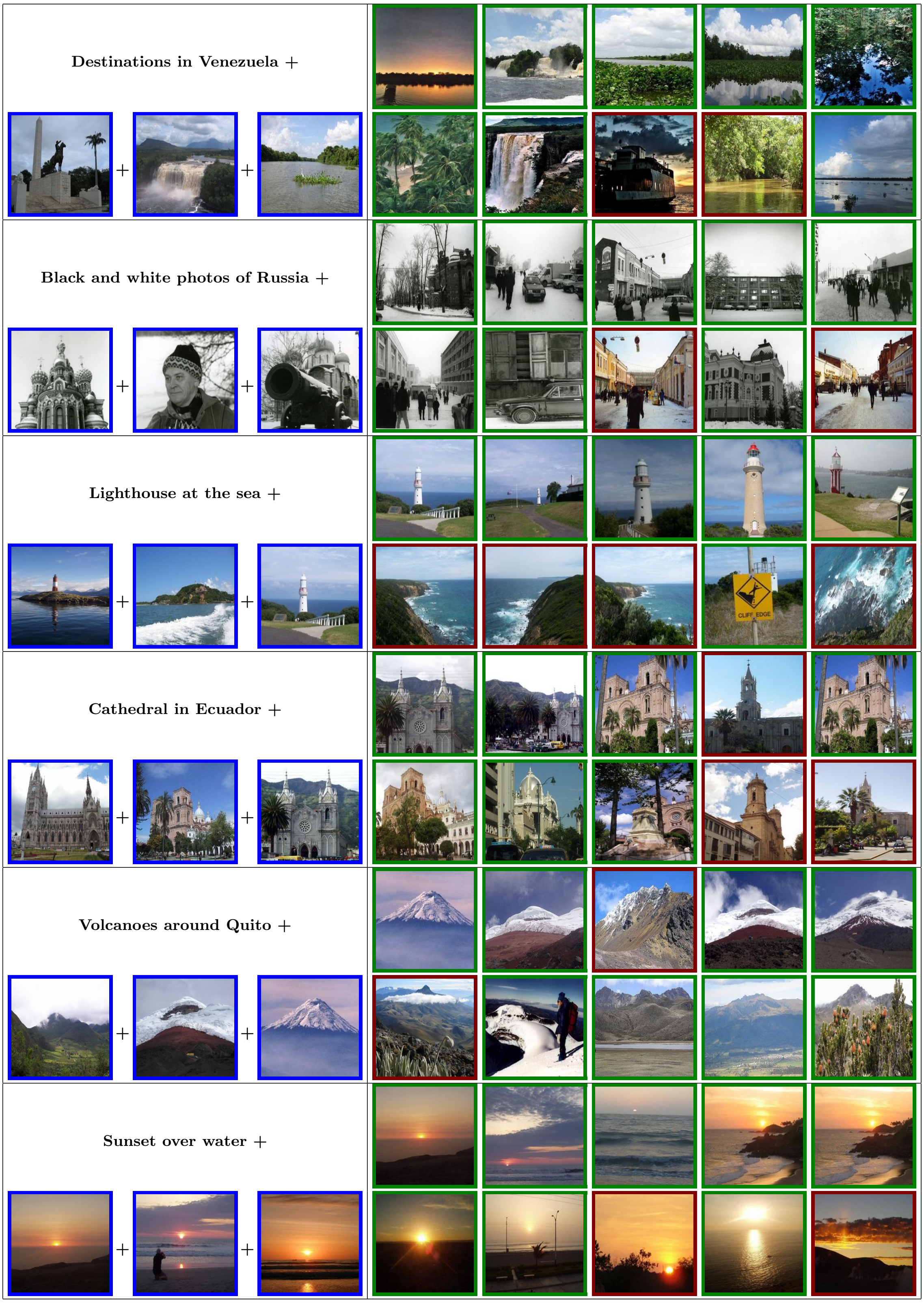


Figure 3.6 *Example queries from the ImageCLEF Photo Retrieval Task: on the left we show the textual query and the three query images (with a blue box), on the right we show our top 10 results. Relevant images are denoted with a green box, while non-relevant images have a red box.*

Method	MAP	P@20
AVEIR (Tollari et al., 2008)	31.8	43.5
UP-GPLSI (Navarro et al., 2008)	33.0	43.1
DCU (O'Hare et al., 2008)	35.1	47.6
XRCE (Ah-Pine et al., 2008)	41.1	57.3
(our implementation)	40.1	56.4
Ours - 2 comp	42.7	59.7
Ours - 6 comp	**43.1**	**59.9**

Table 3.4 *Overview of the best results obtained by participants to the ImageCLEF 2008 Photo Retrieval task, and the proposed CC models using* $k = 50$ *expansion neighbors.*

Comparison to ImageCLEF 2008 participants Finally, in Table 3.4, we compare our results to the best submissions of the ImageCLEF 2008 Photo Retrieval task. Our baseline method is the re-implementation of (Ah-Pine et al., 2008), the winners of the challenge (Arni et al., 2008). However, using the same features, their weighting $\boldsymbol{w} = [1, 2]^\top$, and $k = 2$, we obtain somewhat lower results than the ones reported by ImageCLEF[6]. However using these features and our learning approach that integrates all six components we obtain an improvement of over 3% in both MAP and P@20.

3.6.3 Image annotation experiments

In this section we evaluate our transmedia relevance feedback models for image annotation on the Corel-5K and IAPR data sets. In our experiments we use distance based TagProp, using the JEC distance as a baseline method, we consider using $J = \{200, 400, 1000\}$ neighbors. The linear transmedia model based on neighbor ranks, Eq. (3.18), is denoted as LTP, and the soft-max model, Eq. (3.19), is denoted as STP.

Annotation with transmedia relevance models In our first set of experiments we compare our proposed transmedia relevance models to the baseline on the Corel-5K and IAPR data sets. For the transmedia distance d_{vt} we use the image-to-tag distance, using the intersection-over-union between tags as the textual distance, defined in Eq. (3.24). In Figure 3.7 we detail the MAP and iMAP performance.

The results on the Corel-5K data set (Figure 3.7, top row) show that for most parameter configurations and for both performance measures the transmedia methods outperform the baseline. When comparing the best scoring TagProp (using J=200) with our method (using LTP, K=15, J=1000), we observe that the performance increases from 36.2% to

[6] After several conversations with the authors it is still not exactly clear where the difference in performance might come from, perhaps there is a difference in the used features or the implementation of the feature extraction algorithms.

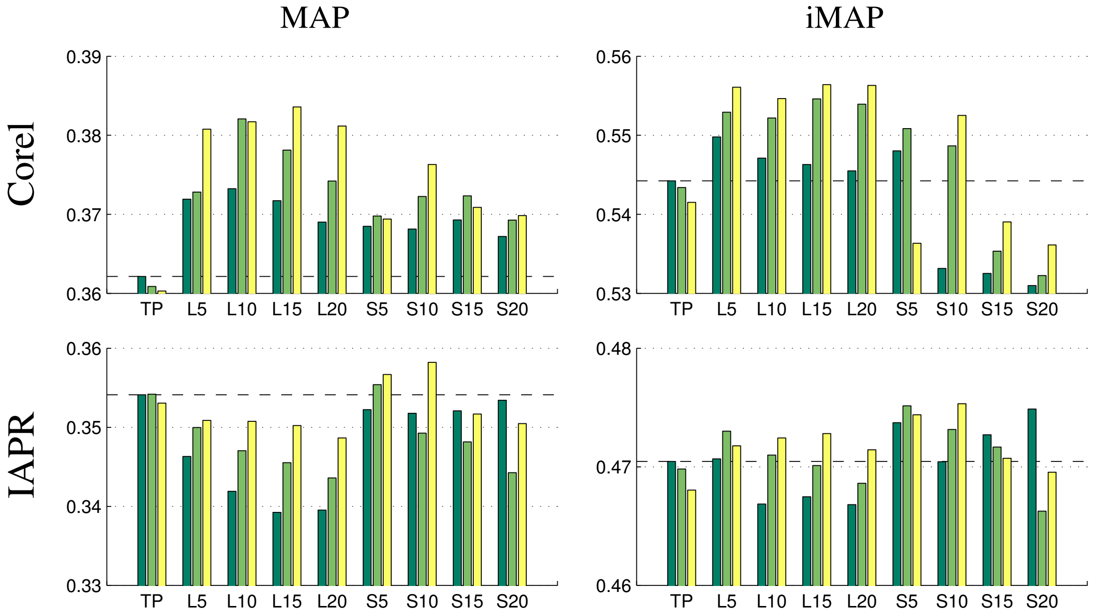


Figure 3.7 *Image annotation performance on Corel 5K and IAPR data sets using TagProp (TP), and transmedia models using the image-to-tag distance. Usage of LTP and STP is indicated by L and S respectively, followed the number of k of the query-expansion neighbors. Results for three different sizes of neighborhoods in TagProp are shown in adjacent bars, $J = \{200, 400, 1000\}$.*

38.4% in MAP. Furthermore, the figures show that LTP generally outperforms STP on this data set. Finally, we observe that if the neighborhood size J is increased the performances of LTP and STP increase as well, while for TagProp the performance slightly decreases.

The results on the IAPR data set (Figure 3.7, bottom row) show that the transmedia relevance feedback models for this database do not obtain much higher performance than the baseline TagProp. Another difference compared to the Corel data set is that on this data set the STP model seems to perform better than the LTP model.

Comparing pseudo-relevance and transmedia relevance models In our second set of experiments we further compare the performance of the LTP and STP with different choices for the distance d_2 that is used in the second step of the query expansion models. We use (a) the image-to-image distance based on JEC, generating a visual pseudo-relevance feedback model, (b) the image-to-tag distance, (c) the image-to-text distance, based on the cross-entropy text distance (for the IAPR data set), and (d) combinations of the previous distances. The results are given in Table 3.5.

On the Corel-5K data set, using visual pseudo-relevance feedback performs similarly as the baseline TagProp, the transmedia model (using the tag-distance) clearly improves the retrieval and annotation performance, and the combination of the two transmedia dis-

		LTP				STP			
		MAP	BEP	iMAP	iBEP	MAP	BEP	iMAP	iBEP
Corel 5K	TagProp	36.0	32.5	54.2	47.6				
	$d_2 = \{\text{Jec}\}$	36.0	32.5	54.2	47.8	36.0	32.5	54.2	47.8
	$d_2 = \{\text{Tag}\}$	**38.1**	33.8	**55.6**	49.3	37.0	33.1	53.6	47.0
	$d_2 = \{\text{Jec, Tag}\}$	37.9	**33.9**	55.5	**49.7**	36.6	32.9	53.7	47.2

		LTP				STP			
		MAP	BEP	iMAP	iBEP	MAP	BEP	iMAP	iBEP
IAPR TC 12	TagProp	35.4	36.0	47.0	42.6				
	$d_2 = \{\text{Jec}\}$	35.1	36.0	46.7	42.2	35.1	36.0	46.7	42.3
	$d_2 = \{\text{Tag}\}$	34.7	35.5	47.1	42.3	35.6	36.3	47.4	42.7
	$d_2 = \{\text{Text}\}$	34.9	35.9	47.5	42.2	**35.9**	36.3	**48.0**	42.8
	$d_2 = \{\text{Jec,Tag}\}$	34.5	35.5	46.8	41.9	35.3	36.3	47.1	42.0
	$d_2 = \{\text{Jec,Text}\}$	34.5	35.7	47.1	42.0	35.5	36.3	47.5	42.8
	$d_2 = \{\text{Tag, Text}\}$	34.7	35.8	47.2	42.1	35.7	**36.5**	47.9	**43.0**

Table 3.5 *Performance of different distances d_2 in the second step of pseudo-relevance and transmedia relevance models, using $J = 1000$ (TagProp neighbors), and $K = 20$ (query expansion neighbors). For both data sets the best performing model per evaluation measure is marked bold.*

tances performs comparable to using just the transmedia model. Just as in Figure 3.7, LTP seems to outperform STP for this database and these settings.

On the IAPR data set, we observe that the performance differences are smaller, and that the STP models perform slightly better than the LTP models. Further we observe that the visual pseudo-relevance model does not increase the performance. The best performance is obtained by using d_2 = Text or d_2 ={Tag, Text}, improving between $.5 - 1\%$ on the different performance measures compared to the baseline.

Learning visual distance weights In our final set of experiments, we learn a combination of visual distances, instead of using the visual JEC distance. We have selected the four visual distances that obtain the highest weights when learning TagProp with the 15 individual feature distances according to (Verbeek et al., 2010). The selected features are the Gist feature, and SIFT features extracted on a dense grid, on Harris key points, and on a dense grid in a 1×3 spatial layout. We define transmedia relevance models using each of them, and thus we combine 4 mono-modal distances and per expanded modality 4 transmedia distances.

In Table 3.6 we compare the results on the IAPR data set of TagProp, LTP and STP. In contrast to the experiments when using only the JEC distance, in this case the LTP models performs slightly better than the STP models to learn a combination of visual distances.

		LTP				STP			
		MAP	BEP	iMAP	iBEP	MAP	BEP	iMAP	iBEP
IAPR TC 12	TagProp	35.7	36.1	49.0	44.1				
	d_2 ={Jec}	35.0	35.3	48.6	44.1	35.0	35.6	48.6	44.0
	d_2 ={Tag}	36.0	**36.7**	49.6	44.6	35.6	36.1	49.2	44.4
	d_2 ={Text}	**36.4**	**36.7**	49.6	44.3	35.7	35.7	49.5	44.2
	d_2 ={Tag, Text}	36.2	36.6	**49.9**	**44.8**	35.8	36.6	49.8	44.6

Table 3.6 *Performance when combining four visual distances with TagProp and its transmedia extensions, using $J = 400$ (TagProp neighbors), and $K = 20$ (query expansion neighbors). The best performing model per evaluation measure is marked bold.*

Using LTP models with both the image-to-tag and the image-to-text transmedia distances we obtain modest improvements on all performance measures.

3.7 Conclusion

In this chapter we have proposed two parameterizations for transmedia relevance feedback models, that generalize the models proposed in (Ah-Pine et al., 2008). Our models extend the latter by incorporating a weighting among the neighbors used to compute the transmedia relevance feedback score.

For image retrieval, we have explored two learning objectives to learn the late fusion weights and the parameters of the pseudo-relevance feedback components from data. We have introduced multiplicative and additive correction terms to learn multi-modal retrieval score functions, to allow for inter-query differences in the distribution of similarity values. The motivation for this is that while ranking performance is invariant to such terms, the objective functions of the learning algorithms are not.

Our experimental results show that learning the parameters of the transmedia components, and the parameters of the late fusion is beneficial. On the ImageCLEF evaluation data set the learned models improve over 3% in MAP and P@20 compared to our baseline method.

For image annotation, we integrated the transmedia relevance feedback components as distances in TagProp. This allows to define neighbors of a query image based on the textual and visual modalities in the data set.

Our experimental results show that on both data sets used in our evaluation, the Corel-5K data set and the IAPR data set, incorporating the transmedia distances leads to higher annotation accuracy. These results indicate that using the textual similarity among images in the training set can improve auto-annotation, even for test images for which only visual information is available.

4

Structured Models for Interactive Image Labeling and Attribute Based Classification

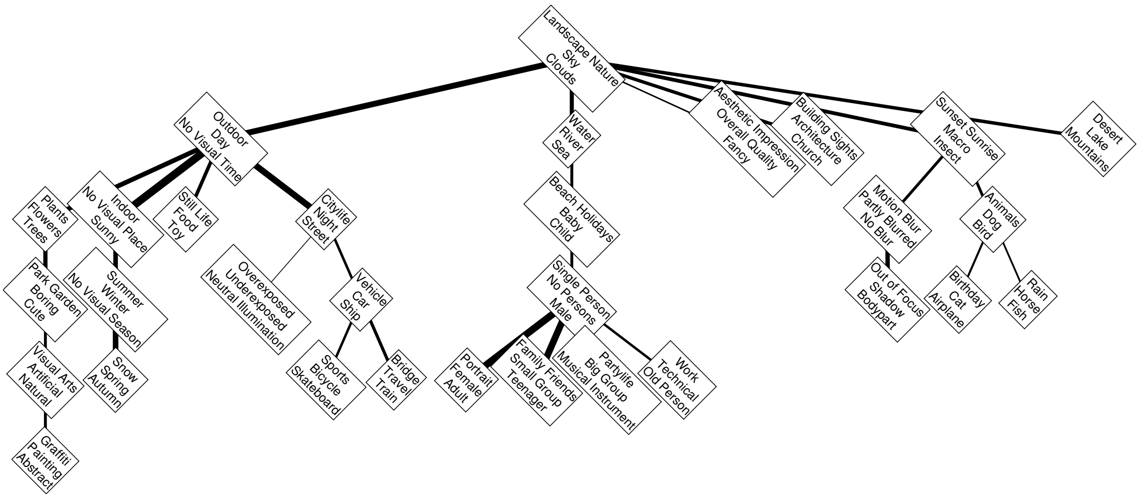

Landscape Nature Sky Clouds
Outdoor Day No Visual Time
Water River Sea
Aesthetic Impression Overall Quality Fancy
Building Sights Architecture Church
Sunset Sunrise Macro Insect
Desert Lake Mountains
Plants Flowers Trees
Indoor No Visual Place Sunny
Still Life Food Toy
Citylife Night Street
Beach Holidays Baby Child
Motion Blur Partly Blurred No Blur
Animals Dog Bird
Park Garden Boring Cute
Summer Winter No Visual Season
Overexposed Underexposed Neutral Illumination
Vehicle Car Ship
Single Person No Persons Male
Out of Focus Shadow Bodypart
Birthday Cat Airplane
Rain Horse Fish
Visual Arts Artificial Natural
Snow Spring Autumn
Sports Bicycle Skateboard
Bridge Travel Train
Portrait Female Adult
Family Friends Small Group Teenager
Partylife Big Group Musical Instrument
Work Technical Old Person
Graffiti Painting Abstract

Illustration of a tree-structure defined over groups of 3 labels, obtained from the *ImageClef'10* data set. While not enforced, semantically related labels are often grouped together.

Contents

In this chapter we address several goals related to the tasks of image labeling and classification. We introduce two different types of structured prediction models, which both explicitly take into account the dependencies among the output labels. These models are then used to leverage user input in an interactive prediction setting, to classify unseen classes using attribute based classification, and finally to learn for a specific ranking measure. This chapter is based on (Mensink et al., 2011a,b, 2012b).

4.1 Introduction

In this chapter we address the problem of image labeling, where the goal is to predict the relevant labels from a given annotation vocabulary for an image. The predicted labels can be used for *e.g.* clustering, retrieval, and attribute-based classification. Most existing systems address the problem of image annotation either in a fully manual way (*e.g.* stock photo sites as Getty images[1]), or in a fully automatic setting where image labels are automatically predicted. In the latter case, most commonly used are independent label predictors based on either classifiers, *e.g.* (Zhang et al., 2007), ranking models, *e.g.* (Grangier and Bengio, 2008), or nearest neighbor predictors such as TagProp, see Section 2.2.2.

We differentiate from this predominant line of work in two ways. First, we propose structured models that take into account the dependencies among the image labels explicitly. Since these models are more expressive, they lead to more accurate image label predictions. Second, we follow an interactive labeling scenario, where a user is asked to confirm or reject, *at test time*, some of the image labels. Such an interactive scenario is, for example, useful when indexing images for stock photography, where a high indexing quality is mandatory, yet fully manual indexing is very expensive and suffers from very low throughput.

[1] Website at: http://www.gettyimages.com

ImageCLEF 10 - 12 labels	Before	Questions	After
	No Vis. Seas.		Indoor
	Neutr Illum.		Female
	No Blur		Adult
	No Pers.	Day	Male
	Day	No Pers.	No Vis. Seas.
	Natural	Indoor	No Vis. Time
	No Vis. Time	Adult	Neutr Illum.
	Outdoor	Female	No Blur
	Cute		Single Person
	Visual Arts		Natural

AwA - 29 attributes	Before	Questions	After
	Fast		Toughskin
	Active		Swims
	Smart		Arctic
	Meatteeth	Toughskin	Water
	Newworld	Paws	Fish
	Agility	Swims	Ocean
	Tail	Mountains	Fast
	Meat	Arctic	Active
	Strong		Strong
	Chewteeth		Smart

Figure 4.1 *Illustration of the effect of interactive image annotation. We show the labels with highest confidence before and after user input (green labels are correct, red ones not), as well as the five labels selected by the system to be set by the user (blue). Example images are from the ImageCLEF data set (*top*), and from the AwA data set (*bottom*).*

The interactive scenario offers an interesting trade-off between accuracy and manual labeling effort. In this case the label dependencies in the proposed models can be leveraged in two ways. First, the structured models are able to transfer the user input for one label to more accurate predictions on other image labels, which is impossible with independent prediction models. Second, using structured models, the system will not query, wastefully, for image labels that are either highly dependent on already provided labels, or predicted with high certainty from the image content. Through inference in the graphical model, the system fuses the information from the image content and the user responses, and is able to identify labels that are highly informative once provided by the user. Experimentally we show that a small amount of user input substantially improves the performance.

In addition to showing the effectiveness of structured models for interactive image labeling, we also explore how the proposed structured models can be exploited in the context of attribute-based image classification (Lampert et al., 2009; Branson et al., 2010). The attributes are shared between different classes and image classification is based on a given attribute-to-class mapping. Hence, attribute values are first predicted for the image and then the attribute-to-class mapping is used to obtain the class probabilities. Predicting the attribute values for an image can be seen as annotating an image with a set of (attribute) labels, therefore we use our structured models at the attribute level. The user interaction will also take place at the attribute level, but in this case the system will ask attribute labels to improve the class predictions rather than the attribute predictions. Experiments on the AwA data set show that, also in this case, the structured models outperform the independent attribute prediction model, both in automatic and interactive scenarios.

The performance of image annotation and retrieval systems is often evaluated based on a ranking measure, *e.g.* computed over the ranking of documents for a given query, or over the ranking of annotation labels for a given image. Well known ranking-based evaluation measures include precision-at-k (p@k) and mean average precision (MAP). However, most existing methods are optimized for prediction accuracy, which might yield suboptimal ranking performance. Therefore, we also consider optimizing the structured models, with label dependencies, for specific ranking measures.

The learning of ranking functions has become an active area of research, see *e.g.* (Le et al., 2010; Weston et al., 2010). A feature that is missing in the existing methods, however, is the ability to encode pairwise correlations between labels. Optimizing structured models with label interactions for a ranking loss is intimately related to the solution of quadratic assignment problems. Although such problems are in general NP-hard, we identify a polynomially-solvable subclass that still enables the modeling of a substantial number of pairwise rank interactions.

The rest of this chapter is organized as follows. In Section 4.2, we discuss how our work is related to recent work on image classification and annotation, and to learning to rank models. Then, we present our structured prediction model in Section 4.3, and its use in attribute-based classification in Section 4.4. In Section 4.5 we describe how to leverage user input in an interactive setting. In Section 4.6 we identify a class of models which allows for modeling label dependencies and to optimize for a ranking loss. Finally, we present experimental results in Section 4.7, and our conclusions in Section 4.8.

4.2 Related work

The dominant line of research for image annotation, object category recognition, and image categorization has focused on methods that deal with one label or object category at a time. The function that scores images for a given label is obtained by means of various machine learning algorithms, such as binary SVM classifiers using different linear or non-linear kernels (Zhang et al., 2007; Perronnin et al., 2010b), nearest neighbor classifiers (Makadia et al., 2008; Guillaumin et al., 2009a), and ranking models trained for retrieval (Grangier and Bengio, 2008) or annotation (Weston et al., 2010). While these methods do not explicitly model dependencies among the image labels, there are correlations in the classifier outputs, since the independent predictors use the same images to train the models and to predict these labels.

Classification is more challenging when dealing with many classes, both when the aim is to assign a single label to an image from many possible ones (Deng et al., 2010), as well as in the setting when for each image several labels should be predicted, *e.g.* all present object categories (Choi et al., 2010). To address the latter, there has been a recent focus on contextual modeling. For example in object recognition, the presence of one class may suppress the presence of another class that is negatively correlated, or promote a positively correlated class *e.g.* (Rabinovich et al., 2007; Desai et al., 2009; Choi et al., 2010).

In Rabinovich et al. (2007) the goal is to label the regions in a pre-segmented image with category labels, and a fully-connected conditional random field model over the regions is used. In Desai et al. (2009) a contextual model is proposed to filter the windows reported by object detectors for several categories. The contextual model includes terms for each pair of object windows that will suppress or favor spatial arrangements of the detections (*e.g. boat* above *water* is favored, but *cow* next to *car* is suppressed). A similar goal is pursued in Choi et al. (2010), where the scores of bounding boxes obtained by discriminatively trained object detectors is enhanced using a tree-structured model. This tree models the presence and location of the object category in the context of all other bounding boxes from the image. The parameters of the tree are learned in a generative way, from images with bounding-boxes. In our work, we also use tree structured models, but over global labels using only presences and absences of the labels, and we learn the complete model discriminatively.

Interactive labeling The interactive image annotation scenario we address in this chapter is related to active learning. The goal of active learning systems is to reduce the amount of labels needed, *e.g.* to train a classifier. The number of labels is reduced by actively selecting the samples from the input data to be labeled. For a literature survey on active learning methods, see *e.g.* (Settles, 2009).

In active learning for image classification, the learning algorithm starts with a set of labeled and unlabeled images. Iteratively, a classification model is learned from the labeled ones. Then, the learned model is used to determine which image is most valuable to be labeled next by the user. Such models have been used to learn from user input at different levels of granularity, *e.g.* by querying image-wide labels or precise object segmentation (Vijayanarasimhan and Grauman, 2009).

In our work, however, the system does not select images to be labeled at training time by a user to improve the model, but we assume that the training set is fully labeled. Instead, for a given image at test time, our system selects labels for which user-input is the most valuable in order to improve predictions on the other labels of the same image.

Attribute-based classification We also apply our structured models to attribute-based image classification, where an image is assigned to a given class based on a class-specific set of given attributes (Lampert et al., 2009; Branson et al., 2010). The advantages of attribute-based classification include the following.

1. It allows for zero-shot recognition, *i.e.* it can recognize images from unseen classes based on attribute predictions and a given attribute-to-class mapping.

2. The attribute representation can in principle encode an exponential number of classes.

3. By sharing attributes over different classes more images are available for each attribute, since images can be pooled from different classes.

In (Branson et al., 2010) a discriminative object recognition system, using SVM classifiers, is combined with a generative class-attribute model: for each class the object attributes values reported by different users (allowing for erroneous user responses and ambiguous object-attribute relationships) are modeled independently. To leverage user input for classification, the system asks the user to label the attribute that minimizes the entropy on the class label. Similarly, we also exploit user input at the level of attributes, but we learn recognition models for each attribute rather than for the object categories.

The zero-shot recognition system of Lampert et al. (2009), combines independent attribute predictors with attribute-to-class specifications to allow for recognizing unseen classes. The models we propose in this chapter go one step further by modeling the dependencies between attribute labels. This allows us to improve the attribute-based recognition, but also to better exploit the user input by asking more informative questions.

Learning to rank While the majority of image classification and annotation methods are optimized for prediction accuracy, the evaluation is often based on a ranking measure. Recently, learning ranking functions has become an active area of research for document retrieval, and image classification. Methods to learn score functions optimized for a variety of different performance measures have been developed, including mean average precision and precision-at-k, see *e.g.* (Joachims, 2005; Yue et al., 2007; Le et al., 2010; Weston et al., 2010). Once these models have been learned, a ranking for image annotation is computed by sorting the labels based on a score function computed for each label independently for a given image.

The class of linear ranking losses as defined in Le et al. (2010) is the set of loss functions that can be written as a linear function of the ranking. This class includes performance measures like the winner-takes-all (*i.e.* is the first item is correct), mean reciprocal rank (*i.e.* defined as $\frac{1}{k}$ where k is the rank of the single relevant item), and precision-at-k (p@k, *i.e.* how many relevant items are in the top-k). In general, optimizing for a linear loss, using independent scoring of labels, leads to a linear assignment problem, which has a worst case complexity that is cubic in the number of labels to rank (Le et al., 2010). An efficient approximate online learning model was introduced in Weston et al. (2010), for cases where it is too expensive to compute the score of the image for all labels.

The popular mean average precision (MAP) evaluation measure, provides a single-figure quality score by averaging p@k over all values of k occupied by relevant documents. However, the MAP measure is not in this class of linear losses. It is a quadratic ranking loss, since the loss contribution of a single label on a specific rank depends also on the relevance of the items with a higher rank. In general, the class of quadratic loss functions leads to NP-hard quadratic assignment problems, however for the special case of the MAP measure an efficient model has been found (Yue et al., 2007).

A feature which is lacking in these models is the possibility to encode pairwise correlations between labels. Recently a method was introduced to optimize for a structured-loss for multi-label classification under sub-modular pairwise label interactions (Petterson and Caetano, 2011). However, this method does not generalize to models for ranking based measures, and uses an approximation to the combinatorial optimization problem. On the contrary, the class of structured models we present, allow for tractable optimization for the precision-at-k measure and can encode a substantial number of label interactions.

4.3 Structured models for image annotation

Our goal is to obtain an expressive model that captures dependencies between the different image labels, but which still allows for tractable inference. We do so by defining a structured output problem using an energy function over the labels from the annotation vocabulary, and include pairwise interaction terms. This is equal to a conditional random field (CRF) where each node represents a label from the annotation vocabulary, and edges between nodes represent interaction terms between the labels.

Let $\boldsymbol{y} = [y_1, \ldots, y_L]^\top$ denote a vector of the L binary label variables, *i.e.* $y_i \in \{0, 1\}$. In this chapter we only consider binary labels, however the models proposed can be trivially extended to cases where labels can take three or more values. We use the probabilistic framework, with the Gibbs distribution, Eq. (2.41), which we repeat for clarity. The probability for a specific configuration $\boldsymbol{y}$, for an image is given by:

$$p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{x})} \exp\big(E(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{\theta})\big), \tag{4.1}$$

where $Z(\boldsymbol{x})$ is the partition function, and $E(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{\theta})$ is the energy function scoring the compatibility between an image $\boldsymbol{x}$ and a label vector $\boldsymbol{y}$, using the parameters $\boldsymbol{\theta}$, which we drop from notation to improve clarity of presentation. In the probabilistic framework, label prediction and elicitation are handled naturally using marginal probabilities and label entropy. In principle, these models can also be formulated in a max-margin framework, but then it is less clear how to define label elicitation strategies.

4.3.1 Tree-structured models on image labels

We start with using tree-structured CRFs, in which inference is tractable and can be performed by standard belief propagation algorithms (Bishop, 2006).

The trees are defined such that each node represents a single label, and $\mathcal{E} = \{e_1, \ldots, e_{L-1}\}$ defines the edges in the tree over the label variables. For now we assume a given set of edges; where $e_l = (i, j)$ indicates the presence of an edge between label i and label j. The energy for a configuration of labels $\boldsymbol{y}$ given an image $\boldsymbol{x}_n$ is:

$$E(\boldsymbol{y}, \boldsymbol{x}) = \sum_{i=1}^{L} \psi_i(y_i, \boldsymbol{x}) + \sum_{(i,j)\in\mathcal{E}} \psi_{ij}(y_i, y_j). \tag{4.2}$$

Where ψ_i denote the unary term for label i and ψ_{ij} denote the pairwise interaction term between labels i and j. For the unary terms we use generalized linear functions:

$$\psi_i(y_i = l, \boldsymbol{x}) = \boldsymbol{\phi}_i(\boldsymbol{x})^\top \boldsymbol{w}_i^l, \tag{4.3}$$

where $\boldsymbol{\phi}_i(\boldsymbol{x})$ is a feature vector for the image which may depend on the label index i, and $\boldsymbol{w}_i^l$ is the weight vector for state $l \in \{0, 1\}$. In Section 4.3.1.1, we will describe two options for learning the unary potential functions.

The pairwise potentials, defined by a scalar parameter for each joint state of the corresponding nodes, are independent of the image input:

$$\psi_{ij}(y_i = s, y_j = t) = v_{ij}^{st}. \tag{4.4}$$

We learn the parameters of the unary and pair-wise potentials for a specific tree-structure by the log-loss for correct prediction: $\mathcal{L} = \sum_n \log p(\boldsymbol{y}_n | \boldsymbol{x}_n)$. We use gradient-based methods, which requires evaluation of the marginal distributions on single variables and pairs of variables connected by edges in the tree. Using y_{in} to denote the value of label i for training image n, we have the following gradients:

$$\nabla_{\boldsymbol{w}_i^l} \mathcal{L} = \Big([\![y_{in} = l]\!] - p(y_i = l | \boldsymbol{x}_n) \Big) \boldsymbol{\phi}_i(\boldsymbol{x}_n), \tag{4.5}$$

$$\nabla_{v_{ij}^{st}} \mathcal{L} = [\![y_{in} = s, y_{jn} = t]\!] - p(y_i = s, y_j = t | \boldsymbol{x}_n). \tag{4.6}$$

4.3.1.1 Learning the unary potentials

In this section we describe two approaches of learning the parameters of the unary potential functions. The first uses a two-stage learning approach, while the second uses a joint learning approach.

Two-stage learning The first method we consider is a two-stage learning approach, where we pre-train binary SVM classifiers for each label. For each label i we define a very compact feature vector $\boldsymbol{\phi}_i(\boldsymbol{x}) = [s_i(\boldsymbol{x}), 1]^\top$, where $s_i(\boldsymbol{x})$ is the SVM score associated with label y_i. Two-stage learning has the advantages that it allows for a flexible choice in the used classifier, and since the number of free parameters in the CRF is limited, it allows for faster training without regularization (Nowozin and Lampert, 2011).

Joint learning The second method we consider is a joint learning approach, where the high-dimensional unary classifiers are learned jointly with the other parameters of the CRF. For each label i we set $\phi_i(\boldsymbol{x}) = \phi(\boldsymbol{x})$, a high-dimensional BoV or FV image representation. Using the Representer Theorem, see Section 2.2, we obtain:

$$\begin{aligned} \psi_i(y_i = l, \boldsymbol{x}_n) = \boldsymbol{w}_i^l\, \boldsymbol{\phi}(\boldsymbol{x})^\top &= \sum_m \alpha_{im}^l \boldsymbol{\phi}(\boldsymbol{x}_m)^\top \boldsymbol{\phi}(\boldsymbol{x}), \\ &= \sum_m \alpha_{im}^l\, k(\boldsymbol{x}_m, \boldsymbol{x}) = \boldsymbol{k}^\top \boldsymbol{\alpha}_i^l, \end{aligned} \tag{4.7}$$

where k is the kernel function, $\boldsymbol{k} = [k(\boldsymbol{x}_m, \boldsymbol{x})]_{m=1}^n$ is the vector of kernel evaluations for input $\boldsymbol{x}$, and $\boldsymbol{\alpha}_i^l$ is the coefficient vector to be learned for label i and state l. In this case, we minimize the regularized empirical loss function, Eq. (2.22), and use λ to trade-off between the log-loss term and the ℓ_2 regularizer.

4.3.1.2 *Obtaining the tree structure*

The interactions between the labels are defined by the edges of the tree. While all labels interact with each other, close by labels have a stronger influence on each other. Finding the optimal tree structure for conditional models is generally intractable (Bradley and Guestrin, 2010), therefore we resort to approximate methods for finding useful tree structures over the labels. We compare two methods to obtain a tree structure.

Mutual information The first method we consider is the optimal tree structure for a generative model. This is obtained by the Chow-Liu algorithm as follows (Chow and Liu, 1968). We define a fully connected graph over the label variables with edge weights given by the mutual information between the label variables. The mutual information between pairs of label variables is estimated from the empirical distribution of the labels in the training data. The optimal tree-structure for a generative model is then given by the maximum spanning tree in this graph.

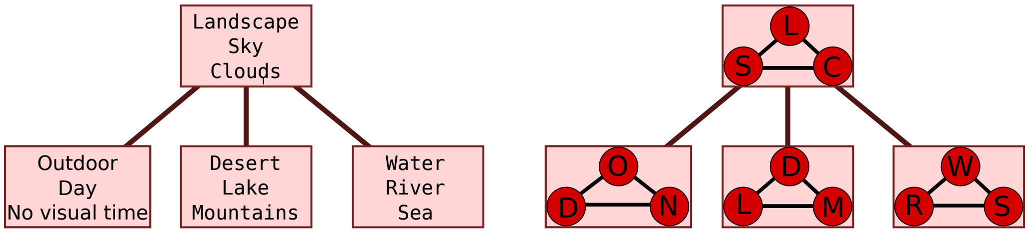


Figure 4.2 *Illustration of the graphical model of a tree with compound nodes, (*left*) a subset of a tree with multiple labels per node, (*right*) the structure of the graphical model is emphasized by showing the fully connected label-nodes in each compound node.*

Gradient based The second method we consider is to obtain the tree structure by iteratively growing a tree. Starting from a completely disconnected graph, in each iteration:

1. we add a single edge to the tree based on the current gradient,
2. we learn the parameters of the current graph, to minimize the log-loss.

We repeat this process until a tree model which spans over all nodes is obtained. To compute the gradient of any edge, also the ones not (yet) used in the current model, we can just use Eq. (4.6). As an indicator of the increase in log-likelihood, which we could obtain by including a particular edge, we use the ℓ_2 norm of the gradient w.r.t. the parameters of that edge. This is motivated by the fact that the ℓ_2 norm of the gradient is proportional to the increase in the log-likelihood by taking an infinitesimal step in the gradient direction.

4.3.2 Extensions of the tree-structured models

While the structured models defined in the previous section have the advantage to allow for tractable inference, they are limited in the label dependencies they can encode. In this section we propose two extensions that allows the encoding of more label dependencies, and yet maintain tractable inference. First, we introduce a graphical model that is a tree over groups of label variables. Second, we consider mixture-of-trees structured models.

4.3.2.1 Trees over groups of label variables

To accommodate for more dependencies between labels in the model, we consider the extension where we group label variables, and then define a tree over these groups. A label group can be seen as a fully connected set of variables in the graphical model. A tree model over those groups implies that the underlying graphical model has a certain cyclic-structure, *i.e.* it contains only local cycles: within each label group, and among

State	Marginal	Nature	Sky	Clouds
1	3.4 %	0	0	0
2	0.0 %	0	0	1
3	9.8 %	0	1	0
4	59.9 %	0	1	1
5	0.4 %	1	0	0
6	0.0 %	1	0	1
7	2.6 %	1	1	0
8	23.9 %	1	1	1
Marginal on label		26.9%	96.2%	83.8%

Figure 4.3 *Example of a compound node that combines three image labels and therefore has* $2^3 = 8$ *states. The label marginals are obtained by summing the node marginal probabilities of the corresponding joint states.*

neighboring groups in the tree, see Figure 4.2 for an illustration. Cyclic models remain tractable as long as they have a low treewidth (Bishop, 2006).

In practice, we determine a group size k, and model each state of the labels explicitly as a state of the compound node, which has 2^k states. In Figure 4.3 we show an example of a compound node with 3 labels, and thus 8 states. If k equals the number of labels L, we have the fully connected model, in which inference is intractable. The group size k relates to the treewidth of the graphical model, and offers a trade-off between expressiveness of the model, computational tractability and the risk of over-fitting on the training data.

Using belief propagation we now obtain node marginals, *i.e.* probabilities for each state of a node. However, we are still interested in the probability of label i being relevant for this image, *i.e.* $p(y_i = 1|\boldsymbol{x})$, since this is used to rank images for a specific label, to sort labels for a specific image, and for label elicitation. The label marginals are trivially obtained by summing the right entries of the node marginal; see Figure 4.3 for an example.

Grouping labels To obtain a partitioning of the labels, we perform agglomerative clustering based on mutual information, fixing in advance a maximum group size k. In each step, we merge the label groups that have the maximum mutual information, while allowing at most k labels per group. In the final partitioning, each label l is assigned to a single group g. With each group of variables, we associate a new variable $\boldsymbol{y}_g$ that takes as values the product space of the values of the labels in the group.

The unary potentials are defined as in Eq. (4.3), where y_i is replaced with $\boldsymbol{y}_g$, and hence take one of the 2^k states according to the values that labels in the group can take. For each state l of the joint-node g a weight vector $\boldsymbol{w}_g^l$ is learned. When we use the pre-trained SVM scores as feature vector, we define $\boldsymbol{\phi}_g(\boldsymbol{x}) = [\{s_i(\boldsymbol{x})\}_{i \in \mathcal{G}_g}, 1]$ as the extended vector of SVM scores associated with the labels in the group $\mathcal{G}_g$. The pairwise potential functions of Eq. (4.4) now link groups of k binary variables, and hence will be defined by 2^{2k} scalars. Therefore, the cost of message passing algorithms scales with $O(G2^{2k})$, where G is the number of groups. In order to maintain tractable inference, the group sizes should be fairly small (we use $k \leq 4$ in our experiments).

We determine a tree structure on the compound nodes using the same ideas as presented in the previous section. On the cover page of this chapter we show a tree with group size $k = 3$, obtained with the Chow-Liu algorithm. Although not forced, semantically related concepts are often grouped together (*e.g.* water related concepts in the *Water-River-Sea* node and plant related concepts in the *Plants-Flowers-Trees*) or they are in neighboring nodes (*e.g.* person related concepts around the *Single Person-No Person-Male* node).

4.3.2.2 Mixture-of-trees

As a second extension to incorporate more label dependencies we consider mixture-of-trees. The mixture models are defined either over trees with different group sizes k or over trees with different structures over a fixed set of nodes.

A mixture of T different trees, indexed by t, is defined as:

$$p(\boldsymbol{y}|\boldsymbol{x}_n) = \sum_{t=1}^{T} \pi_t p_t(\boldsymbol{y}|\boldsymbol{x}_n), \tag{4.8}$$

where π_t denotes the mixing weight, and $p_t(\boldsymbol{y}|\boldsymbol{x}_n)$ the probabilistic model of tree t.

The label marginals $p(y_i|\boldsymbol{x})$ are in this case obtained as the *mixture of the marginals* computed in the component models. This is easily seen from the following identities:

$$\begin{aligned} p(y_i|\boldsymbol{x}) &= \sum_{\boldsymbol{y}\backslash i} p(\boldsymbol{y}|\boldsymbol{x}) = \sum_{\boldsymbol{y}\backslash i} \sum_{t} \pi_t p_t(\boldsymbol{y}|\boldsymbol{x}) \\ &= \sum_{t} \pi_t \sum_{\boldsymbol{y}\backslash i} p_t(\boldsymbol{y}|\boldsymbol{x}) = \sum_{t} \pi_t p_t(y_i|\boldsymbol{x}). \end{aligned} \tag{4.9}$$

In the first and last equations we use the definition of the marginal probability, in the second we use the definition of the mixture, and in the third we swap the two sums.

We train each tree model independently, and then average the predictions of the individual trees using $\pi_t = 1/T$. Alternatively, the mixing weights can be learned concurrently while learning the trees, *e.g.* by using the EM algorithm to infer which tree corresponds to which image, possibly improving results.

Our mixture-of-trees model is related to (Pletscher et al., 2009), where a mixture over random spanning trees is used for approximate learning and inference in a single underlying intractable CRF model. Different from their work, we perform inference and learning independently in each tree, and mix maximum spanning trees of different node sizes.

4.4 Structured attribute-based classification

In this section we consider how the structured prediction models defined in the previous sections can be used for attribute-based image classification. This refers to a classification paradigm where an image is assigned to a given class $z \in \{1, \dots, C\}$ based on a set of attribute values (Branson et al., 2010; Lampert et al., 2009). An image belongs to exactly one class, but attributes are shared between different classes. For example, in the Animals with Attributes (AwA) data set (Lampert et al., 2009) different animals are defined in terms of attributes such as *has stripes, has paws, swims*, etc.

4.4.1 Structured attribute prediction

We apply our structured prediction model at the level of attributes, *i.e.* we learn a tree structured model over attributes, and the binary values y_i now refer to the presence or absence of an attribute for an image. As in (Lampert et al., 2009), we assume that the deterministic mapping between attributes and the C object classes is given, and denote the attribute configuration of class c by $\boldsymbol{y}_c$.

We define the distribution over classes by normalizing the likelihoods of the corresponding attribute configurations:

$$p(z = c|\boldsymbol{x}) = \frac{p(\boldsymbol{y}_c|\boldsymbol{x})}{\sum_{c'} p(\boldsymbol{y}_{c'}|\boldsymbol{x})} = \frac{\exp\big(E(\boldsymbol{y}_c|\boldsymbol{x})\big)}{\sum_{c'} \exp\big(E(\boldsymbol{y}_{c'}|\boldsymbol{x})\big)}. \tag{4.10}$$

The evaluation of $p(z|\boldsymbol{x})$ does not require belief-propagation, it suffices to evaluate $E(\boldsymbol{y}_c, \boldsymbol{x})$ for the C configurations $\boldsymbol{y}_c$, since the partition function $Z(\boldsymbol{x})$ cancels out.

4.4.2 Correction Terms

When using our attribute classification models as such, we observe that some classes tend to be much more often predicted than others, and that the prediction errors are mainly caused by assigning images to these over-predicted classes. As this also holds for the independent attribute prediction model, we assume the reason might be that some classes have rare (combinations of) attribute values.

In order to overcome this, we introduce a class-specific correction term u_c that plays a similar role as a class prior probability in a generative probabilistic model. We redefine the class prediction model of Eq. (4.10), as:

$$\tilde{p}(z = c|\boldsymbol{x}) = \frac{\exp\big(E(\boldsymbol{y}_c|\boldsymbol{x}) + u_c\big)}{\sum_{c'} \exp\big(E(\boldsymbol{y}_{c'}|\boldsymbol{x}) + u_{c'}\big)}. \tag{4.11}$$

To set the correction terms, we appeal to logistic discriminant training. If we have ground truth class labels for the training images, given by z_n, we could optimize the log-likelihood of correct classification, which is a concave function of the u_c:

$$\begin{aligned}\tilde{\mathcal{L}} &= \sum_n \log \tilde{p}(z = z_n|\boldsymbol{x}_n) \\ &= \sum_n E(\boldsymbol{y}_{z_n}|\boldsymbol{x}_n) + \sum_n u_{z_n} - \sum_n \log \sum_c \exp\big(E(\boldsymbol{y}_{z_n}|\boldsymbol{x}_n) + u_c\big) \\ &= \text{const.} + \sum_c n_c u_c - \sum_n \log \sum_c \exp\big(E(\boldsymbol{y}_{z_n}|\boldsymbol{x}_n) + u_c\big), \end{aligned} \tag{4.12}$$

where $n_c = \sum_n [\![z_n = c]\!]$ denotes the number of examples of class c. The gradient of the log-likelihood w.r.t. u_c is obtained as:

$$\nabla_{u_c} \tilde{\mathcal{L}} = n_c - \sum_n \tilde{p}(z = c|\boldsymbol{x}_n). \tag{4.13}$$

Both the log-likelihood and the partial derivative can be computed without access to the labels of the individual samples z_n; it suffices to know the label counts n_c.

Furthermore, from Eq. (4.13) we see that for the stationary point of $\tilde{\mathcal{L}}$ we have: $\sum_n \tilde{p}(z = c|\boldsymbol{x}_n) = n_c$. Therefore, setting the correction terms to maximize Eq. (4.12) will ensure that —in expectation— the test classes are predicted as often as they should.

Note that Lampert et al. (2009) also have integrated class specific correction terms in their attribute-based classification model, that uses independent attribute prediction models. They use $u_c = \ln p^*(\boldsymbol{y}_c)$, with:

$$p^*(\boldsymbol{y}_c) = \prod_l^L \sum_{c'}^C \frac{1}{C} [\![y_{lc} = y_{lc'}]\!].$$

In their model, classes with a high likelihood under a generative model are penalized in the discriminative model.

Setting the class counts In attribute-based classification, the training data is only labeled at the attribute level, and we do not have necessarily access to the counts of the class labels on the training data, let alone the case of zero-shot learning where we surely do not have the class counts. In these cases we can set the class proportions uniformly, $n_c = N/C$, so that the model will, in expectation, predict all classes equally often.

In reality, the test classes are not equally represented, and therefore, setting the u_c based on uniform proportions n_c is, in principle, not optimal. However, preliminary experiments where we set the u_c to match the label count on the training set, have shown only marginal further improvements in classification accuracy. Calibrating the models is also possible by using the (true or uniform) label counts n_c from the test images, leading to a transductive learning scenario. But again, preliminary experiments have shown that also this has only a minor impact on classification accuracy.

We thus conclude that it is important to set the correction terms so as to avoid grossly over or under predicting certain classes, but that it is less important to finely tune these terms using other than uniform counts n_c or using the test images instead of the training images.

Correction terms using mixture-of-trees In this section we briefly discuss how we handle the correction terms when using the mixture-of-trees. In this case, we mix the class predictions made by the different models as:

$$p(z = c|\boldsymbol{x}) = \sum_t \pi_t p_t(z = c|\boldsymbol{x}), \tag{4.14}$$

where the π_t are the mixing weights associated with different tree-structured models and $p_t(z = c|\boldsymbol{x})$ indicates the class prediction from one such a tree model.

To balance the class predictions of the mixture model, we learn separate correction terms for each component model $p_t(z = c|\boldsymbol{x})$ as described above. Doing so ensures that the mixture-of-trees model is also calibrated, since:

$$\begin{aligned}\sum_n p(z{=}c|\boldsymbol{x}_n) &= \sum_t \pi_t \sum_n p_t(z = c|\boldsymbol{x}_n) \\ &= \sum_t \pi_t n_c = n_c.\end{aligned} \tag{4.15}$$

In the first equality we use the definition of the mixture Eq. (4.14), and we swap the sums; in the second equality we use the fact that each tree of the mixture has been calibrated and therefore $\sum_n p_t(z = c|\boldsymbol{x}_n) = n_c$; the last equality follows from the fact that the mixing weights sum to one. This shows that balancing each tree ensures that the mixture-of-trees model is calibrated as well.

4.4.3 Effectiveness of correction terms

To show the effectiveness of the proposed correction terms, we conducted an experiment on two classification settings for the AwA data set. In the first classification setting, we follow the settings of (Lampert et al., 2009), where the test set consist of 10 classes, while training is performed on the other 40 classes. Our second setting uses all 50 classes of

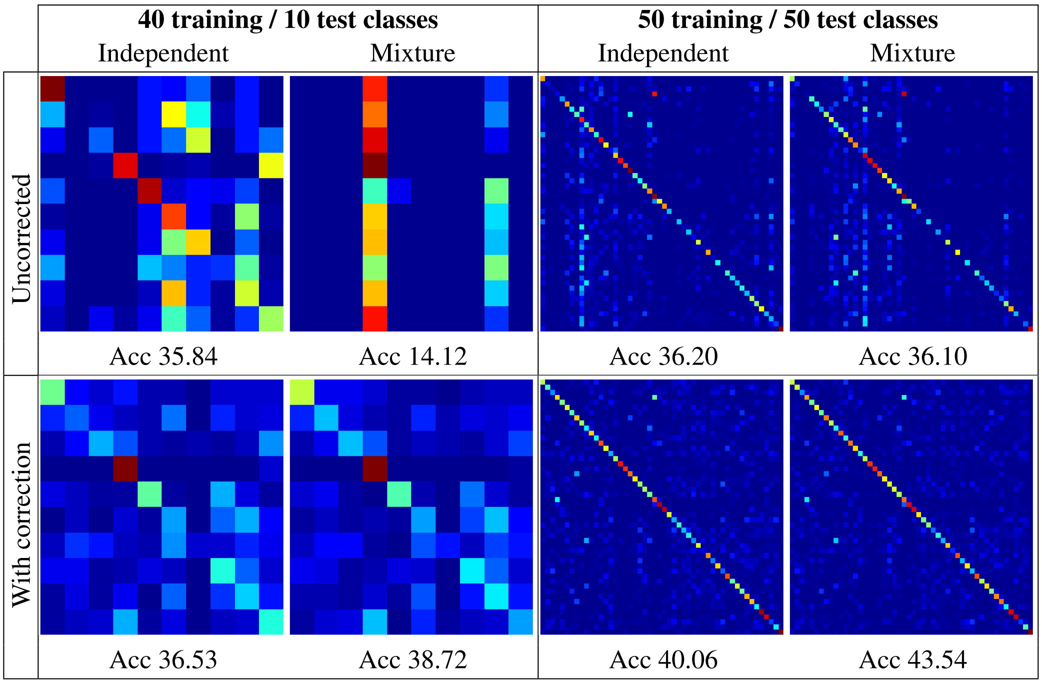


Figure 4.4 *Influence of the correction terms for attributed-based classification when using independent models and mixtures-of-trees models. See text for details.*

the AwA data set for testing, and training images are sampled from all classes to learn the attribute prediction models.

The results of this experiment are shown in Figure 4.4. On the top row, the confusion matrices are shown as obtained using Eq. (4.10), *i.e.* without incorporating the correction terms. It shows the confusion matrices both for the independent model (*i.e.* without pairwise terms) and for our mixture-of-trees structured model using the two classification settings. The bottom row shows the confusion matrices when the correction terms are used, as given in Eq. (4.11), using uniform class proportions $n_c = N/C$

The four panels on the top row show the imbalance of the class predictions for any of the methods and settings. For example, in the first panel, we see that using independent attribute prediction models, class 2 is hardly predicted for any test image, while in the second panel, we observe that the mixture-of-tree structured model only frequently predicts class 4 and 9 for the test images. In the two right-most panels, we also observe severe differences in how often the classes are predicted, *c.f.* the vertical stripes in the confusion matrices. This shows that the imbalance in the predictions is not due to using different test classes and training classes. The bottom row shows a more balanced prediction over the classes, which demonstrates how the correction terms can suppress or promote certain classes, allowing us to reduce the severe imbalance in how often the test classes are predicted.

4.5 Interactive image annotation and classification

In the interactive image annotation scenario, a user is asked iteratively to reveal the value of a selected label or attribute. This user input will then be taken into account for predictions on the other labels, or other classes for attributed-based classification.

While a random choice of labels is a possibility, we will experimentally show that this is far from optimal. We propose a label selection strategy whose aim is to minimize the uncertainty of the remaining labels, or class, given the test image. The proposed strategy resembles query strategies used in active learning (Settles, 2009), and the maximum information gain criterion (Branson et al., 2010).

4.5.1 Label elicitation for image annotation

In this section we describe an entropy-based method to select the label to be set by a user in an interactive image labeling scenario. Our goal is to select the label i for which knowing its ground truth value minimizes the uncertainty on the other labels. To achieve this, we propose to select the label i which minimizes the entropy of the distribution on the label vector $\boldsymbol{y}$ given the expected user input for that label.

Let us use y_i^l to denote $y_i = l$, and $\boldsymbol{y}_{\setminus i}$ to denote all label variables except y_i. Since the value of y_i of label i is not known prior to the moment that it is set by the user, we evaluate the expected conditional entropy:

$$H(\boldsymbol{y}_{\setminus i}|y_i, \boldsymbol{x}) = \sum\nolimits_l p(y_i = l|\boldsymbol{x}) H(\boldsymbol{y}_{\setminus i}|y_i^l, \boldsymbol{x}), \tag{4.16}$$

where

$$H(\boldsymbol{y}_{\setminus i}|y_i^l, \boldsymbol{x}) = -\sum\nolimits_{\boldsymbol{y}_{\setminus i}} p(\boldsymbol{y}_{\setminus i}|y_i^l, \boldsymbol{x}) \ln p(\boldsymbol{y}_{\setminus i}|y_i^l, \boldsymbol{x}). \tag{4.17}$$

Using the fact that $H(\boldsymbol{y}|\boldsymbol{x})$ does not depend on the selected label i, and given the basic identity of conditional entropy, see *e.g.* (Bishop, 2006), we have:

$$H(\boldsymbol{y}|\boldsymbol{x}) = H(y_i|\boldsymbol{x}) + H(\boldsymbol{y}_{\setminus i}|y_i, \boldsymbol{x}). \tag{4.18}$$

We hence conclude that minimizing Eq. (4.16) for y_i is equivalent to maximizing $H(y_i|\boldsymbol{x})$. Therefore, we select the label $i^* = \operatorname{argmax}_i H(y_i|\boldsymbol{x})$, to be set by the user.

In order to select a collection of labels to be set by a user, we proceed sequentially by first asking the user to set just one label. We then repeat the procedure while conditioning on the label(s) already provided by the user. Another possibility is to select a group of labels at once, which is nevertheless suboptimal as it cannot leverage information contained in the user input in the selection procedure.

To compute the label marginals while conditioning on the user input, we add an additional unary term per node. The value of this unary term depends on the user input: we add zero energy to all (joint-)states that are compatible with the user input, and infinite energy to those that are not. In the example of Figure 4.3, where we have three labels per node, the user input *Sky=true* would incur infinite energy for states $1, 2, 5$, and 6 of the compound node, which ensures they receive zero probability.

For interactive image labeling, it is interesting to evaluate the proposed methods using a user study, where several people are asked to annotate images using the proposed methods. In such a real life setting, the model should allow for ambiguous user annotations as in (Branson et al., 2010). However, this falls beyond the scope of this manuscript.

4.5.2 Attribute elicitation for image classification

In the case of attribute-based image classification we could use the same label elicitation strategy as for image annotation, as described above. However, since the final aim is to improve the class prediction, we use an attribute elicitation criterion that is geared towards minimizing uncertainty on the class label, rather than uncertainty at the attribute level.

The main insight is that the information obtained from a revealed attribute value depends on the agreement among the classes on this attribute. If some of the probable classes do not agree with the observed value it will rule out the classes with a contradicting attribute value and concentrate the probability mass on the compatible classes. Therefore, any informative question will at least rule out one of the possible classes, and thus at most $C-1$ attributes need to be set by the user.

In order to select the attribute which should be set by the user, we minimize the conditional class entropy $H(z|y_i, \boldsymbol{x})$. Using the identity:

$$H(z, \boldsymbol{y}|\boldsymbol{x}) = H(y_i|\boldsymbol{x}) + H(z|y_i, \boldsymbol{x}) + H(\boldsymbol{y}_{\setminus i}|z, y_i, \boldsymbol{x}), \tag{4.19}$$

we make the following observations.

1. The left-hand-side, $H(z, \boldsymbol{y}|\boldsymbol{x})$, does not depend on the choice of attribute y_i to elicit.
2. The last term $H(\boldsymbol{y}_{\setminus i}|z, y_i, \boldsymbol{x})$ equals zero, since for each class there is a unique setting of the attribute values.

Therefore, selecting the attribute to minimize the remaining entropy on the class label is equivalent to selecting the attribute with the largest marginal entropy $H(y_i|\boldsymbol{x})$.

In the attribute-based classification model, however, $p(y_i|\boldsymbol{x})$ is differently defined as in the image annotation model. Here the probability $p(y_i|\boldsymbol{x})$ is implicitly defined through Eq. (4.10), which essentially rules-out all attribute configurations, except the ones that correspond to one of the C classes. Therefore, we have:

$$p(\boldsymbol{y}|\boldsymbol{x}) = \sum_c p(z = c|\boldsymbol{x}) \, [\![\boldsymbol{y} = \boldsymbol{y}_c]\!], \tag{4.20}$$

and

$$p(y_i|\boldsymbol{x}) = \sum_{\boldsymbol{y}_{\setminus i}} p(\boldsymbol{y}|\boldsymbol{x}) = \sum_c p(z = c|\boldsymbol{x}) \, [\![\boldsymbol{y}_i = \boldsymbol{y}_{ic}]\!], \tag{4.21}$$

where y_{ic} denotes the value of attribute i for class c.

We note that the attribute elicitation mechanism for interactive attributed-based image classification is not changed when using different variants of the model (using correction terms, using trees over groups of attributes, or mixtures of such models). In all cases we obtain a class prediction model $p(z = c|\boldsymbol{x})$, which, combined with the class specific label configuration $\boldsymbol{y}_c$, is used to compute marginals over the attribute variables:

$$p(y_i = 1|\boldsymbol{x}) = \sum_c p(z = c|\boldsymbol{x}) \, y_{ic}. \tag{4.22}$$

The label marginals are used to select the attribute to be set by the user. As for image annotation, sequences of user queries are generated progressively by conditioning on the image and all the attribute labels given so far to determine the next attribute to query.

4.6 Learning quadratic ranking functions

In this section we describe how we can learn for a ranking loss function while modeling pairwise correlations for image annotation. For this, we use the margin-rescaling structured SVM framework (Taskar et al., 2003; Tsochantaridis et al., 2005, see Section 2.2.1). The goal is to learn a score function that will compute a ranking Π for the labels of an image. For clarity we repeat the SVM learning objective:

$$\ell_{\text{MR}}(\boldsymbol{x}, \boldsymbol{y}, \theta) = \max_{\Pi} \Big\{ \Delta(\Pi, \boldsymbol{y}) + f(\boldsymbol{x}, \Pi; \theta) \Big\} - f(\boldsymbol{x}, \tilde{\Pi}; \theta), \tag{4.23}$$

where $\Delta(\Pi, \boldsymbol{y})$ is a ranking-based cost function, $f(\boldsymbol{x}, \Pi; \theta)$ is a score function with parameters $\boldsymbol{\theta}$, and $\tilde{\Pi}$ is any ranking that attains the minimum possible loss. We define the ranking by the permutation matrix Π, where $\Pi_{ik} = 1$ denotes that label i is placed on rank

k. In order to be a valid permutation, Π has the following constraints, $\forall_{ik}: \ \Pi_{ik} \in \{1, 0\}$, $\forall_k: \ \sum_i \Pi_{ik} = 1$ and $\forall_i: \ \sum_k \Pi_{ik} = 1$.

In this chapter we consider linear ranking cost functions as defined in Le et al. (2010):

$$\Delta(\Pi, \boldsymbol{y}) = \langle \Pi \boldsymbol{a}, \boldsymbol{b}(\boldsymbol{y}) \rangle, \tag{4.24}$$

where $\boldsymbol{a}$ is a cost-specific vector, and $\boldsymbol{b}(\boldsymbol{y})$ is a vector which encodes ground truth relevance annotation. This class of ranking functions contains, among others, the p@k cost, which is obtained by setting $\boldsymbol{b} = 1 - \boldsymbol{y}_n$ and $a_i = \frac{1}{k}$ for $1 \leq i \leq k$, and $a_i = 0$ otherwise. In this manner, placing an irrelevant label in the top k increases the loss by $\frac{1}{k}$.

4.6.1 Quadratic score functions

The quadratic score functions we consider can be written as the sum of a quadratic and a linear function of the permutation matrix:

$$f(\boldsymbol{x}, \Pi) = f_l(\boldsymbol{x}, \Pi) + f_q(\boldsymbol{x}, \Pi), \tag{4.25}$$

$$= \boldsymbol{s}^\top \ \Pi \ \boldsymbol{c} + \text{Tr}\{F \ \Pi \ D \ \Pi^\top\}, \tag{4.26}$$

where in the linear term, $\boldsymbol{s}$ is a vector of label scores for an image, and $\boldsymbol{c}$ is a design vector which typically encourages labels with a high score to be ranked high. In the quadratic term, F is a matrix of scores per label combination, and D is a design matrix, which could, for example, encode an incentive to put labels i and j close to each other in the ranking if $F_{ij} > 0$, and to place them far apart otherwise.

Finding the maximizer of score functions defined by Eq. (4.25), takes the form of a quadratic assignment problem (QAP) (Koopmans and Beckmann, 1957; Burkard et al., 2009). Since in general QAPs are NP-hard, the above formulation is of limited practical importance, unless approximate solutions are accepted or additional structure is imposed on the quadratic terms.

4.6.2 Polynomial-time inference for precision-at-k

A few specific classes of QAPs are known to be solvable in polynomial time. One such class, is the class of problems where the interactions are restricted to form a c-star graph (Erdogan, 2006), see Figure 4.5 for an illustration. The c-star graph models pairwise interactions between any node and a small group of c nodes in the core of the star, while among the nodes outside this core there are no pairwise interactions.

In this case, we can solve the QAP as follows. For given positions in the ranking of the C elements in the core of the star, the remaining items have no interactions among each

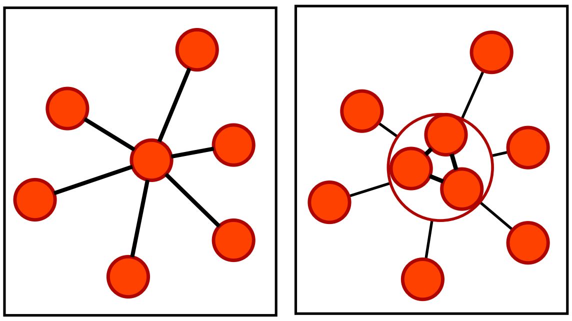


Figure 4.5 *Illustration of a C star-graph, with a core consisting of a single node (*left*), and 3 nodes (*right*). Given the ranks of the core elements, the graph becomes independent and the QAP can be solved as a linear assignment problem, see text for details.*

other. The remaining optimization problem is reduced to a linear assignment problem (LAP), which can be solved in $O(N^3)$, for N elements to rank. As there are $\binom{N}{C}C!$ placements of the C core elements, the original QAP can be solved in $O(N^{C+3})$, for both inference and loss-augmented prediction.

Score functions for precision-at-k In this section we define our model for the specific case where the evaluation measure at hand is precision-at-k (p@k). Before defining our score functions, we observe that p@k is sensitive only to these elements which are in the top k, and neither to their ordering, nor to the ordering of the elements outside the top k. Therefore, as far as the cost function is concerned, instead of considering the set of all permutations, we restrict ourselves to the set of all subsets of size k as possible outputs. We design a score function that exhibits the same invariances as the loss, since there is little interest in differentiating among outputs that are equivalent w.r.t. the loss function.

The linear term $f_l(\boldsymbol{x}, \Pi) = \boldsymbol{s}^\top \Pi \boldsymbol{c}$ is designed to score highest for selecting the k labels with the highest score for an image. This can be achieved by defining $\boldsymbol{c}$ such that $c_i = 1$ for $1 \leq i \leq k$, and $c_i = 0$ otherwise. Since $\boldsymbol{c}$ is constant in the first k elements, it is invariant to permutations among these, and similar for the last $N - k$ elements.

The quadratic term $f_q(\boldsymbol{x}, \Pi) = \text{Tr}\{F \Pi D \Pi^\top\}$ is also designed to be invariant for permutations among the top k labels, and among labels not in the top k. To that end, we define the matrix D to have a corresponding block structure where all elements in the same block have an equal value:

$$D = \begin{pmatrix} D_1 & D_2 \\ D_3 & D_4 \end{pmatrix},$$

where D_1 is a $k \times k$ matrix, D_4 a $(n-k) \times (n-k)$ matrix, and $D_2 = D_3^\top$ a $k \times (n-k)$ matrix. Without loss of generality, we can set $D_4 = \mathbf{0}$, since subtracting a constant from all blocks will not change the predictions, nor the bound on the loss function. Similarly, without loss of generality, blocks D_2 and D_3 can be set to zero by adding constants to the score functions s_i, and finally, we can set D_1 to contain only 1's.

Using these design choices, and $\boldsymbol{t}$ to denote the indicator vector of which labels are in the top k, *i.e.* $t_i = [\![\pi_i \leq k]\!]$, where π_i is the rank of label i, the score function is simplified to:

$$f(\boldsymbol{x}, \Pi) = \boldsymbol{t}^\top \boldsymbol{s} + \boldsymbol{t}^\top F \boldsymbol{t} = s \sum_i t_i s_i + \sum_{i,j} t_i t_j f_{ij}. \tag{4.27}$$

4.6.3 Efficient implementation of c-star models

In this section we show who we can efficiently compute the score functions defined in the previous section for c-star models. We invoke the observation that an c-star QAP can be solved by solving a set of LAPs, one for each assignment of the core of the star. Using i to index over labels not in the core of the star, and c and c' to index over core-labels, we can express the score function for a fixed assignment of the core as:

$$f(\boldsymbol{x}, \Pi) = \sum_i t_i \Big(s_i + \sum_c t_c f_{ic} \Big) + \sum_c t_c s_c + \sum_{c,c'} t_c t_{c'} f_{cc'}. \tag{4.28}$$

For fixed assignments of the core, the last two terms are constant. Therefore, the score is maximized by selecting the elements i with the largest values of:

$$s_i + \sum_c t_c f_{ic}.$$

The best overall subset of k labels can be found by looping over the 2^C configurations of the core elements to be in the top k or not, and for each configuration to find the best labels to add to the top k from the non-core elements. Selecting the k largest elements in a list of length N can be done in $O(N)$ (Blum et al., 1973), and sorting these in $O(k \log k)$. Therefore, the overall algorithm to find the best subset of k items has running time $O(2^C(N + k \log k))$. The same holds for loss augmented inference, since we can write $\Delta_{p@k}(\Pi, \boldsymbol{y}) = \langle \boldsymbol{t}, \boldsymbol{1} - \boldsymbol{y} \rangle / k$, and by replacing the s_i with $s_i + (1 - y_i)/k$, the best subset is obtained in the same manner.

Gradient calculation In practice we use a sub-gradient descend method to learn the parameters of the model. At each gradient evaluation we compute the loss-augmented prediction Π^*, and the maximum scoring zero loss ranking $\tilde{\Pi}^*$ to compute the gradient of L w.r.t. the parameters. Note that in this way we obtain the true gradient, except when either of the two maximizers is not unique, which is a set of measure zero in the parameter space.

Using $\boldsymbol{t}^*$ and $\tilde{\boldsymbol{t}}^*$ to denote the indicator vectors for items ranked in the top k according to Π^* and $\tilde{\Pi}^*$ respectively, the gradients of the loss-bound in Eq. (4.23) are easily found using the definition of the score function in Eq. (4.27):

$$\nabla_F \, \ell_{\text{MR}} = \boldsymbol{t}^* \boldsymbol{t}^{*\top} - \tilde{\boldsymbol{t}}^* \tilde{\boldsymbol{t}}^{*\top}, \tag{4.29}$$

$$\nabla_s \, \ell_{\text{MR}} = \boldsymbol{t}^* - \tilde{\boldsymbol{t}}^*. \tag{4.30}$$

Thus for an interaction parameter f_{ij} we obtain a gradient signal if i and j are both in the top k according to Π^*, but not according to $\tilde{\Pi}^*$, or vice versa. Similarly, for the label scores s_i we obtain a gradient signal if Π^* and $\tilde{\Pi}^*$ do not agree on i being in the top k.

4.7 Experimental evaluation

In this section we describe our experimental evaluation. We first present the used data sets, features and evaluation measures. Followed by, in Section 4.7.2, the results on automatic and interactive image annotation, in which we experiment with different features for the unary terms, different graph structures, and compare to state-of-the art methods. In Section 4.7.3 we present the results on attribute-based image classification, in Section 4.7.4 we show results of a multi-word query retrieval experiment, and in Section 4.7.5 we present the results when learning for the precision-at-k measure.

4.7.1 Dataset, features and evaluation measures

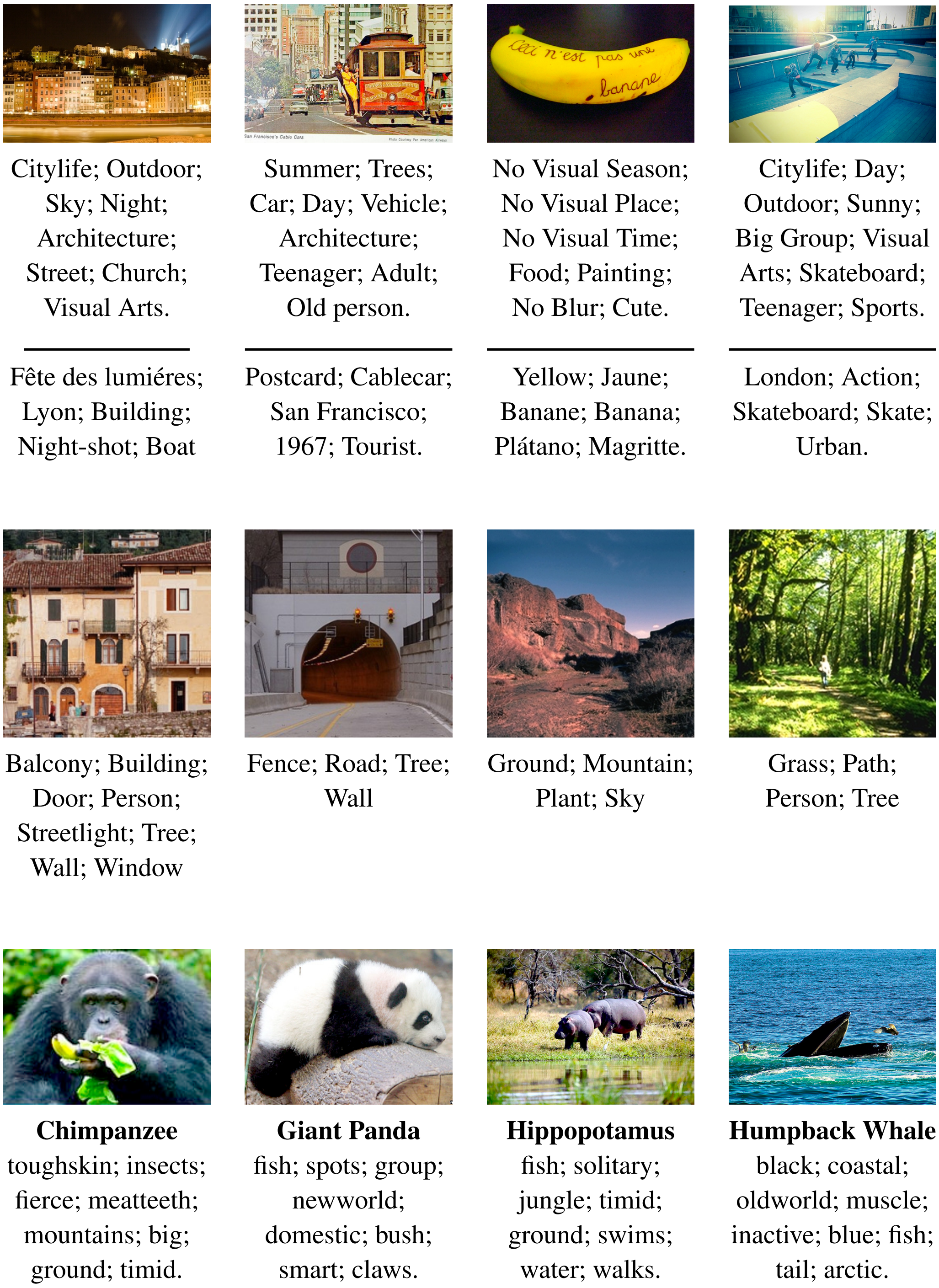
Before presenting our experimental results we first describe the different data sets, features and evaluation functions we use. We also provide the details of the two-stage learning approach we use in most of our experiments.

4.7.1.1 Data sets and features

We have performed experiments on three recent public data sets, which we describe in turn below together with the features used for each data set. In Figure 4.6 we show example images from all three data sets.

ImageCLEF'10 data set We use *ImageClef'10* to refer to the subset of the MIR-Flickr data set (Huiskes and Lew, 2008) that was used as training set in the ImageClef'10 Photo Annotation Challenge[2](Nowak and Huiskes, 2010). For the challenge, the training set consisted of 8,000 images and the images were labeled with 93 diverse concepts.

[2] Available at `http://www.imageclef.org/datasets`.



Citylife; Outdoor; Sky; Night; Architecture; Street; Church; Visual Arts.

Summer; Trees; Car; Day; Vehicle; Architecture; Teenager; Adult; Old person.

No Visual Season; No Visual Place; No Visual Time; Food; Painting; No Blur; Cute.

Citylife; Day; Outdoor; Sunny; Big Group; Visual Arts; Skateboard; Teenager; Sports.

Fête des lumiéres; Lyon; Building; Night-shot; Boat

Postcard; Cablecar; San Francisco; 1967; Tourist.

Yellow; Jaune; Banane; Banana; Plátano; Magritte.

London; Action; Skateboard; Skate; Urban.

Balcony; Building; Door; Person; Streetlight; Tree; Wall; Window

Fence; Road; Tree; Wall

Ground; Mountain; Plant; Sky

Grass; Path; Person; Tree

Chimpanzee toughskin; insects; fierce; meatteeth; mountains; big; ground; timid.

Giant Panda fish; spots; group; newworld; domestic; bush; smart; claws.

Hippopotamus fish; solitary; jungle; timid; ground; swims; water; walks.

Humpback Whale black; coastal; oldworld; muscle; inactive; blue; fish; tail; arctic.

Figure 4.6 *Example images from the* ImageClef'10 *data set (*top row*), the* SUN'09 *data set (*middle row*), and the* AwA *data set (*bottom row*) together with some of the relevant labels. For the* ImageClef'10 *data set we also show some of the Flickr tags which are provided for both the training and test set.*

In this case we tackle a multi-modal labeling task, since for each image the corresponding set of Flickr-tags are provided, both at train time and test time. Hence, in our experiments we use an early-fusion concatenation of visual and textual features. As visual features we use FVs over 128-D SIFT and 96-D LCS descriptors, which are reduced to 64 dimensions by PCA. We use a MoG with 256 Gaussians as a visual vocabulary, and compute the FV w.r.t. to mean and covariance only. To incorporate the weak geometry of the scene we use three spatial layouts (1×1, 2×2, and 1×3). Finally, the FVs are power and ℓ_2 normalized. More details on the used image features is given in Section 2.1.2. As textual representation for an image we use the ℓ_2 normalized vector of binary absences and presences of the 698 most common Flickr-tags.

In most of our experiments on *ImageClef'10*, we split the data into five folds, *e.g.* by using fold 1, we learn training classifiers and model parameters on fold 2 to 5, and evaluating the model on fold 1. We report results averaged over the folds, unless otherwise stated. For the sake of clarity we omit standard deviations since they are small compared to the differences between the prediction methods.

In Section 4.7.2.4 we compare the performance of our structured models to the participants of the *ImageClef'10* challenge. For these experiments we use the complete training set of the challenge and evaluate performance on the 10,000 images from the test set. Our baseline method is the winning system of the challenge (Nowak and Huiskes, 2010; Mensink et al., 2010a).

SUN'09 data set The *SUN'09* data set was introduced in (Choi et al., 2010) to study the influence of contextual information on localization and classification[3]. For this data set we use the same visual features as for *ImageClef'10* and we use the training set of 4367 images and the test set of 4317 images as defined by the authors. In contrast to the *PASCAL VOC 2007* (Everingham et al., 2007) data set, which has only 20 labels and over 50% of the images having only a single label, the *SUN'09* set contains more labels (107) and around 5 labels per image on average.

Animals with Attributes data set The *Animals with Attributes* (*AwA*, Lampert et al., 2009) data set contains images of 50 animal classes, and a definition of each class in terms of 85 attributes, for each class around 600 images are available. We follow the authors, using the provided features[4], the same sum of RBF-χ^2 kernels, and the same 40 training and 10 test classes. The features provided are: color histograms, local self-similarity, pyramid HOG, SIFT and colorSIFT, and SURF. We use this data set both to test image annotation of the 85 attributes and attribute-based classification.

[3] Available at `http://people.csail.mit.edu/myungjin/HContext.html`.

[4] Available at `http://attributes.kyb.tuebingen.mpg.de/`.

4.7.1.2 Evaluation measures and implementation

Evaluation measures For the image annotation and classification experiments we measure the performance of the methods using:

- MAP, a retrieval performance measure, which is the mean average precision (AP) over all labels, where AP is computed over the ranked images for a given label.
- iMAP, correlates to the number of corrections needed to obtain a correct image labeling; it is the mean AP over all images, where AP is computed over the ranked labels for an image.

Two-stage learning of unary potentials In most of our experiments, we apply a two-stage learning approach for the unary potentials. As a feature vector for the unary potentials in our structured models we use pre-trained binary SVM classifier scores. To obtain representative SVM classification scores for the training set, we use a method similar to Platt scaling (Platt, 2000). We use a subset of the training set to obtain classification scores for another subset of the training set. This is important because SVM classifiers will (almost) perfectly separate the training set, due to the high capacity of our image features. If we would use SVM scores which separate perfectly the training set to train the parameters of the structured model, it would seem that any structure is unnecessary.

We split each training set into several subsets (in our experiments, we have used 4 or 5 subsets), and for each image n, the classification score $s_i(\boldsymbol{x}_n)$ is obtained by training a binary SVM for concept i on the union of subsets not containing the image n. This assures us that the obtained scores are unbiased, *i.e.* the data is not perfectly separated, and allows us to learn the parameters of the structured models.

For the independent models, we use these unbiased scores to learn a sigmoid function, transforming SVM outputs into probabilistic outputs (Platt, 2000). For images in the test set we use the SVM scores obtained by classifiers trained on all training images.

The classification scores, train/test splits for *ImageClef'10* data set, and the multi-word queries (see Section 4.7.4) are available for download[5].

4.7.2 Image annotation and classification

In this section we evaluate our structured predictions models in the fully automatic and interactive image annotation task on the three data sets. The comparison in this section is between the independent model and trees using the SVM based unary potentials, with the tree structure being obtained based on the mutual information. We also consider using a mixture-of-trees that has different node sizes.

[5] Available at `http://lear.inrialpes.fr/~mensink/data`.

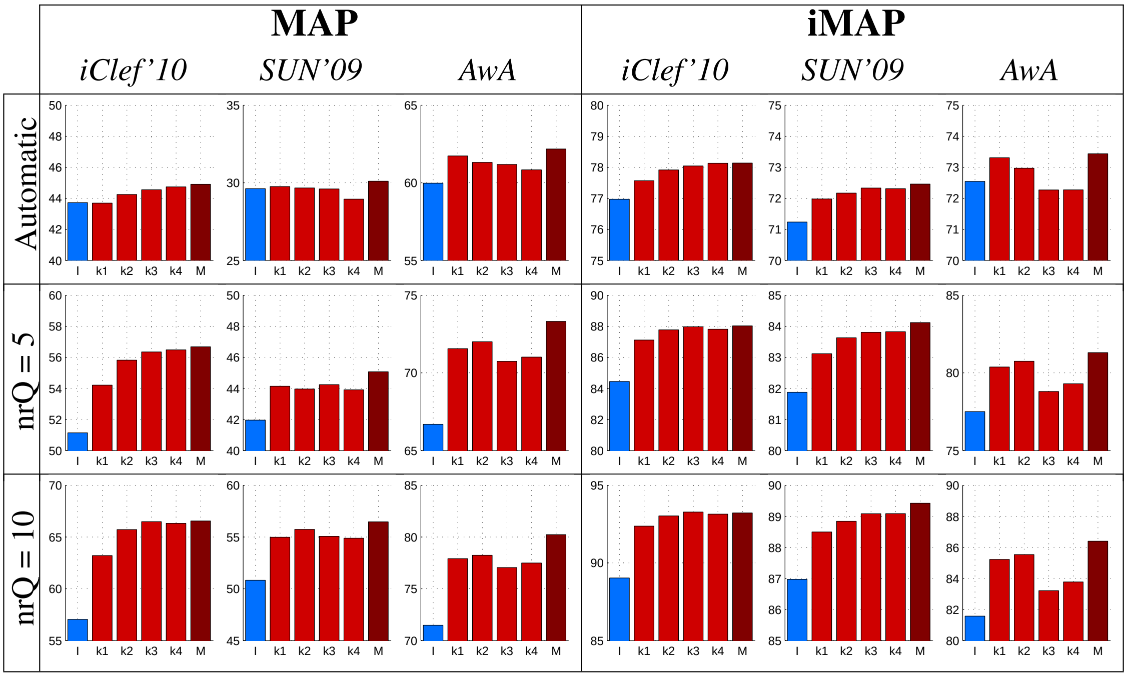


Figure 4.7 *Overview of the performance for the fully automated prediction setting (*top row*), and an interactive setting with 5 and 10 questions (*middle and bottom row*). We compare results of the independent model (blue), the trees with group sizes* $1 \leq k \leq 4$ *(light-red), and the mixture-of-trees model (dark-red). Note the different y-scales.*

4.7.2.1 Fully automatic image annotation

First, we analyze the influence of the structured models in the setting of fully automatic label prediction. Therefore, we evaluate the image annotation performance on MAP and iMAP, the results can be found in Figure 4.7, first row. For each data set, we compare the independent prediction model (blue) against trees with a group size of $k = 1 \dots 4$ (light-red), and to the mixture of these 4 trees (dark-red).

To the best of our knowledge, our independent classifiers (the blue bars in Figure 4.7) have state-of-the-art MAP performance on *ImageClef'10* (conform Section 4.7.2.4). For the *SUN'09* and *AwA* data sets, we are the first to report MAP over image labels/attributes. In Section 4.7.2.4 we show that our baseline classifier outperforms previously published results on *SUN'09* using another evaluation measure. For the *AwA* data set, in Section 4.7.3, we compare our baseline classifier in a zero-shot setting to the state-of-the art results of (Lampert et al., 2009).

From Figure 4.7, we can observe that the MAP/iMAP performance of the structured prediction models is about $1 - 1.5\%$ higher than of the independent model. The performance differences between the models with different group sizes k should be seen as a trade-off between model expressiveness and overfitting on the training data. For all data sets the mixture-of-trees performs the best.

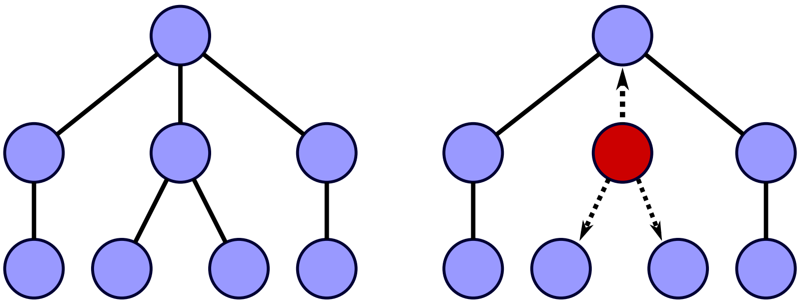


Figure 4.8 *Illustration of the effect of user interaction, (*left*) a tree without user input is shown, (*right*) a tree is shown where one node (marked red) is observed due to user input. As a result the tree is decomposed into several independent sub-trees.*

The improvement of the structured models over the independent model is relatively modest in the fully automatic setting. This might be due to the fact that the trees only propagate visual information in this case, which is already very well captured by the independently trained SVM classifiers. In the next section, we will show that in an interactive annotation scenario, the tree based structures can much better exploit and propagate user input than an independent model.

4.7.2.2 Interactive image annotation

In order to further show the benefit of the proposed structured model, we simulate an interactive image annotation system. The system iteratively selects a label based on the entropy selection criterion, to be set by the "oracle" (the ground truth in our experiments, but this could be a user). The annotation results obtained after 5 and 10 "*questions*" (*i.e.* labels asked to the oracle) are shown in the second and third rows of Figure 4.7.

As expected, in this setting the structured models benefit more from the user input, since they propagate the information provided by the user to update their belief of all labels. The independent model can only update the predictions of the questioned image/label combinations (setting them to either 1 or 0), which explains the increase in their MAP/iMAP performance. In the structured models, on the other hand, some of the label variables become observed due to the user input. These nodes now no longer propagate visual information, but they send messages based on their observed values for the labels to the nodes connected to them, see the illustration in Figure 4.8. This new information translates to better predictions on the unknown labels in the tree.

Hence, the overall gain in annotation prediction accuracy for the tree structured models is much higher than for the independent model. Concerning the different tree models, again the mixture-of-trees generally performs the best.

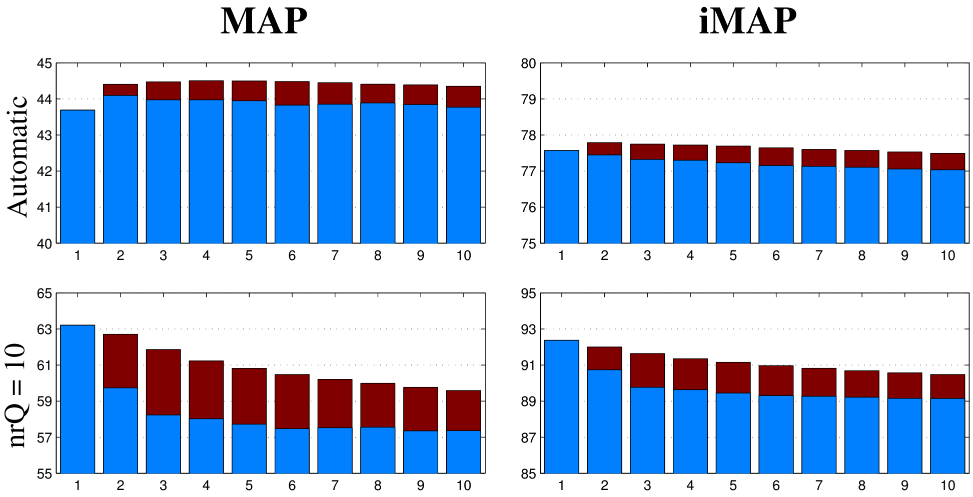


Figure 4.9 *Performance of different single label trees obtained by iterating the Chow-Liu algorithm (*blue*), and mixtures of the trees up to step t (*red*). We show results for the fully automatic setting (*top row*) and interactive setting with 10 questions (*bottom row*).*

4.7.2.3 Further analysis of the proposed models

In this section we further analyze some of the characteristics of our models including the tree structure, the unary potentials and the label elicitation strategies. All experiments are conducted on the *ImageClef'10* data set.

Selecting effective dependency structures The power of label predictions using structured models relies on the chosen dependency structure between the labels. Since both using a fully connected label dependency model, as well as obtaining the optimal tree structure for a discriminative model are intractable, we resort to approximate methods. In the experiments above, we have used tree structures obtained by the Chow-Liu algorithm and mixture-of-trees with multiple labels per node. In both cases the mutual information was exploited to compute the structure.

In this section, we conduct two further experiments with the aim of evaluating the effect of the selected tree structure on the annotation. In both cases, we use trees with a single label per node ($k = 1$).

In the first experiment we test the following hypothesis: *"the mixture-of-trees outperforms the individual tree models, only because it encodes multiple label dependencies"*. Therefore, we build several tree structures consecutively, by computing the maximum spanning tree (MST) over the mutual information matrix, such that each tree uses only edges, which were not used by any of the previous trees. The first tree we obtain in this way equals the optimal tree according to the Chow-Liu algorithm.

	MAP			iMAP		
	Auto	Q 5	Q 10	Auto	Q 5	Q 10
Gradient based	43.6	54.3	63.3	77.5	87.1	92.3
Chow-Liu algorithm	43.7	54.2	63.2	77.6	87.1	92.4

Table 4.1 *Comparison between different methods for obtaining a tree structure with a singe label per node.*

	Automatic				
	$k = 1$	$k = 2$	$k = 3$	$k = 4$	Mixt
Two-stage	43.3	44.0	44.6	44.8	44.9
Jointly-learned	41.6	41.6	42.0	40.4	41.8

	After 10 questions				
Two-stage	62.9	65.7	66.5	66.4	66.7
Jointly-learned	60.2	61.6	62.2	61.7	63.2

Table 4.2 *Performance in MAP of different learning approaches for the unary potentials, for trees with $1 \leq k \leq 4$ and the mixture of the four trees.*

In Figure 4.9, we show the performance of the individual trees and of the mixtures of these trees. For the mixture of step t, we make use of an equal weighting of the trees obtained in the first t steps. From this figure we see that in the fully automatic setting, a mixture of these trees can slightly improve the performance over the individual trees including the Chow-Liu tree. However, in the interactive setting we observe that the model using the Chow-Liu tree outperforms any of the other trees, or mixtures thereof, both on MAP and iMAP. Furthermore, comparing these results with those in Figure 4.7, it becomes clear that mixing different single node trees leads to a much lower improvement, than considering the mixture-of-trees with different group sizes k.

In the second experiment, we compare two different methods to build the tree, using a single label per node. The first method uses the Chow-Liu algorithm (as in previous experiments) and the second method builds a tree based on gradient information. To obtain the "gradient tree", we iteratively add edges based on the current gradients of the model. The results in Table 4.1 show that the MAP/iMAP performances of the two methods are very similar both for the fully automatic setting and for the interactive setting.

We conclude from these experiments that the structure obtained with the Chow-Liu algorithm – which gives the optimal tree for a generative model – and the mixture-of-trees with different number of labels per node are effective methods to obtain dependency structures for our model.

Joint learning of unary potentials In the experiments so far, we have followed a two-stage learning approach and used the pre-trained SVM classifier scores as features in the unary potentials. Joint learning of the unary and pairwise potentials might be more effective since the unary potentials can take into account the effect of the pairwise potentials. To test this, we set $\phi(\boldsymbol{x}_n)$ to be the concatenation of visual and textual features, yielding a high-dimensional vector, and use the kernel representation of Eq. (4.7) to optimize the regularized log-likelihood defined in Eq. (2.22). We vary the regularization parameter λ in the range $[10^{-6}, 10^{-2}]$, and report the best results. In this experiment, we have used only fold 1 of the *ImageClef'10* data set, instead of averaging over all folds. The reason is that the computational cost for this experiment is much higher, and since we have observed similar behavior on different folds, we expect that the results will generalize to the other folds as well.

From Table 4.2, we see that the joint learning of the unary potentials never matches the performance obtained with the pre-trained SVM classifier based unary potentials. This observation is consistent along all tested settings, the fully automatic and the interactive evaluation setting, as well as different label group sizes per node.

These results are in contrast with those of (Nowozin et al., 2010), where pre-training is shown to be competitive yet outperformed by joint learning. Note that our work differs from theirs in at least two important ways:

1. Our unary potentials for image labeling use global FV image representations, these are probably much stronger than their unary potentials for pixel-wise labeling, which use only local features.

2. Our pre-trained SVM scores resemble test-time prediction scores, since they are obtained in a cross-validation manner (see Section 4.7.1), while in (Nowozin et al., 2010) such a procedure is not followed.

We interpret these findings as an indication that in the presence of strong unary potentials, it is important to use unary scores that are representative of the test data scores.

Label elicitation strategy Here we show the benefit of the proposed label elicitation methods. Therefore we compare the performances of the independent model and the mixture-of-trees model using two different label elicitation strategies. The first strategy is the entropy based selection criteria, described in Section 4.5. The second is to randomly select labels, for which we report the mean performance over 10 evaluations using different randomly selected questions.

The results in Figure 4.10 show the performance of the independent predictors (*blue*) and our mixture model (*red*), from no user input to complete user-input. We can see that both models benefit more from the label entropy based elicitation mechanism compared to the

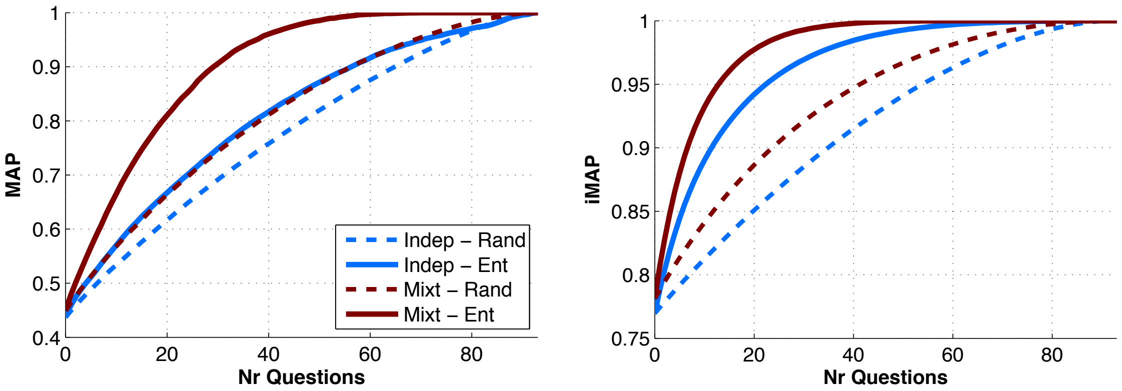


Figure 4.10 *Performance of MAP and iMAP as a function of the number labels set by the user on* ImageClef'10 *data set.*

randomly selected labels. Furthermore, we observe that our structured method achieves correct labeling after significantly fewer questions than the independent predictors.

4.7.2.4 Comparison to related work

In this section we compare our methods to state-of-the-art results obtained on the *ImageClef'10* and *SUN'09* data sets.

ImageClef 2010 photo annotation challenge Here, we compare to the participants of the ImageClef'10 VCDT task. For this experiment we use the training and test split as provided by the organizers of the challenge, and we report performance using the *interpolated* MAP, while in the rest of the chapter we report non-interpolated MAP.

In Table 4.3 we show the performance of our baseline independent models and TagProp, which we submitted to the challenge, we show the top performing results of other the participants in the ImageClef'10 challenge and we show the performances of our structured mixture-of-trees methods. For an overview of the challenge, including the participants, different methods and results, see (Nowak and Huiskes, 2010). In the table, we indicate for each model whether it uses only the visual modality (V) or uses both the visual and textual modalities (V&T).

Our baseline system was the winner of the challenge when using both modalities. In Section 2.2.2 we have compared TagProp and SVM classifiers on the same data set, and reported the influence of using the different feature types and modalities. In Table 4.3, we also show the performance of the mixture-of-trees models, employing a two-stage learning approach, when using both modalities (V&T) and when using only the visual modality (V). Again, we observe that the structured models outperform the independent models by about 1% MAP for both sets of features.

Team and method		Modality	MAP
SVM		V&T	**45.5**
TagProp		V&T	43.7
SVM		V	38.9
ISIS - MKL	(van de Sande and Gevers, 2010)	V	40.7
HHI	(Mbanya et al., 2010)	V	34.9
IJS	(Dimitrovski et al., 2010)	V	33.4
MEIJI	(Motohashi et al., 2010)	V&T	32.6
Mixture-of-trees		V&T	**46.7**
		V	40.0

Table 4.3 *Performance in MAP on the ImageClef 2010 VCDT Challenge.*

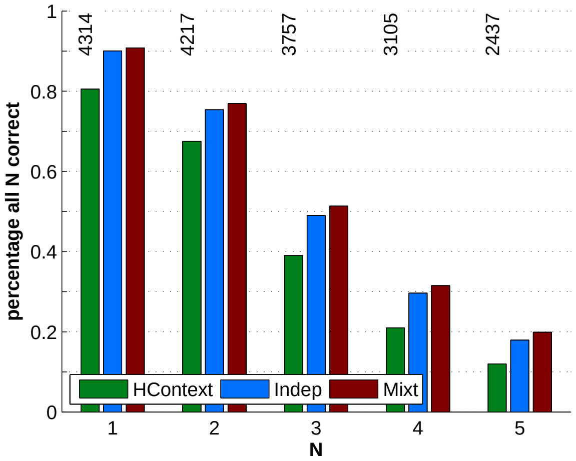


Figure 4.11 *Comparison of the performance of our independent and mixture-of-trees methods to the results of (Choi et al., 2010) on the* SUN'09 *data set.*

Comparison to the hierarchical context model In this section, we compare our method to the state-of-the-art results on the *SUN'09* obtained with the hierarchical context method (HContext) proposed in (Choi et al., 2010). For this comparison we used their evaluation method, which computes the percentage of images in which the top N predicted labels are all correct, taken over the images with at least N labels.

Results for our independent and structured models along with the results of Choi et al. (2010) are shown in Figure 4.11. We observe that our independent method clearly outperforms the HContext method, even in spite of the fact that the HContext model uses the object bounding boxes during training, while our independent method uses only global image labels. The performance difference can be partially explained by the stronger image representation (Fisher Vectors) we use compared to their Gist (Oliva and Torralba, 2001) features. This comparison shows the strength of our baseline method.

	Auto	1	2	3	4	5	6	7	8
Indep	38.1	55.5	71.0	79.9	86.1	91.1	95.3	97.7	99.6
Mixt	40.4	59.2	75.7	88.8	96.0	99.1	99.7	100.0	100.0

Table 4.4 *Zero-shot attribute-based classification accuracy of the independent and mixture-of-trees models. Initial results, and after user input for 1 up to 8 attributes.*

4.7.3 Attribute-based prediction of unseen classes

In this section we evaluation the performance of our attribute prediction models in the context of zero-shot classification. The *AwA* data set was introduced for transfer learning by means of sharing attributes used to represent different classes. We use the zero-shot prediction paradigm, where the test classes and training classes are disjoint. Hence, in this section we evaluate the performance of our structured models in predicting class labels of images from unseen classes based on the class specific configuration of the 85 attributes. To compare our approach to the state-of-the art, we use the same settings and the same evaluation measure (mean of the diagonal of the normalized confusion matrix) as in (Lampert et al., 2009).

For these experiments we use the independent model and the mixture-of-trees model with unary terms using the two-stage learning strategy, the trees obtained with the mutual-information criteria, and we use the suggested entropy based attribute elicitation for the interactive scenario. Table 4.4 shows the performance of the independent model[6] and our mixture-of-trees model.

Note that the tree-structured models learn attribute dependencies for the training classes which are different from the test classes, *i.e.* during testing we encounter combinations of attributes which we have never been seen before. Still, our model is able to take advantage of the learned attribute dependencies to significantly improve over the results of the independent model.

4.7.4 Multi-word query retrieval

In this section we evaluate our structured models for keyword based image retrieval. In the image classification tasks described in the previous sections we have used the MAP performance measure. This resembles in fact the evaluation of an image retrieval system where the query consists of single labels, since it computes the AP per label, and then takes the mean. In a general purpose image retrieval system however, users tend to

[6] Our baseline result of 38.1 is somewhat below the result of 40.5 reported in (Lampert et al., 2009). After several conversations with the authors, we conclude that this is probably due to the use of different class correction terms.

Query length	1	2	3	4	5
Number of queries	93	1,535	9,343	28,929	53,807
Independent model	43.7	26.7	21.2	19.1	18.3
Mixture-of-trees	44.9	27.8	22.2	20.0	19.2

Table 4.5 *Performance for text based image retrieval, in MAP, using multi-word queries.*

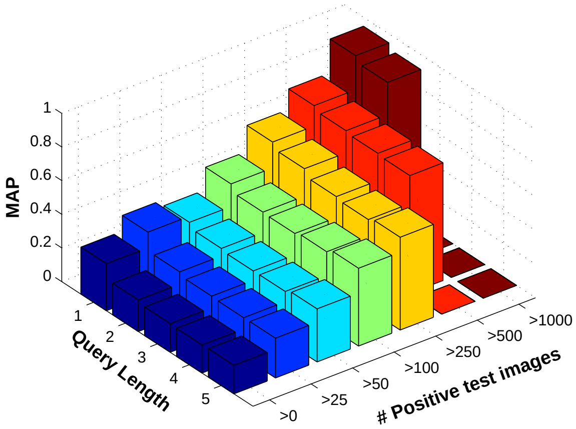


Figure 4.12 *Performance of multi-word query image retrieval, grouped by the query length and the number of positive images in the data set.*

use multi-word queries to find images or documents. Therefore, in this experiment we evaluate our proposed models for multi-word queries using the ImageClef'10 data set.

For this experiment we have created a query set containing all multi-word queries up to length 5, with at least 5 positive images in all of the test folds. A positive image means that all words from the query are relevant for this image according to the ground truth. This yields a query set of about 95.000 queries. All results are averaged over the 5 test folds of the ImageClef'10 data set.

Let $\{\boldsymbol{y}_q\}$ denote the set of labels for a specific query. For each query, we rank the images according to the likelihood $p(\{\boldsymbol{y}_q\}|\boldsymbol{x})$, *i.e.* the marginal that the query terms are relevant. For the tree model, we have:

$$p(\{\boldsymbol{y}_q\}|\boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x})} \sum_{\substack{\boldsymbol{y} \\ \text{s.t.}\forall i \in \{\boldsymbol{y}_q\}: y_i = 1}} \exp\left(E(\boldsymbol{y}, \boldsymbol{x})\right) = \frac{Z_q(\boldsymbol{x})}{Z(\boldsymbol{x})},$$

where $Z(\boldsymbol{x})$ is the partition function, and $Z_q(\boldsymbol{x})$ is the query-specific partition sum. The term $Z_q(\boldsymbol{x})$ can easily be computed using standard BP: it equals to the partition function while clamping all labels $\{\boldsymbol{y}_q\}$ to 1. For the mixture-of-trees model we use $p(\{\boldsymbol{y}_q\}|\boldsymbol{x}) = \sum_t \pi_t \, p_t(\{\boldsymbol{y}_q\}|\boldsymbol{x})$, and for the independent model we use:

$$p(\{\boldsymbol{y}_q\}|\boldsymbol{x}) = \prod_{i \in \{\boldsymbol{y}_q\}} p(y_i = 1|\boldsymbol{x}).$$

In Table 4.5 we compare the mixture-of-trees model to the independent model. We observe an improvement of about 1% in MAP when using the mixture-of-trees over the independent model, regardless of the query length.

In contrast to what Table 4.5 suggests, it seems that the difficulty of a query is mainly determined by the number of positive images available for that query in the data set, which in turn depends on the length of the query. To demonstrate this, in Figure 4.12 we show the performance in MAP as a function of the query length and the frequency of positive images in the test set. This figure shows that the MAP performance is much more influenced by the number of positive images available for the query, than the number of words in the query. Indeed, for short queries there tend to be many more positive images, so the overall performance is higher than for the longer queries. Furthermore, if we fix the query length, the performance increases when the number of positive images in the test set increases.

4.7.5 Learning to rank for p@k

In this section we present our experimental evaluation to learn the ranking of labels for an image. For this we use, again, the training set of the *ImageClef'10* data set. In the experiments we compare an independent model to a c-star model with 5 nodes in the core, both optimized for precision-at-k. The nodes selected for the core are chosen to optimize the mutual information of the core to the other labels, in a greedy fashion.

When learning p@k models, however, there is not a unique target prediction when an image does not have precisely k relevant labels. Therefore, in general there will be many sets of k labels that attain the minimum possible loss for a given image. To bypass this issue we use a loss which has a unique minimal solution. Such a loss is given by the precision-at-k, when k equals the number of relevant items *for this image*. This is also known as the *break-even precision* (BEP), which we use in our experiments.

In Table 4.6 we show the results of our models on BEP. We consider using a two-stage learning approach and compare them to a joint training method, similar as we did for the tree models. We observe that using the joint learning approach gives a small but marked improvement over using the two-stage approach, this contrasts to our findings for the tree structured models, and might be because the SVMs were trained for accuracy and not for BEP. We also see that the c-star structured models improve performance by about 1–1.5%.

To investigate the influence of the structured models more, we also evaluate them in an interactive image labeling setting. In this case we cannot use the entropy criterion for selecting the labels, but we use a related method adjusted for the max-margin framework. To select the label to present to the user, we compute for each label the maximum score that can be obtained when it is forced to be inside or outside the top k. The difference between the two scores relates to the confidence of the system that the label should be

	Independent				c-Star, $c = 5$			
	Auto	Q1	Q5	Q10	Auto	Q1	Q5	Q10
Two-stage learning	66.6	68.5	75.0	81.1	68.0	69.8	76.1	81.6
Joint learning	67.8	69.7	76.1	82.1	68.6	70.6	77.2	83.3

Table 4.6 *Break Even Point precision on* ImageClef'10 *for the independent and structured models. The performance is measured in a fully automatic setting versus an interactive setting after* $Q = \{1, 5, 10\}$ *questions.*

		Auto	Q1	Q5	Q10
Independent	Rank	67.8	69.7	76.1	82.1
Independent	Class	68.3	69.9	75.4	80.8
c-Star	Rank	68.6	70.6	77.2	83.3
Mixture-of-trees	Class	69.1	71.7	79.6	86.7

Table 4.7 *Comparison of BEP performance between the models optimized for the ranking measure (Rank) and the models optimized for classification accuracy (Class). The performance is shown for the automatic setting and the interactive setting.*

selected or not. Intuitively, if the two scores are far apart the model prefers a specific state of the label (low entropy), while if the scores are very close the model does not care (high entropy). For each image we select the label with the lowest score difference. The results of the interactive image labeling experiment are also shown in Table 4.6. It shows that the structured model consistently improves over the independent model.

In Table 4.7, we compare the independent and c-star models optimized for the BEP measure, with the independent and mixture-of-trees models optimized for classification accuracy, on the BEP measure. For the models optimized for classification we have used the two-stage training strategy, the trees based on mutual information, and the entropy-based elicitation strategy, while for the ranking models we have used the joint learning strategy.

From the results we observe that the structured models (c-star and mixture-of-trees) always outperform any of the independent models, and that the mixture-of-trees model outperforms the c-star model. This might be the result of the larger number of label-interactions which is encoded in the mixture-of-trees model, compared to the c-star model. For the independent models we observe that the model optimized for classification outperforms the ranking-model in the fully automatic setting. However, in the interactive scenario the results are reversed. A possible explanation is that the label elicitation strategy selects better labels to be set by the user.

4.8 Conclusion

In this chapter we have introduced several structured image labeling models to capture label dependencies.

First, we have defined a tree-based model and extended it to capture more label dependencies, by using multiple labels per node, and using mixtures of such models. We have explored different options for the unary potentials, and two methods to obtain the graphical structures. We have found that the best performance is obtained using a mixture-of-trees, where each tree has a different group size of k labels per node, and where the unary potentials are given by pre-trained SVM classifiers.

Second, we have considered an interactive scenario, where the system is allowed to ask a user to set the value of a small number of labels at test time. Such a scenario offers a trade-off between label accuracy and labeling effort. The proposed structured models are able to transfer user input to other image labels yielding more accurate predictions. This holds even more when the labels are selected following the entropy based criterion to reduce the remaining uncertainty on the other labels.

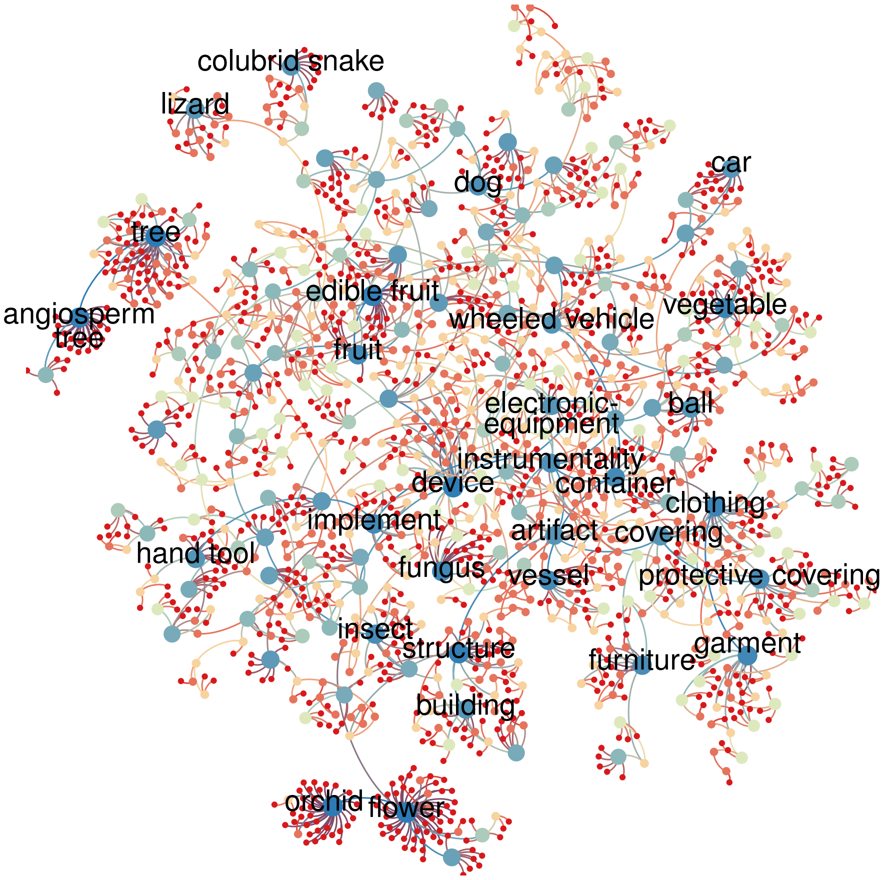
Third, besides the use for image labeling, we have also successfully applied the structured models for attribute-based classification. In this case the structured models are defined at the attribute-level, and the attribute predictions are used for classification. Also, for the attribute-based prediction we have used the interactive setting, and our structured models obtained higher accuracy than the independent model with a small amount of user input on the attribute level.

Finally, we have considered learning for the precision-at-k ranking cost function. We have shown the relation between structured ranking models, which models label dependencies and quadratic assignment problems. While quadratic assignment problems are in general NP hard, we have identified a subclass which is solvable in polynomial time. Using the c-star models, combined with a scorings function which shares the same invariances as the precision-at-k measure, the algorithm runs in $O(2^C(N + k \log k))$.

We have experimentally shown that the c-star models yield a modest but consistent improvement over the model assuming independence among the labels. However, the mixture-of-trees model, while optimized for classification accuracy, outperforms the c-star model. This might be the results of the larger number of label dependencies encoded in the mixture-of-trees model.

5

Large Scale Metric Learning for Distance-Based Image Classification



Visualization of a subset of the Image-Net hierarchy. The size of a node indicates its level in the hierarchy, for the 1,000 classes of the ILSVRC'10 data set only leave nodes are used.

Image courtesy of Zeynep Akata.

Contents

In this chapter we address the goal of large scale image classification, in a setting which allows for classification of classes not seen during training. We cast this problem into one of learning a low-rank metric, which is shared across all classes. For classification we explore k-nearest neighbor (k-NN) and nearest class mean (NCM) classifiers. Both methods can incorporate images from new classes at (near) zero cost: by adding them to the database (for k-NN), or by computing class means (for NCM). To recognize the new classes, the methods rely on a metric that was trained on other classes. This chapter is based on (Mensink et al., 2012a, Submitted2012).

5.1 Introduction

In this chapter we focus on the problem of large-scale image classification, which gained a lot of research interest since the introduction of the ImageNet data set (Deng et al., 2009). The predominant approach to this problem is to treat it as a classification problem. Recently, impressive results have been reported on 10,000 or more classes (Deng et al., 2010; Weston et al., 2011; Sánchez and Perronnin, 2011; Le et al., 2012). A drawback of these methods, however, is that when images of new categories become available, new classifiers have to be trained at a relatively high computational cost.

In contrast to these approaches, we consider methods which enable the addition of new classes and new images to existing classes at (near) zero cost. Such a method could be used continuously, while new images and classes become available, and additionally iterated from time to time with a (computationally heavier) method to update the metric using all available data to ensure best performance.

We explore two methods which allow for adding new images and classes on the fly.

1. The k-nearest neighbor (k-NN) classifier is a highly non-linear and non-parametric classifier that has shown competitive performance for image annotation (Deng et al., 2010; Weston et al., 2011; Guillaumin et al., 2009a). To incorporate new images (of new classes), they have to be added to the database, and can be used

for classification without further processing. Its main drawback is that the nearest neighbor search for classification is computationally demanding for large and high-dimensional data sets.

2. The nearest class mean classifier (NCM), represents classes by their mean feature vector, see *e.g.* (Webb, 2002). Contrary to the k-NN classifier, this is a linear classifier which leads to efficient classification. The cost of computing the mean can be neglected with respect to the cost of feature extraction and this operation does not require accessing images of other classes. The complete distribution of the training data of a class is, however, only characterized by its mean and it is therefore unclear whether this is sufficient for competitive performance.

The success of both methods critically depends on the metric which is used to compute the distance between an image and other images (for k-NN) or class means (for NCM). Therefore, we cast our classifier learning problem as one of learning a low-rank Mahalanobis distance which is shared across all classes. We explore several strategies for learning such a metric. For k-NN classification, we use the Large Margin Nearest Neighbor (LMNN) framework (Weinberger et al., 2006) and investigate two variations similar to the ideas presented in (Checkik et al., 2010; Weinberger and Saul, 2009) that significantly improve its classification performance. We also introduce an efficient gradient evaluation method, which allows for faster training. For the NCM classifier, we propose a novel metric learning algorithm based on multi-class logistic discrimination, where a sample from a class is enforced to be closer to its class mean than to any other class mean in the projected space.

Most of the experiments conducted in this chapter use the ImageNet Large Scale Visual Recognition Challenge 2010 (ILSVRC'10) data set. This data set contains over 1.2M training images of 1,000 classes. To apply the proposed metric learning techniques on this data set, we employ stochastic gradient descend (SGD) algorithms, which access only a small fraction of the training data at each iteration (Bottou, 2010). In addition, we use product quantization to fit the high-dimensional image features of the data sets in memory.

We compare the performance of the k-NN and NCM classifiers to a baseline approach. As baseline we use the state-of-the-art approach of (Sánchez and Perronnin, 2011), which uses Fisher Vector (FV) image representations and one-vs-rest linear SVM classifiers. For fair comparison, the k-NN and NCM classifiers use the same FV image representations.

We also show the generalization performance of the k-NN and NCM classifiers to new classes. In a first experiment, we train a metric on a subset of 800 classes of ILSVRC'10 and include the 200 held-out classes at test time. In a second experiment, we train a metric on the ILSVRC'10 data set and apply it to the larger ImageNet-10K data set, which consists of 4.5M training images of 10K classes (Deng et al., 2010). Once the metric is learned, we compute the 10K class means on 64K dimensional features in less than an hour on a single CPU, while learning one-vs-rest linear SVMs on the same data takes on the order of 280 CPU days.

Finally, we explore a zero-shot setting where we estimate the class means of novel classes based on related classes in the ImageNet hierarchy. We show that the zero-shot class mean can be effectively combined with the empirical mean of a small number of training images. This provides an approach that smoothly transitions from settings without training data to ones with abundant training data.

The rest of the chapter is organized as follows. In Section 5.2 we discuss a selection of related work which is most relevant to the topics of this chapter. In Section 5.3 we present metric learning techniques for k-NN classifiers. In Section 5.4 we introduce the NCM classifier together with an extension to use multiple means. We present extensive experimental results in Section 5.5, analyzing different aspects of the proposed methods and comparing them to the current state-of-the-art in different application settings such as large scale image annotation, transfer learning and image retrieval. Finally, in Section 5.6 we conclude this chapter.

5.2 Related work

In this section we discuss some of the most relevant work on large-scale image annotation, metric learning, and transfer learning.

Large-scale image annotation The ImageNet data set (Deng et al., 2009) has been a catalyst for research on large-scale image annotation. To ensure scalability, linear classifiers such as SVMs are often used (Sánchez and Perronnin, 2011; Lin et al., 2011). Additionally, to speed up classification, dimension reduction techniques could be used (Weston et al., 2011), or a hierarchy of classifiers could be learned (Bengio et al., 2011; Gao and Koller, 2011).

The current state-of-the-art uses efficient linear SVM classifiers trained in a one-vs-rest manner (Lin et al., 2011; Perronnin et al., 2012). Besides one-vs-rest training, large-scale ranking-based formulations have also been explored in (Weston et al., 2011). Interestingly, this approach performs joint classifier learning and dimensionality reduction of the image features. Operating in a lower-dimensional space acts as a regularization during learning, and also reduces the cost of prediction at test time. Our proposed NCM approach also learns low-dimensional projection matrices but the weight vectors are constrained to be the class means. This allows the efficient addition of novel classes.

In (Deng et al., 2010; Weston et al., 2011) k-NN classifiers were found to be competitive with linear SVM classifiers in a very large-scale setting involving 10,000 or more classes. The drawback of k-NN classifiers, however, is that they are expensive in storage and computation, since in principle all training data needs to be kept in memory and accessed to classify new images. The storage issue is also encountered when SVM classifiers are trained since all training data needs to be processed in multiple passes. To reduce the storage needs, we exploit PQ encoding to compress high-dimensional image signatures when learning our metrics.

Metric learning There is a large body of literature on metric learning, but here we limit ourselves to highlighting relevant methods that learn metrics for (image) classification problems. Other methods aim at learning metrics for verification problems and essentially learn binary classifiers that threshold the learned distance to decide whether two images belong to the same class or not *e.g.* (Nowak and Jurie, 2007; Guillaumin et al., 2009b). Metric learning is also used for text retrieval, where documents are described by high-dimensional but sparse feature vectors, and ranked according to their relevance for a given query *e.g.* (Bai et al., 2010).

Among the methods that learn metrics for classification, the Large Margin Nearest Neighbor (LMNN, Weinberger et al., 2006, see also Section 2.2.2) approach is specifically designed to support k-NN classification. It tries to ensure that for each image a predefined set of target neighbors from the same class is closer than samples from other classes. Usually the set of target neighbors is chosen and fixed using the ℓ_2 metric in the original space. This can be problematic since the ℓ_2 distance might be quite different than the optimal metric. Therefore, we explore two variants of LMNN that avoid using such a pre-defined set of target neighbors, similar to the ideas presented in (Weinberger and Saul, 2009; Checkik et al., 2010). Both variants lead to significant improvements. Since the LMNN objective decomposes over triplets of images, it can be sampled in an SGD training procedure, and therefore this method can scale to large data sets.

The large margin nearest local mean classifier (Chai et al., 2010) assigns a test image to a class based on the distance to the mean of its nearest neighbors in each class. This method was reported to outperform LMNN, but requires computing all pairwise distances between training instances and therefore does not scale well to large data sets. Similarly, TagProp (Guillaumin et al., 2009a) suffers from the same problem. It consists in assigning weights to training samples based on their distance to the test instance and in computing the class prediction by the total weight of samples of each class in a neighborhood. Because of their poor scaling properties, we do not consider the comparison to these methods in our experiments.

Closely related to our metric learning approach for the NCM classifier is the LESS model of (Veenman and Tax, 2005). They learn a diagonal scaling matrix to modify the ℓ_2 distance by rescaling the data dimensions, and include an ℓ_1 penalty on the weights to perform feature selection. In their case, NCM is used to address small sample size problems in binary classification, *i.e.* cases where there are fewer training points (tens to hundreds) than features (thousands). Our approach differs significantly in that (i) we work in a multi-class setting and (ii) we learn a low-dimensional projection matrix, which allows for reducing the dimensionality of our high-dimensional features.

The method of (Zhou et al., 2008) is also related to our method since they use a NCM classifier and an ℓ_2 distance in a subspace that is orthogonal to the subspace with maximum within-class variance. However, their technique involves computing the first eigenvectors of the within-class covariance matrix, which has a computational cost between $O(D^2)$ and $O(D^3)$, which again is undesirable for high-dimensional feature vectors. Moreover, this metric is heuristically obtained, rather than directly optimized for maximum classification performance.

Transfer learning The term transfer learning is used to refer to methods that share information across classes during learning. Examples of transfer learning in computer vision include the use of part-based or attribute class representations. Part-based object recognition models (Fei-Fei et al., 2006) define an object as a spatial constellation of parts, and share the part detectors across different classes. Attribute-based models (Lampert et al., 2009), which we have used in Chapter 4, characterize a category (*e.g.* a certain animal) by a combination of attributes (*e.g.* is yellow, has stripes, is a carnivore), and share the attribute classifiers across classes. Other approaches include biasing the weight vector learned for a new class towards the weight vectors of classes that have already been trained (Tommasi and Caputo, 2009). Zero-shot learning (Larochelle et al., 2008) is an extreme case of transfer learning where for a new class no training instances are available, but a description is provided in terms of parts, attributes, or other relations to already learned classes. In (Rohrbach et al., 2011) various transfer learning methods were evaluated in a large-scale setting using the ILSVRC'10 data set. They found transfer learning methods to have little added value when training images are available for all classes. In contrast, transfer learning was found to be effective in a zero-shot learning setting, where classifiers were trained for 800 classes, and the performance was tested in a 200-way classification across the held-out classes.

In this chapter we also aim at transfer learning, in the sense that we allow only a trivial amount of processing on the data of new classes, storing in a database, or averaging, and rely on a metric that was trained on other classes to recognize the new ones. In contrast to most work on transfer learning, we do not use any intermediate representation in terms of parts or attributes, nor do we train classifiers for the new classes. While also considering zero-shot learning, we further evaluate performance when combining a zero-shot model inspired by (Rohrbach et al., 2011) with progressively adding more training images per class, from one up to thousands. We find that the zero-shot model provides an effective prior when a small amount of training data is available.

5.3 Metric learning for k-NN classification

In this section we discuss metric learning for k-NN classifiers, where we follow and extend the approach of LMNN (Weinberger et al., 2006). We learn Mahalanobis distance functions of the form:

$$d_M(\boldsymbol{x}, \boldsymbol{x}') = (\boldsymbol{x} - \boldsymbol{x}')^\top M (\boldsymbol{x} - \boldsymbol{x}'), \tag{5.1}$$

where $\boldsymbol{x}$ and $\boldsymbol{x}'$ are D dimensional vectors, and M is a positive definite matrix. We focus on low-rank metrics with $M = W^\top W$ and $W \in \mathbb{R}^{d \times D}$, where $d \leq D$.

The Mahalanobis distance induced by W is equivalent to the ℓ_2 distance after linear projection of the feature vectors on the rows of W:

$$d_W(\boldsymbol{x}, \boldsymbol{x}') = \quad (\boldsymbol{x} - \boldsymbol{x}')^\top W^\top W (\boldsymbol{x} - \boldsymbol{x}') \quad = \quad ||W\boldsymbol{x} - W\boldsymbol{x}'||_2^2. \tag{5.2}$$

The dimensionality d of the low-rank matrix W is used as a regularization parameter, and to reduce the classification and storage cost compared to a full-rank Mahalanobis distance.

We do not consider using the more general formulation of $M = W^\top W + S$, where S is a diagonal matrix, as in (Bai et al., 2010). While this formulation requires only D additional parameters to estimate, it still requires computing distances in the original high-dimensional space. That is costly when using the dense and high-dimensional (4K-64K) Fisher Vectors we use to represent images.

5.3.1 Large margin nearest neighbor metric learning

The main objective of LMNN is to learn a metric, such that images from the same class are closer than images from other classes. We have discussed LMNN in Section 2.2.2, but for clarity we restate its main principles. The learning objective is based on triplets consisting of a query image q, a positive image p from the same class, and a negative image n from another class. The loss for such a triplet is given by:

$$L_{qpn} = \left[1 + d_W(\boldsymbol{x}_q, \boldsymbol{x}_p) - d_W(\boldsymbol{x}_q, \boldsymbol{x}_n)\right]_+. \tag{5.3}$$

The hinge-loss is zero if the negative image n is at least one distance unit further from the query q than the positive image p, and the loss is positive otherwise. The sub-gradient of the loss of a triplet is given by:

$$\nabla_W L_{qpn} = [\![L_{qpn} > 0]\!]\ 2\,W \left((\boldsymbol{x}_q - \boldsymbol{x}_p)(\boldsymbol{x}_q - \boldsymbol{x}_p)^\top - (\boldsymbol{x}_q - \boldsymbol{x}_n)(\boldsymbol{x}_q - \boldsymbol{x}_n)^\top \right). \tag{5.4}$$

We use the sum of the per-triplet loss as the final learning objective:

$$L = \sum_{q=1}^{N} L_q \tag{5.5}$$

$$L_q = \sum_{p \in P_q} \sum_{n \in N_q} L_{qpn}, \tag{5.6}$$

where P_q and N_q denote a predefined set of positive and negative images for the query $\boldsymbol{x}_q$.

In Weinberger et al. (2006) the set of targets P_q for a query q is set to the query's k-nearest neighbors from the same class in the original space. This implicitly assumes that the ℓ_2 distance in the original space is a good similarity measure, which is in practice not always the case. Therefore, we consider two alternative strategies:

1. The set of targets P_q is defined to contain all images of the same class as q. We refer to this method as 'All' in the experiments. This is similar to (Checkik et al., 2010) where the same type of loss was used to learn image similarity defined as the scalar product between images feature vectors after a learned linear projection.

2. The set of targets P_q is dynamically determined to contain the k images of the same class that are closest to q using the current metric. We refer to this method as 'Dynamic' in the experiments. This method corresponds to minimizing the loss function also with respect to the choice of P_q. Hence, different target neighbors can be selected than the ones obtained in the original space. A similar approach has been proposed in (Weinberger and Saul, 2009), where every T iterations P_q is redefined using target neighbors according to the current metric.

In the next section we propose an efficient gradient evaluation method, which allows to approximate the dynamic set of target neighbors at each iteration at a negligible additional cost compared to using a fixed set of target neighbors or using all images of the same class as target neighbors.

5.3.2 SGD training: triplet sampling and gradient calculation

The objective function of the k-NN metric learning approach described in the previous section is defined over triplets of images. Even if we disregard the set of target neighbors, each query image is paired with all images of other classes as a negative image. Thus, if we have N training images evenly distributed among C classes, this would yield already $N(N - \frac{N}{C})$ pairs. For the ILSVRC'10 data set, we have $N \approx 10^6$ and $C = 10^3$ which leads to roughly 10^{12} pairs, and to even more triplets.

Evaluating the exact gradient over all training triplets is computationally infeasible, therefore we rely on SGD training, where in each iteration we estimate the gradient using triplets from a limited set of $m \ll N$ images. Below, we detail how we can increase the efficiency of the SGD training by using an appropriate strategy to sample triplets of images, and by using an efficient algorithm for gradient evaluation.

Triplet sampling strategy In this section we describe a sampling strategy which generates about 4 million triplets by using only 300 images. Using a small number of m images per SGD iteration is advantageous since the cost of the gradient evaluation is in large part determined by (1) the cost of computing the projections $W\boldsymbol{x}$ for the m image signatures to the low dimensional space using projection matrix W, and (2) the cost of decompressing the PQ encoded signatures if these are used.

In our sampling strategy, we first select uniformly at random a class c from which we will sample query images. We then sample ρm images from class c, with $0 < \rho < 1$, and the remaining $(m - \rho m)$ images are uniformly sampled from the other classes. We can consider the number of triplets t we can generate as a function of ρ for a given 'budget' of m images to be accessed. In the case where P_q is set according to the 'All' strategy, the number of triplets t that can be generated for a specific ρ is given by:

$$t(\rho) = (\rho m)(\rho m - 1)(m - \rho m), \tag{5.7}$$

since we can pair the ρm images with the $\rho m - 1$ other images from the same class, and each pair forms a triplet with any of the $m - \rho m$ negative sampled images. The number of triplets t can be approximated by:

$$t(\rho) \approx m^3 \rho^2 (1 - \rho), \tag{5.8}$$

and hence, the number of triplets is maximized when we choose $\rho \approx \frac{2}{3}$. In our experiments, we use $\rho = \frac{2}{3}$ and $m = 300$ images per iteration, leading to about 4 million triplets per iteration.

Roughly the same number of triplets could also be generated by sampling two images for each of the $C = 1,000$ classes. In this case, there are two query images per class, forming a pair with the other positive image, and each pair can form a triplet with the $2(C - 1)$ images of other classes, leading to $4C(C - 1) \approx 4M$ triplets. In this manner, however, we would need to access $m = 2,000$ images, which is about 7 times more costly than using the described approach with $m = 300$.

When we use one of the other methods for P_q we do the following:

- For a fixed set of target neighbors, we still sample $\frac{m}{3}$ negative images, and take as many query images together with their target neighbors until we obtain $\frac{2m}{3}$ images allocated for the positive class.

- For a dynamic set of target neighbors we simply sample $\frac{2m}{3}$ positive and $\frac{m}{3}$ negative images, and select the dynamic target neighbors among the sampled positive images. Although approximate, this avoids computing the dynamic target neighbors among all the images in the positive class and still gives good results, see Section 5.5.

Efficient gradient evaluation Here, we introduce an efficient method to compute the gradient. By making use of sorting distances w.r.t. query images, it computes the gradient without explicitly iterating over all triplets, and it allows for approximating the closest neighbors when P_q is dynamically determined at a negligible additional cost.

By observing that the gradient Eq. (5.4) takes the form of outer products of the feature vectors, we can write the gradient in matrix notation as:

$$\nabla_W L = W X A X^\top, \tag{5.9}$$

where X contains as columns the feature vectors $\boldsymbol{x}$ of the m images used in a particular SGD iteration, and A is a coefficient matrix. This shows that, once A is available, the gradient can be computed in $O(m^2)$ time, even if a much larger number of triplets is used. First, we consider the case when P_q is set to all images of the same class.

A closer look at the gradient per query reveals that:

$$\begin{aligned}\nabla_W L_q = +2W \sum_p \Big(\sum_n [\![L_{qpn} > 0]\!] \Big) (\boldsymbol{x}_p \boldsymbol{x}_p^\top - \boldsymbol{x}_q \boldsymbol{x}_p^\top - \boldsymbol{x}_p \boldsymbol{x}_q^\top) \\ - 2W \sum_n \Big(\sum_p [\![L_{qpn} > 0]\!] \Big) (\boldsymbol{x}_n \boldsymbol{x}_n^\top - \boldsymbol{x}_q \boldsymbol{x}_n^\top - \boldsymbol{x}_n \boldsymbol{x}_q^\top).\end{aligned} \tag{5.10}$$

Therefore, the coefficient matrix A can be computed from the number of hinge-loss generating triplets in which each $p \in P_q$ and each $n \in N_q$ occurs:

$$A_{qn} = 2 \sum_p [\![L_{qpn} > 0]\!], \qquad A_{pq} = -2 \sum_n [\![L_{qpn} > 0]\!], \tag{5.11}$$

$$A_{qq} = \sum_p A_{qp} - \sum_n A_{qn}, \qquad A_{pp} = \sum_q A_{qp}, \qquad A_{nn} = \sum_q A_{qn}. \tag{5.12}$$

To compute the coefficients A_{qn} and A_{qp} we use the following algorithm:

1. Sort all distances w.r.t. query q in ascending order; to account for the offset in the hinge-loss use $d_W(\boldsymbol{x}_q, \boldsymbol{x}_p) + 1$ as distance, for each positive image.
2. Accumulate, from start to end, the number of negative images up to each position.

3. Accumulate, from end to start, the number of positive images after each position.
4. Read-off the number of hinge-loss generating triplets of image p or n w.r.t. query q.

The same algorithm can be applied when using a small set of fixed, or dynamic target neighbors. In particular, it allows the target neighbors to be determined dynamically at a negligible additional cost. Making use of the already sorted list, we can trivially identify the target neighbors. In this case only the selected target neighbors obtain non-zero coefficients, therefore in step 3 of the algorithm, we simply accumulate the number of target neighbors after each position, instead of the number of positive images.

The cost of this algorithm is $O(m \log m)$ per query, and thus $O(m^2 \log m)$ when using $O(m)$ query images per iteration. This is significantly faster than explicitly looping over all $O(m^3)$ triplets to check if they generate a hinge-loss.

Note that while this algorithm enables fast computation of the gradient of the hinge-loss, the value of the hinge-loss itself cannot be determined using this method. However, this is not problematic if an SGD approach is used for learning, since it only requires gradient evaluations, not function evaluations.

5.4 Metric learning for NCM classification

In this section we define our NCM classifier, where we use a multi-class logistic regression objective to learn a low-rank Mahalanobis distance between images and class means. We start with introducing the basic model, and describe it relation to existing models. Then we present an extension to use multiple centroids per class, which transforms the NCM into a non-linear classifier. Finally, we explore some variants of the objective which allow for smaller SGD batch sizes, and we give some insights in the critical points of the objective function.

5.4.1 Nearest class mean classifier

We formulate the NCM classifier using multi-class logistic regression, see Section 2.2.1, and define the probability for a class c given an image feature vector $\boldsymbol{x}$ as:

$$p(c|\boldsymbol{x}) = \frac{\exp\big(-d_W(\boldsymbol{x}, \boldsymbol{\mu}_c)\big)}{\sum_{c'=1}^{C} \exp\big(-d_W(\boldsymbol{x}, \boldsymbol{\mu}_{c'})\big)}, \tag{5.13}$$

where $\boldsymbol{\mu}_c$ is the mean of the feature vectors $\boldsymbol{x}_i$ from class $c \in \{1, \ldots, C\}$, and d_W represents the Mahalanobis distance function of Eq. (5.2). This definition may also be interpreted as giving the posterior class probabilities of a generative model where:

$$p(\boldsymbol{x}_i|c) = \mathcal{N}(\boldsymbol{x}_i; \mu_c, \Sigma), \tag{5.14}$$

is a Gaussian with mean $\boldsymbol{\mu}_c$, and class-independent covariance matrix Σ, which is set such that $\Sigma^{\text{-1}} = W^\top W$. The class probabilities $p(c)$ are set to be uniform over all classes.

To learn the projection matrix W, we minimize the negative log-likelihood on the ground-truth class labels $y_i \in \{1, \ldots, C\}$ on the training images:

$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^{N} \ln p(y_i | \boldsymbol{x}_i), \tag{5.15}$$

where we use implicit regularization, by the dimension d of the projection matrix W. The gradient of this objective function is easily verified to be:

$$\nabla_W \mathcal{L} = \frac{2}{N} \sum_{i=1}^{N} \sum_{c=1}^{C} \Big([\![y_i = c]\!] - p(c|\boldsymbol{x}_i) \Big) W (\boldsymbol{\mu}_c - \boldsymbol{x}_i)(\boldsymbol{\mu}_c - \boldsymbol{x}_i)^\top. \tag{5.16}$$

5.4.2 Relation to existing models

The NCM classifier is linear in $\boldsymbol{x}_i$ since we assign an image i to the class c^* with minimum distance:

$$\begin{aligned} c^* &= \operatorname*{argmin}_c \big\{ d_W(\boldsymbol{x}_i, \boldsymbol{\mu}_c) \big\}, && (5.17) \\ &= \operatorname*{argmin}_c \big\{ ||W\boldsymbol{x}_i||_2^2 + ||W\boldsymbol{\mu}_c||_2^2 - 2\boldsymbol{\mu}_c W^\top W \boldsymbol{x}_i \big\}, && (5.18) \\ &= \operatorname*{argmin}_c \big\{ b_c - \boldsymbol{w}_c^\top \boldsymbol{x}_i \big\}, && (5.19) \end{aligned}$$

where the class specific bias b_c and weight vector $\boldsymbol{w}_c$ are defined as:

$$\begin{aligned} b_c &= \ ||W\boldsymbol{\mu}_c||_2^2, && (5.20) \\ \boldsymbol{w}_c &= 2\boldsymbol{\mu}_c^\top (W^\top W). && (5.21) \end{aligned}$$

This observation allows us to relate the NCM classifier to other linear methods. For example, we obtain standard multi-class logistic regression if the restrictions on b_c and $\boldsymbol{w}_c$ are removed. This restriction, however, allows us to add new classes at near-zero cost, since the class specific parameters b_c and $\boldsymbol{w}_c$ are defined using only the class mean $\boldsymbol{\mu}_c$ and a class-independent metric W.

The NCM classifier is also closely related to the WSABIE method of (Weston et al., 2011). In the latter, a class c is scored using:

$$f_c(\boldsymbol{x}_i) = \boldsymbol{v}_c^\top W \boldsymbol{x}_i, \tag{5.22}$$

where $W \in \mathbb{R}^{d \times D}$ is also a low-rank projection matrix, and $\boldsymbol{v}_c$ is a class specific weight vector of dimensionality d, learned from the data. In NCM however, we enforce that $\boldsymbol{v}_c = W\boldsymbol{\mu}_c$, which allows us to add a class without learning $\boldsymbol{v}_c$ from newly labeled data.

The NCM classifier can also be related to the solution of ridge-regression, or regularized linear least-squares regression, which also uses a linear score function $f_c(\boldsymbol{x}_i) = b_c + \boldsymbol{w}_c^\top \boldsymbol{x}_i$. However the parameters b_c and $\boldsymbol{w}_c$ are learned by optimizing the squared loss:

$$L_{\text{RR}} = \frac{1}{N} \sum_i \Big(f_c(\boldsymbol{x}_i) - y_{ic} \Big)^2 + \lambda ||\boldsymbol{w}_c||_2^2, \tag{5.23}$$

where λ acts as regularizer, and where $y_{ic} = 1$, if image i belongs to class c, and $y_{ic} = 0$ otherwise. The ridge regression loss L_{RR} can be minimized in closed form and leads to:

$$b_c = \frac{n_c}{N}, \quad \text{and} \qquad \boldsymbol{w}_c = \frac{n_c}{N} \boldsymbol{\mu}_c^\top (\Sigma + \lambda I)^{-1}, \tag{5.24}$$

where Σ is the (class-independent) data covariance matrix, and n_c denotes the number of images in class c. Just like the NCM classifier, the ridge-regression solution also allows generalizing to new classes at near-zero cost. The class specific parameters can be found from the class mean $\boldsymbol{\mu}_c$ and class count n_c, once the data covariance matrix Σ has been estimated. Moreover, if n_c is equal for all classes, the score function becomes similar to our NCM classifier where we set W such that $W^\top W = (\Sigma + \lambda I)^{-1}$.

5.4.3 Non-linear classification using multiple centroids per class

In this section we extend the NCM classifier to allow for more flexible class representations and non-linear classification. The idea is to represent each class by a set of centroids, instead of only the class mean. Assume we have obtained a set of k centroids $\{\boldsymbol{m}_{cj}\}_{j=1}^k$ for each class c. We define the posterior probability for a centroid $\boldsymbol{m}_{cj}$ of class c as:

$$p(\boldsymbol{m}_{cj}|\boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x})} \exp\big(-d_W(\boldsymbol{x}, \boldsymbol{m}_{cj}) \big), \tag{5.25}$$

where the normalizer $Z(\boldsymbol{x})$ sums over all classes and all centroids:

$$Z(\boldsymbol{x}) = \sum_c \sum_j \exp\big(-d_W(\boldsymbol{x}, \boldsymbol{m}_{cj}) \big). \tag{5.26}$$

The posterior probability of a specific class c is then given by:

$$p(c|\boldsymbol{x}) = \sum_j p(\boldsymbol{m}_{cj}|\boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x})} \sum_j \exp\big(-d_W(\boldsymbol{x}, \boldsymbol{m}_{cj}) \big). \tag{5.27}$$

This model also corresponds to a generative model, where the probability for a feature vector $\boldsymbol{x}$, to be generated by class c, is given by a Mixture-of-Gaussians:

$$p(\boldsymbol{x}|c) = \sum_j \pi_{cj}\, \mathcal{N}(\boldsymbol{x}_i; \boldsymbol{m}_{cj}, \Sigma), \tag{5.28}$$

with equal mixing weights π_{cj}, and the covariance matrix Σ is shared among all classes. We refer to this method as the Nearest Class Multiple Centroids (NCMC) classifier.

To learn the projection matrix W, we resort once again on minimizing the negative log-likelihood, Eq. (5.15). The derivative of $\mathcal{L}$ w.r.t. W is given by:

$$\nabla_W \mathcal{L} = \sum_{i,c,j} \Big[p(\boldsymbol{m}_{cj}|\boldsymbol{x}_i) - p(\boldsymbol{m}_{cj}|\boldsymbol{x}_i, y_i, c) \Big]\, W(\boldsymbol{m}_{cj} - \boldsymbol{x}_i)(\boldsymbol{m}_{cj} - \boldsymbol{x}_i)^\top, \tag{5.29}$$

where

$$p(\boldsymbol{m}_{cj}|\boldsymbol{x}_i, y_i, c) = [\![c = y_i]\!]\, \frac{p(\boldsymbol{m}_{cj}|\boldsymbol{x}_i)}{\sum_{j'} p(\boldsymbol{m}_{cj'}|\boldsymbol{x}_i)}. \tag{5.30}$$

To obtain the centroids used in the NCMC classifier, we apply k-means clustering to each class separately, on the features $\boldsymbol{x}$ belonging to that class, using the ℓ_2 distance before projection on W. The NCM classifier is a special case of the NCMC classifier when $k = 1$. On the other hand, in the limit that each image in the training set is used as a class centroid, we obtain a weighted k-NN classifier which resembles TagProp.

Instead of using a fixed set of class means, it could be advantageous to iterate the k-means clustering and the learning of the projection matrix W. Such a strategy allows the set of class centroids to represent more precisely the distribution of the images in the projected space. Therefore, it might improve the classification performance, however the experimental validation of such a strategy falls beyond the scope of this chapter.

5.4.4 NCM objectives for small SGD batches

The gradients of the NCM learning objective sums over all classes, and for each class it computes the probability $p(c|\boldsymbol{x})$, according to Eq. (5.16). This is computationally expensive, since it requires the distance from an image to all means, in the normalizer $Z(\boldsymbol{x})$. Furthermore, each distance requires the class specific bias $b_c = ||W\boldsymbol{\mu}_c||_2^2$, which implies that each mean has to be projected on the metric W.

The cost of projecting m image vectors $\boldsymbol{x}$ and C class mean vectors $\boldsymbol{\mu}_c$ on the matrix $W \in \mathbb{R}^{d\times D}$ is $O\big((m + C)Dd\big)$, and the cost of computing the distances between the m images and the C class means after projection equals $O(mCd)$. Thus, the total complexity to compute the Mahalanobis distances is $O\big(md(D + C) + CDd\big)$.

Since only the first term scales with the number of images m used in an SGD iteration, it is not advantageous to use $m \ll C = 1,000$. In that case the cost would be dominated by the second term. This restricts the number of images m used in the SGD updates, and as a result each update is relatively expensive. Below, we consider two alternative objective functions that allow using smaller batch sizes.

First, we consider replacing the multi-class loss by a sum of binary one-versus-one losses:

$$\mathcal{L}_{\text{WW}} = \sum_i \sum_{c \neq y_i} - \ln \sigma\big(d_W(\boldsymbol{x}_i, \boldsymbol{\mu}_c) - d_W(\boldsymbol{x}_i, \boldsymbol{\mu}_{y_i})\big), \tag{5.31}$$

where the log-sigmoid is used to provide a smooth and differentiable loss, that encourages each class center to be further away from the sample than that of the ground-truth class. We note that this loss is similar to the ℓ_{WW} loss function used in the multi-class SVM of (Weston and Watkins, 1999), which we have discussed in Section 2.2.1, albeit we use the log-loss instead of the hinge-loss. The $\mathcal{L}_{\text{WW}}$ loss provides an upper-bound of the multi-class loss, which is tight if $\mathcal{L}_{\text{WW}}$ is zero.

We can also relate $\mathcal{L}_{\text{WW}}$ to objective functions used in ranking problems (Joachims, 2002; Grangier and Bengio, 2008) and LMNN (Weinberger et al., 2006), if we see the image vector $\boldsymbol{x}$ as a query and the objective is to rank classes according to their distances. The loss is then defined as a sum over triplets $(\boldsymbol{x}_i, y_i, c)$, where for each triplet a penalty is incurred if the non-relevant class c is ranked higher than the relevant class y_i according to the metric W. For example LMNN uses the hinge loss per triplet, *c.f.* Eq. (5.3).

The interest of the $\mathcal{L}_{\text{WW}}$ objective is that it decomposes additively over the classes. This allows to use per SGD iteration a single triplet $(\boldsymbol{x}_i, y_i, c)$, or a small number of $m \ll C$ triplets. The cost to compute the gradient of m triplets is $O(mDd)$. Unfortunately, we experimentally find that optimizing for the $\mathcal{L}_{\text{WW}}$ objective significantly decreases the tperformance, compared to the original multi-class logistic regression objective of Eq. (5.15).

Second, we consider replacing the Euclidean distance in Eq. (5.13) by the negative dot-product plus a class specific bias s_c. The probability for class c is then defined as:

$$p(c|\boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x})} \exp\left(2\, \boldsymbol{x}^\top W^\top W \boldsymbol{\mu}_c - s_c\right). \tag{5.32}$$

As learning objective we still use the multi-class logistic regression loss of Eq. (5.15). The efficiency gain stems from the fact that the norm of the projected mean, $b_c = ||W\mu_c||_2^2$ that appears in the Euclidean distance is replaced by the scalar bias s_c, that needs to be learned from the data.

Using the dot-product and the scalar biases, we can avoid projecting the mean-vectors on W, by twice projecting the sample vectors: $\hat{\boldsymbol{x}}_i = \boldsymbol{x}_i^\top W^\top W$, and computing the dot-products in high dimensional space $\langle \hat{\boldsymbol{x}}_i, \boldsymbol{\mu}_c \rangle$. For a batch of m images, the first step cost

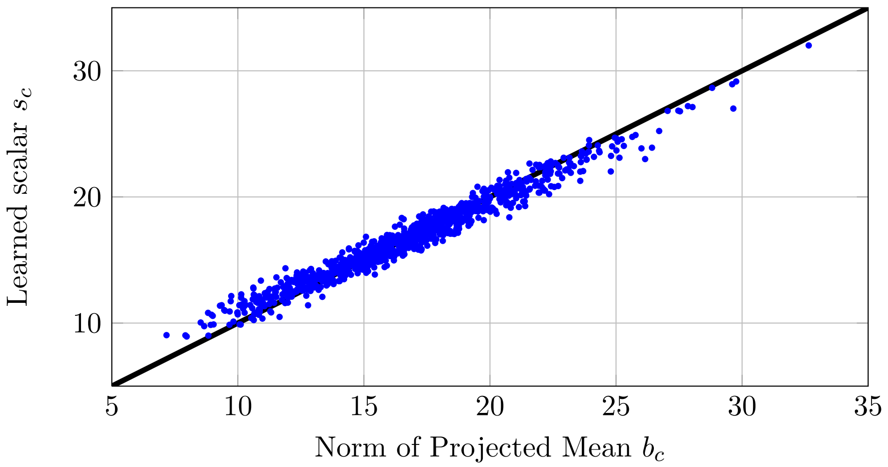


Figure 5.1 *A strong correlation is observed between class-specific bias s_c and the norm of the mean b_c computed after projection on W. The black line represents the identity.*

$O(mDd)$, and the latter cost $O(mCD)$. The total complexity for a batch using the dot-product formulation is therefore $O(mD(d+C))$ compared to $O(md(D+C)+CDd)$ for the Euclidean distance. Hence we observe, that the dot-product formulation is much more efficient when using small batches of $m \ll C$ images.

Experimentally we find that using this formulation yields comparable results to using the Euclidean distance. A potential disadvantage of this approach, however, is that we need to determine the class-specific bias s_c when data of a new class becomes available, which would require more training than just computing the data mean for the new class. Interestingly, we find a strong correlation between the learned bias s_c and the norm of the projected mean b_c, shown in Figure 5.1. Indeed, the classification performance differs insignificantly if at evaluation time we set $s_c = ||W\mu_c||_2^2 = b_c$ instead of the value that was found during training. Thus, even though we train the metric by using class-specific biases, we can use the learned metric in the NCM classifier where we replace the bias with the norm of the projected mean. This bias is then easily computed for new classes.

5.4.5 Critical points of low rank approximation

We use a low-rank approximation of the Mahalanobis distance, where $M = W^\top W$, as a way to reduce the number of parameters and to gain computational efficiency. Learning a full Mahalanobis distance matrix M, however, has the advantage that the distance is linear in the parameters M and that the multi-class logistic regression objective of Eq. (5.15) is therefore convex in M. Using a low-rank formulation, on the other hand, yields a distance which is quadratic in the parameters W, and therefore the objective function is no longer convex. In this section we investigate the critical-points of the low-rank formulation by analyzing W when the optimization reaches a (local) minimum, and considering the gradient for the corresponding full matrix $M = W^\top W$.

The gradients of the NCM objective w.r.t. to M and W, respectively, can be written as:

$$\nabla_M \mathcal{L} = \frac{1}{N} \sum_{i,c} \alpha_{ic} \, \boldsymbol{z}_{ic} \boldsymbol{z}_{ic}^\top \equiv H, \tag{5.33}$$

$$\nabla_W \mathcal{L} = \frac{2}{N} \sum_{i,c} \alpha_{ic} \, W \boldsymbol{z}_{ic} \boldsymbol{z}_{ic}^\top \equiv 2WH, \tag{5.34}$$

where we use $\alpha_{ic} = [\![y_i = c]\!] - p(c|\boldsymbol{x}_i)$, and $\boldsymbol{z}_{ic} = \boldsymbol{\mu}_c - \boldsymbol{x}_i$. From the gradient w.r.t. W we immediately observe that $W = 0$ leads to a degenerate case to obtain a zero gradient and the same applies separately per row of W.

Here, we concentrate on the non-degenerate case. We observe that H is a symmetric matrix, containing the difference of two positive definite matrices. In the analysis we use the eigenvalue decomposition of $H = V \Lambda V^\top$, with the columns of V being the eigenvectors, and the eigenvalues are on the diagonal of Λ.

We can now express the gradient for W as:

$$\nabla_W \mathcal{L} = 2WV\Lambda V^\top. \tag{5.35}$$

Thus the gradient of the i-th row of W, which we denote by $\boldsymbol{g}_i$, is given by a linear combination of the eigenvectors of H:

$$\boldsymbol{g}_i \equiv 2 \sum_j \lambda_j \langle \boldsymbol{w}_i, \boldsymbol{v}_j \rangle \boldsymbol{v}_j, \tag{5.36}$$

where $\boldsymbol{w}_i$ and $\boldsymbol{v}_j$ denote the i-th row of W and the j-th column of V respectively. Thus an SGD gradient update will drive a row of W towards the eigenvectors of H that (1) have a large positive eigenvalue, and (2) are most aligned with that row of W. This is intuitive, since we would expect the low-rank formulation to focus on the most significant directions of the full-rank metric.

Moreover, the expression for the gradient in Eq. (5.36) shows that at a critical point W^* of the objective function, all linear combination coefficients are zero:

$$\forall_{i,j} : \lambda_j \langle \boldsymbol{w}_i^*, \boldsymbol{v}_j \rangle = 0.$$

This indicates that at the critical point, for each row $\boldsymbol{w}_i^*$ and each eigenvector $\boldsymbol{v}_j$ it holds that either $\boldsymbol{w}_i^*$ is orthogonal to $\boldsymbol{v}_j$, or that $\boldsymbol{v}_j$ has a zero associated eigenvalue, *i.e.* $\lambda_j = 0$. Thus, at a critical point W^*, the corresponding gradient for the full rank formulation at that point, with $M^* = W^{*\top} W^*$, is zero in the subspace spanned by W^*.

Given this analysis, we believe that it is unlikely to attain poor local minima using the low rank formulation: the gradient updates for W are aligned with the most important directions of the corresponding full-rank gradient, and at convergence the full-rank gradient is zero in the subspace spanned by W. To confirm our intuition, we have experimentally investigated this by training several times when starting from different random initializations of W, and using different random samplings in each SGD iteration. We observe that the differences in classification performance on the converged metrics are at most $\pm 0.1\%$ on any of the error measures used, and that the number of SGD iterations selected by the early stopping procedure are of the same order.

5.5 Experimental evaluation

In this section we experimentally validate our models described in the previous sections. We first describe the data set and evaluation measures used in our experiments, followed by the presentation of our results for k-NN classification and NCM classification using metric learning.

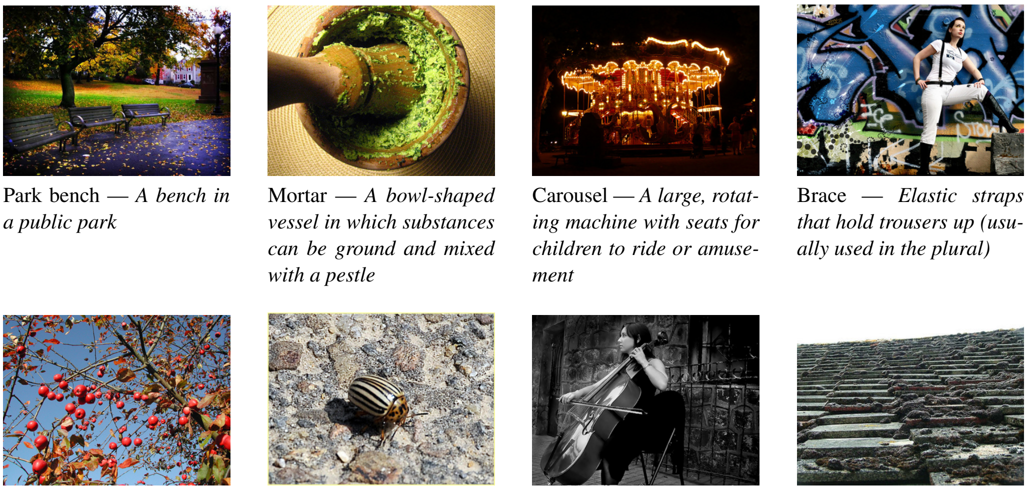
5.5.1 Experimental setup and baseline approach

Dataset In most of our experiments we use the data set of the ImageNet Large Scale Visual Recognition 2010 challenge (ILSVRC'10)[1]. This data set contains 1.2M training images of 1,000 object classes (with between 660 to 3047 images per class), a validation set of 50K images (50 per class), and a test set of 150K images (150 per class), see Figure 5.2 for some example images and classes.

In some of the experiments we use the ImageNet-10K data set, introduced in (Deng et al., 2010). This data set consists of 10,184 classes, obtained as the nodes of the ImageNet hierarchy which contained more than 200 images each. We follow (Sánchez and Perronnin, 2011) and use 4.5M images as a training set, 50K as a validation set and the other ± 4.5M images as a test set.

Features We represent each image with a Fisher Vector (FV) computed over densely extracted 128 dimensional SIFT and 96 dimensional LCS descriptors, both projected to 64 dimensions with PCA. FVs are extracted and normalized separately for both channels and then combined by concatenating the two feature vectors. We do not make use of spatial pyramids. In our experiments we use FVs extracted using a vocabulary of either 16 or 256 Gaussians. For 16 Gaussians, this leads to a 4K dimensional feature vector, which

[1] See http://www.image-net.org/challenges/LSVRC/2010/index



Park bench — *A bench in a public park*

Mortar — *A bowl-shaped vessel in which substances can be ground and mixed with a pestle*

Carousel — *A large, rotating machine with seats for children to ride or amusement*

Brace — *Elastic straps that hold trousers up (usually used in the plural)*

Crab apple — *Small sour apple; suitable for preserving; "crabapples make a tangy jelly"*

Leaf beetle — *Brightly colored beetle that feeds on plant leaves; larvae infest roots and stems*

Violoncello — *A large stringed instrument; seated player holds it upright while playing*

Tile roof — *A roof made of fired clay tiles*

Figure 5.2 *Example of images from the ILSVRC'10 data set, with their associated class and class descriptions.*

requires about 20GB for the 1.2M training set (using 4-byte floating point arithmetic). This fits into the RAM of our 32GB servers.

For 256 Gaussians, the FVs are 16 times larger, *i.e.* 64K dimensional, which would require 320GB of memory. Hence, we compress the feature vectors using product quantization. In each iteration of the SGD learning, we decompress the features of a limited number of images, and use these (lossy) reconstructions for the gradient computation.

Evaluation measures We report the average top-1 and top-5 flat error used in the ILSVRC'10 challenge. The flat error is one if the ground-truth label does not correspond to the top-k labels with highest score, and zero otherwise. The idea for using the top-5 error instead of the top-1 error, is to allow an algorithm to identify multiple objects in an image and not being penalized if one of the objects identified was in fact present, but not included in the ground truth (since images are annotated with only a single object/topic).

Unless specified otherwise, we report the top-5 flat error on the test set; we use the validation set for parameter tuning only. In tables, we highlight the best result per row or group in bold, and do so for each feature set if several are used. Additionally, the baseline performance is also highlighted if it is best.

Baseline approach For our baseline, we follow the state-of-the-art approach of (Perronnin et al., 2012) and learn weighed one-vs-rest SVMs with SGD, where the number of negative images in each iteration is sampled proportionally to the number of positive images for that class. The results of the baseline can be found in Table 5.2 and Table 5.6. We see that the 64K dimensional features lead to significantly better results than the 4K ones, despite the lossy PQ compression.

In Table 5.2, the performance using the 64K features is slightly better than the ILSVRC'10 challenge winners (Lin et al., 2011) (28.0 vs. 28.2 flat top-5 error), and close to the results of (Sánchez and Perronnin, 2011) (25.7 flat top-5 error), wherein an over 1M dimensional image representation was used. In Table 5.6 our baseline shows state-of-the-art performance on ImageNet-10K when using the 64K features, obtaining 78.1 vs 81.9 flat top-1 error (Perronnin et al., 2012).

SGD training and early stopping To learn the projection matrix W, we use SGD training and sample at each iteration a fixed number of m training images to estimate the gradient. Following (Bai et al., 2010) , we use a fixed learning rate and do not include an explicit regularization term, but rather use the projection dimension $d \leq D$, as well as the number of iterations as an implicit form of regularization. For all experiments we use the following early stopping strategy:

1. SGD training is run for a large number of iterations ($\approx$ 750K-2M),
2. the performance on the validation set is evaluated every 50K iterations (for k-NN) or every 10K iterations (for NCM), and
3. the metric which yields the lowest top-5 error is selected.

If on par, the metric giving the lowest top-1 error is chosen. Similarly, all hyper-parameters, like the value of k for the k-NN classifiers, are validated in this way. Unless stated otherwise, training is performed using the ILSVRC'10 training set, and validation using the described early stopping strategy on the provided 50K validation set.

It is interesting to notice that while the compared methods (k-NN, NCM, and SVM) have different computational complexities, the number of images seen by each algorithm before convergence is of the same order of magnitude. For example, training of the SVMs, on the 4K features, converge after $T \approx 100$ iterations, and each iteration takes about 64 negative images per positive image, per class. In the ILSVRC'10 data set, each class has roughly $p = 1,200$ positive images, and consists of $C = 1,000$ classes. Therefore the total number of images seen during training of the SVMs is $TC(65p) = 7.800$M images. The NCM classifier requires much more iterations, $T \approx$ 500K, but uses at each iteration only $m = 1,000$ images, and trains only a single metric. Therefore the total number of images seen during training is roughly $Tm =$ 500M. And finally, the k-NN classifier, requires even more iterations, $T \approx$ 2M, but uses only $m = 300$ images per iteration. The total number of images seen before convergence is therefore $Tm =$ 600M.

		k-NN classifiers						
	SVM	ℓ_2	ℓ_2	Fixed		All	Dynamic	
	Full	Full	+ PCA	10	20		10	20
Flat top-1 error	**60.2**	75.4	77.2	72.9	72.8	67.9	**65.2**	66.0
Flat top-5 error	**38.2**	55.7	57.3	50.6	50.4	44.2	**39.7**	40.7

Table 5.1 *Comparison of different k-NN classification methods 4K dimensional features. For all methods, except those indicated by 'Full', the data is projected to a 128 dimensional space.*

5.5.2 k-NN metric learning results

We start with an assessment of k-NN classifiers using metrics learned with the methods described in Section 5.3, and consider the impact of the different choices for the set of target images P_q, and the projection dimensionality. Given the cost of k-NN classifiers, we focus our experiments on the 4K dimensional features. We initialize W as a PCA projection, and determine the number of nearest neighbors to use for classification on the validation set; typically 100 to 250 neighbors are optimal.

Target selection for k-NN metric learning. In the first experiment we compare the three different options of Section 5.3 to define the set of target images P_q, while learning projections to 128 dimensions using the LMNN method. For fixed and dynamic targets, we experimented with various numbers of targets on the validation set and found that using 10 to 20 targets yields the best results.

The results in Table 5.1 show that all methods lead to metrics that are better than the ℓ_2 metric in the original space, or after a PCA projection to 128 dimensions. Furthermore, we can improve over using a fixed set of neighbors by using all within-class images as targets, and even further by using dynamic targets. The success of the dynamic target selection can be explained by the fact that among the three alternatives, it is the most closely related to the k-NN classifier objective. The best performance on the flat top-5 error of 39.7 using 10 dynamic targets is, however, slightly worse than the 38.2 error rate of the SVM baseline.

Impact of projection dimension on k-NN classification. Next, we evaluate the influence of the projection dimensionality d on the performance, by varying d between 32 and 1024. We only show results using 10 dynamic targets, since this performed best among the evaluated k-NN methods. From the results in Table 5.2 we see that a projection to 256 dimensions yields the lowest error of 39.0, which still remains somewhat inferior to the SVM baseline.

	4K features							64K features			
Projection dim.	32	64	128	256	512	1024	Full	128	256	512	Full
SVM baseline							38.2				**28.0**
k-NN, dynamic 10	47.2	42.2	39.7	**39.0**	39.4	42.4					
NCM, learned metric	49.1	42.7	39.0	37.4	**37.0**	**37.0**		31.7	31.0	**30.7**	
NCM, PCA + ℓ_2	78.7	74.6	71.7	69.9	68.8	68.2	**68.0**				**63.2**
NCM, PCA + inv. cov.	75.5	67.7	60.6	54.5	49.3	46.1	**43.8**				
Ridge-regression, PCA	86.3	80.3	73.9	68.1	62.8	58.9	**54.6**				
WSABIE	51.9	45.1	41.2	39.4	38.7	**38.5**		32.2	30.1	**29.2**	

Table 5.2 *Performance of k-NN and NCM classifiers, as well as baselines, using the 4K and 64K dimensional features, for various projection dimensions, and comparison to related methods, see text for details.*

5.5.3 Nearest class mean classification results

We now consider the performance of NCM classifiers and the related methods described in Section 5.4. In Table 5.2 we show the results for various projection dimensionalities.

We first consider the results for the 4K dimensional features. Our first, unexpected, observation is that our NCM classifier (37.0) outperforms the more flexible k-NN classifier (39.0), and even slightly outperforms the SVM baseline (38.2) when projecting to 256 dimensions or more. Interestingly, using just the ℓ_2 distance, instead of a learned metric, the situation is reversed and the k-NN classifier is better (55.7, see Table 5.1) than the NCM classifier (68.0).

Our implementation of WSABIE (Weston et al., 2011) scores slightly worse (38.5) than the baseline and our NCM classifier. Ridge-regression leads to significantly worse results (54.6) and pre-processing the data with PCA further degrades its performance.

We further consider two variants of the NCM classifier where we use PCA to reduce the dimensionality. Fist, we use the ℓ_2 metric after PCA, and observe that, just like for k-NN, the ℓ_2 metric with or without PCA leads to poor results (68.0). Second, inspired by ridge-regression, we use NCM with the metric W generated by the inverse of the regularized covariance matrix, such that $W^\top W = \Sigma + \lambda I^{-1}$, see Section 5.4.2. We tuned the regularization parameter λ on the validation set, as was also done for ridge-regression. This leads to results (43.8) that are better than using the ℓ_2 metric, and also substantially better than ridge-regression (54.6). The results are however significantly worse than using our learned metric, in particular when using low-dimensional projections.

For the 64K dimensional features, we observe that the results of the NCM classifier (30.7) are somewhat worse than the SVM baseline (28.0), but significantly better than using ℓ_2 distance (63.2). WSABIE obtains an error of 29.2, in between the SVM and NCM.

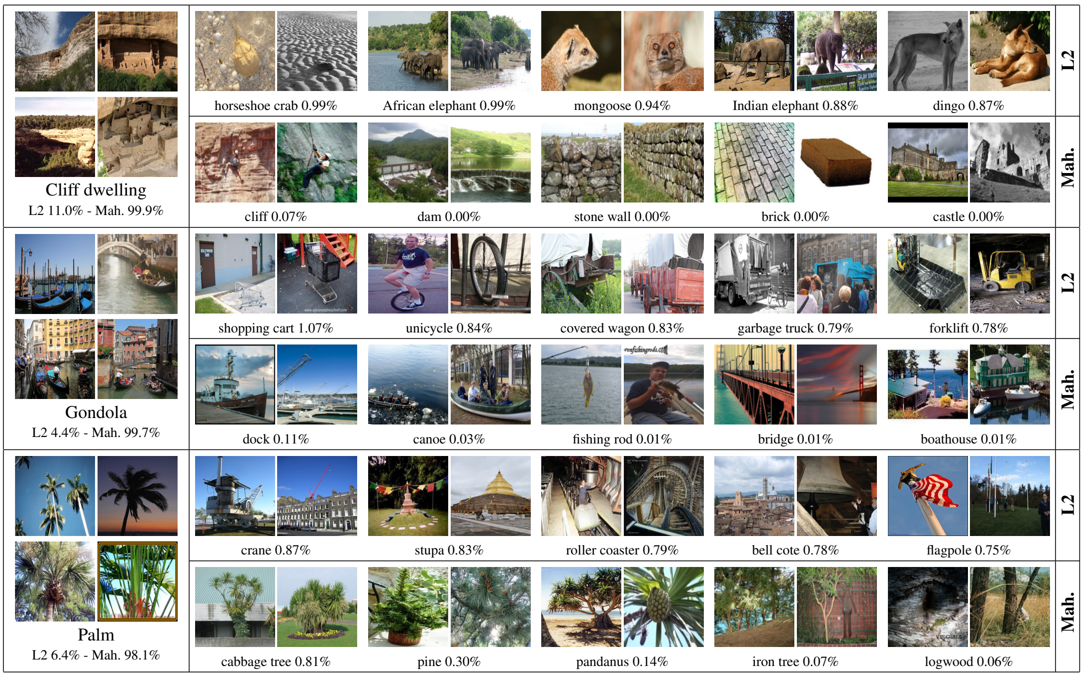


Figure 5.3 *The nearest classes for three reference classes using the the ℓ_2 distance and Mahalanobis metric learned for the NCM classifier. Class probabilities are given for a simulated image signature that equals the mean of the reference class, see text for details.*

Note that the SVM classifiers and WSABIE do not allow generalization to new classes without training, in contrast to k-NN, NCM and ridge-regression, where new classes can be added at near-zero cost.

Influence of metric learning on semantic class neighbors. In Figure 5.3 we illustrate the difference between the ℓ_2 and the Mahalanobis metric induced by a learned projection from 64K to 512 dimensions. For three reference classes we show the five nearest classes, based on the distance between class means. We also show the probabilities on the reference class and its five neighbor classes according to Eq. (5.13). The feature vector $\boldsymbol{x}$ is set as the mean of the reference class, *i.e.* a simulated perfectly typical image of this class. For the ℓ_2 metric, we used our metric learning algorithm to learn a scaling of the ℓ_2 metric to minimize Eq. (5.15). This does not change the ordering of classes, but ensures that we can compare probabilities computed using both metrics. We find that, as expected, the learned metric has more semantically related class neighbors. Moreover, we see that by using the learned metric, most of the probability mass is assigned to the reference class, whereas the ℓ_2 metric leads to rather uncertain classifications.

Non-linear classification using multiple class centroids. In these experiments we use the non-linear NCMC classifier, introduced in Section 5.4.3, where each class is represented by a set of k centroids. We obtain the k centroids per class by using the k-means

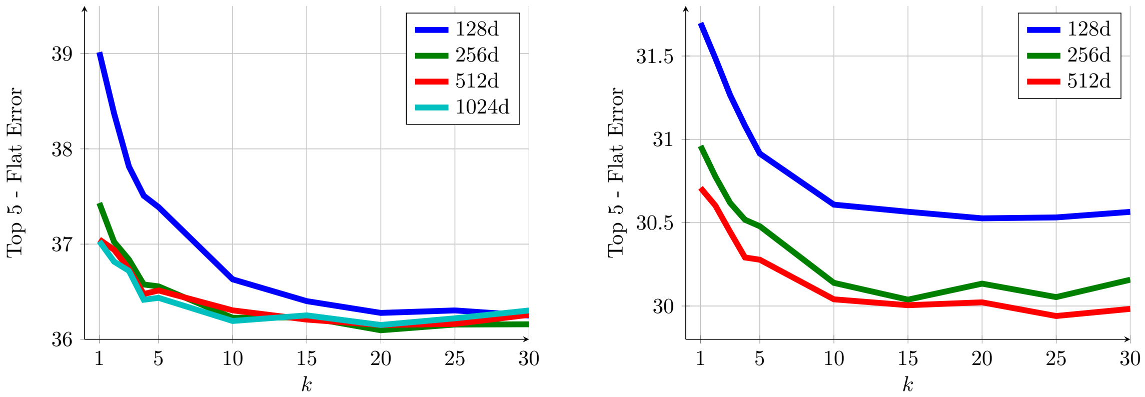


Figure 5.4 *The top-5 performance of the NCMC-test classifier, for the 4K (*left*) and 64K (*right*) features. NCMC-test uses at test time $k > 1$ on metrics obtained with NCM.*

	NCM	NCMC-test		NCMC		
Proj. dim.			*(k)*	5	10	15
128	39.0	36.3	*(30)*	36.2	**35.8**	36.1
256	37.4	36.1	*(20)*	35.0	**34.8**	35.3
512	37.0	36.2	*(20)*	34.8	**34.6**	35.1

Table 5.3 *The top-5 performance of the NCMC classifier using the 4K features, compared to the NCM baseline and the best NCMC-test classifier (with the value of k in brackets).*

algorithm using the ℓ_2 metric in the original space. Since the cost of training these classifiers is k times higher, we run two sets of experiments.

In the first set of experiments, we evaluate using the NCMC classifier at test time with $k = [2, \ldots, 30]$, while using a metric obtained by the NCM learning objective. For each value of k the early stopping strategy is used to determine the best metric. We refer to this method as NCMC-test.

In Figure 5.4, we show the performance of NCMC-test for the 4K and 64K features. From the results we observe that a significant performance improvement can be obtained by using the non-linear NCMC classifier. For both features and all projection dimensionalities the NCMC-test approach outperforms NCM, especially when using $d = 128$ the performance increase is significant.

In the second set of experiments we only use the 4K features and train for the NCMC objective with $k \in \{5, 10, 15\}$. In Table 5.3, we show the performance and compare the results to the NCM method and the best NCMC-test method. From the results we observe that when learning using the NCMC objective we can further improve the performance of the non-linear classification, albeit for a higher training cost. When using as little as 512 projection dimensions, we obtain the very impressive performance of 34.6 on the top-5

	Trained on all classes					Trained on 800 classes		
	4K features			64K features		4K features		64K
Dim	SVM	k-NN	NCM	SVM	NCM	k-NN	NCM	NCM
Full / ℓ_2	37.6			**27.7**		54.2	66.6	61.9
128		39.0	38.6		31.7	42.2	42.5	39.9
256		38.4	36.8		30.8	42.4	40.4	**37.8**
512			**36.4**		30.6		39.9	**37.8**
1024			36.5				**39.6**	

Table 5.4 *Performance of 1,000-way classification among test images of 200 classes not used for metric learning, and control setting with metric learning using all classes. The results are compared to the SVM baselines trained on all classes and to the NCM classifier when using the ℓ_2 distance.*

error, with $k = 10$ centroids. That is an improvement of about 2.4 absolute points over the NCM classifier (37.0), and 3.6 absolute points over SVM classification (38.2).

5.5.4 Generalization to new classes and using few samples

Given the encouraging classification accuracy of the NCM and k-NN classifiers observed above, we now explore their ability to generalize to new classes in several experiments.

Generalization to novel classes not seen during training. In this set of experiments we use approximately 1M images corresponding to 800 random classes to learn metrics, and evaluate the generalization performance on 200 held-out classes. The error is evaluated in a 1,000-way classification task, and computed over the 30K images in the test set of the held-out classes. The early stopping strategy uses the validation set of the 200 unseen classes. Performance among test images of the 800 training classes changes only marginally and would obscure the changes among the test images of the 200 held-out classes.

In Table 5.4 we show the performance of NCM and k-NN classifiers for several projection dimensions, and compare it to the control setting where the metric is trained on all 1,000 classes. The results show that both classifiers generalize remarkably well to new classes. For comparison we also include the results of the SVM baseline, and of the k-NN and NCM classifiers when using the ℓ_2 distance, evaluated over the 200 held-out classes. In particular for 1024 dimensional projections of the 4K features, the NCM classifier achieves an error of 39.6 over classes not seen during training, as compared to 36.5 when using all classes for training. For the 64K dimensional features, the drop in performance is larger, but it is still surprisingly good considering that training for the novel classes consists only in computing their means.

	4K features					64K features			
Proj. dim.	128	256	512	1024	ℓ_2	128	256	512	ℓ_2
Top-1 error	91.8	90.6	90.5	**90.4**	95.5	87.1	86.3	**86.1**	93.6
Top-5 error	80.7	78.7	**78.6**	**78.6**	89.0	71.7	70.5	**70.1**	85.4

Table 5.5 *Performance of the NCM classifier on the ImageNet-10K data set, using metrics learned on the ILSVRC'10 data set, and comparison to NCM using ℓ_2 distance.*

Method	NCM	SVM baseline		[1]	[2]	[3]	[4]
Dim.	64K-512	4K	64K	21K	128K	131K	
Top-1 error	86.1	86.0	**78.1**	93.6	81.9	83.3	80.8

Table 5.6 *Comparison of performance on the ImageNet-10K data set. Showing the best performing NCM (64K features projected to 512 dimensions), the baseline SVMs, and previously reported results: [1] SVM results with 21K dimensional features (Deng et al., 2010), [2] with 128K dimensional features (Perronnin et al., 2012), [3] with 131K dimensional features (Sánchez and Perronnin, 2011), and [4] logistic regression on learned image features (Le et al., 2012). Note that the used training and test splits might be different between the different methods.*

To further demonstrate the generalization ability of the NCM classifier using learned metrics, we also compare it against the SVM baseline on the ImageNet-10K data set. We use projections learned and validated on the ILSVRC'10 data set, and only compute the means of the 10K classes. The results in Table 5.5 and Table 5.6 show that even in this extremely challenging setting the NCM classifier performs remarkably well compared to earlier mentioned SVM-based results of (Deng et al., 2010; Sánchez and Perronnin, 2011; Perronnin et al., 2012; Le et al., 2012) and our baseline, all of which require training 10K classifiers. We note that, to the best of our knowledge, our SVM baseline results exceed the previously known state-of-the-art on this data set. However, training our SVM baseline system took 9 and 280 CPU days respectively for the 4K and 64K features, while the computation of the means for the NCM classifier took approximately 3 and 48 CPU minutes respectively. This represents roughly a *8,500 fold speed-up* as compared to the baseline, without counting the time to learn the projection matrix.

Accuracy as a function of the number of training images of novel classes. In this set of experiments we consider the error as a function of the number of images that are used to compute the means of novel classes. Inspired by (Rohrbach et al., 2011), we also include results of a zero-shot learning experiment, where we use the ImageNet hierarchy to estimate the mean of novel classes from the means of related training classes, see Figure 5.5 for an illustration. We follow the idea of (Rohrbach et al., 2011) and estimate the

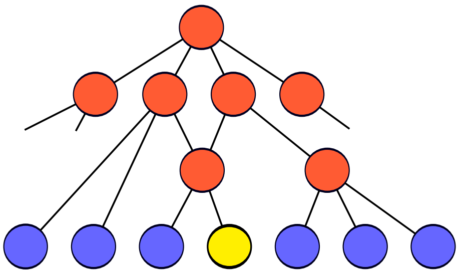


Figure 5.5 *Illustration of zero-shot learning using the ImageNet hierarchy. The class mean of the new class (yellow node) is estimated as the average of all its ancestor nodes. The mean for each of the internal nodes (red) is computed as the average of the means of all its descendant leaf nodes (blue). Only leaf nodes are used as classes in the ILSVRC'10.*

mean of a novel class $\boldsymbol{\mu}_z$ based on the means of all its ancestor nodes in the ILSVRC'10 class hierarchy:

$$\boldsymbol{\mu}_z = \frac{1}{|\mathcal{A}_z|} \sum_{a \in \mathcal{A}_z} \boldsymbol{\mu}_a, \tag{5.37}$$

where $\mathcal{A}_z$ denotes the set of ancestors of node z, and $\boldsymbol{\mu}_a$ is the mean of ancestor a. The mean of an internal node, $\boldsymbol{\mu}_a$, is computed as the average of the means of all its descendant training classes.

If we view the estimation of each class mean as the estimation of the mean of a Gaussian distribution, then the mean of a sample of images $\boldsymbol{\mu}_s$ corresponds to the Maximum Likelihood (ML) estimate, while the zero-shot estimate $\boldsymbol{\mu}_z$ can be thought of as a prior. We combine the prior with the ML estimate to obtain a maximum a-posteriori (MAP) estimate $\boldsymbol{\mu}_p$ of the class mean:

$$\boldsymbol{\mu}_p = \frac{n\boldsymbol{\mu}_s + m\boldsymbol{\mu}_z}{n+m}, \tag{5.38}$$

where $\boldsymbol{\mu}_s$ is weighed by the number of images n that were used to compute it, and $\boldsymbol{\mu}_z$ obtains a weight m determined on the validation set (Gauvain and Lee, 1994).

In Figure 5.6 we analyze the performance of the NCM classifier trained on the images of the same 800 classes used above, with a learned projection from 4K and 64K to 512 dimensions. The metric and the parameter m are validated using the images of the 200 held out classes of the validation set. We again report the error among test images of the held-out classes, both in a 1,000-way classification as above, and in a 200-way classification as in (Rohrbach et al., 2011). We repeat the experiment 10 times, and show error-bars at three times standard deviation.

For the error to stabilize we only need approximately 100 images to estimate the class means. The results also show that the prior leads to zero-shot performance of 66.5 (4K

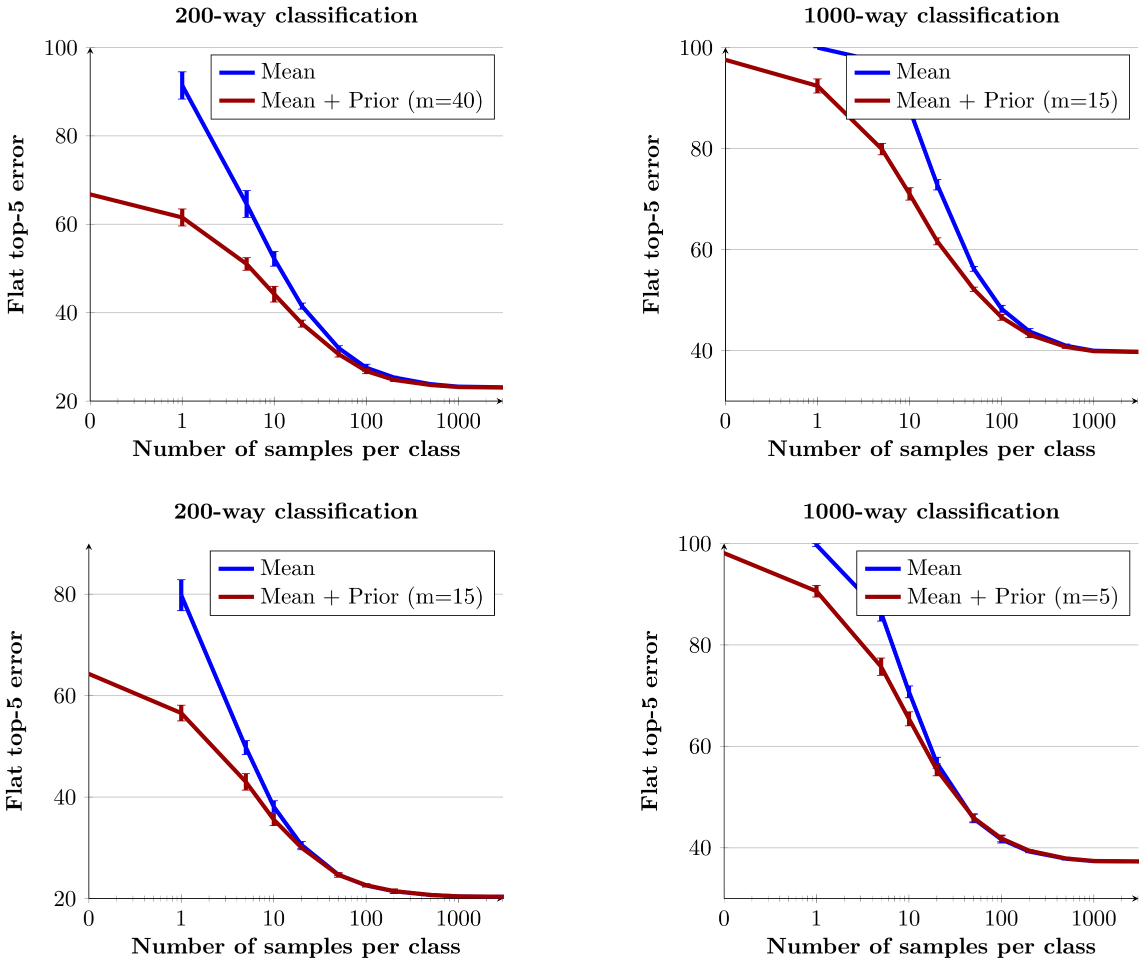


Figure 5.6 *Performance of NCM as a function of the number of images from the classes not used during training. Performance is shown for the 4K (*top*) and the 64K (*bottom*) features, and for a 200-way (*left*) and 1,000-way (*right*) classification setting. In each panel, the performance with (*red*) and without (*blue*) the zero-shot prior is shown.*

features) and 64.0 (64K features). These results are comparable to the result of 65.2 reported in (Rohrbach et al., 2011), even though a different set of 200 test-classes were used. Note that they also used different features, however their baseline performance of 37.6 top-5 error is comparable to our 4K features (38.2). More importantly, we show that the zero-shot prior can be effectively combined with the empirical mean to provide a smooth transition from the zero-shot setting to a setting with many training examples. Inclusion of the zero-shot prior leads to a significant error reduction in the regime where ten images or less are available.

Instance level image retrieval. Query-by-example image retrieval can be seen as an image classification problem where only a single positive sample (the query) is provided and negative examples are not explicitly provided. In this case the class mean simplifies to

INRIA Holidays data set				
without projection: 77.4%				
Dim.	PCA	JSCL	NCM	NCM-class
32	61.3	67.7	**69.3**	63.3
64	68.0	73.6	**75.4**	68.8
128	72.3	76.4	**79.6**	73.1
256	75.0	78.3	**80.2**	74.0
512	76.8	78.9	**80.6**	73.5

UKB data set			
without projection: 3.19			
Dim.	PCA	JSCL	NCM
32	2.82	3.04	**3.07**
64	3.01	**3.23**	**3.23**
128	3.08	3.31	**3.33**
256	3.15	**3.36**	3.32
512	3.18	**3.36**	3.31

Table 5.7 *Instance level image retrieval results,* (left)*, on the Holidays data set (performance in MAP), and* (right)*, on the UKB data set (performance is* $4\times$ *recall@4). The NCM methos is compared to the PCA baseline and the JSCL (Gordo et al., 2012).*

the query, and we use a metric learned for our NCM classifier on an auxiliary supervised data set to retrieve the most similar images for a given query.

Using classifiers to learn a metric for image retrieval was recently also considered in (Gordo et al., 2012). They found the Joint Subspace and Classifier Learning (JSCL) method to be the most effective. This basically amounts to jointly learning a set of classifiers and a projection matrix W using the WSABIE scoring function, Eq. (5.22), and minimizing the hinge-loss on class labels. After training, the classifiers are discarded and only the learned projection matrix W is used to compute distances between query and database images.

For this experiment we use the same public benchmarks as in (Gordo et al., 2012). First, the INRIA Holidays data set introduced by (Jégou et al., 2008) consists of 1,491 images of 500 scenes and objects. In the standard evaluation protocol, one image per scene / object is used as a query to search within the remaining images; the accuracy is measured as the mean average precision over the 500 queries (MAP). Second, the University of Kentucky Benchmark data set (UKB) introduced by (Nistér and Stewénius, 2006) contains 10,200 images of 2,550 objects. We follow the standard evaluation protocol, where each image is used as a query to search in the database. The performance is measured by $4\times$recall@4 averaged over all queries, hence the maximal score is 4.

For both data sets we extract the 4K image features also used in our earlier experiments, which are the same ones as those used in (Gordo et al., 2012). To compute the distance between two images, we use the cosine-distance, *i.e.* the dot-product on ℓ_2-normalized vectors. In analogy to (Gordo et al., 2012) we use the NCM objective to train a metric from the ILSVRC'10 data set, and do early stopping based on retrieval performance. To avoid tuning on the test data, the cross-validation is performed on the other data set, *i.e.* when testing on UKB we cross-validate the parameters on Holidays and vice-versa.

In Table 5.7 we compare the performance of the NCM based metrics with that of JSCL, and also include the performance of a baseline PCA method and the using the 4K descriptors without any projection. Finally, for the Holidays data set we included the NCM metric obtained for classification, *i.e.* using early-stopping based on classification performance of the ILSVRC'10 validation data set (NCM-class).

From the results we observe that the NCM metrics yield similar performance gains as the JSCL method on both data sets. In both cases a projection to only 128 dimensions yields equal or better retrieval performance as using the original 4K dimensional features. On the Holidays data set the NCM metric outperforms the JSCL metric, while on the UKB data set JSCL slightly outperforms NCM for projection dimensionalities $d \geq 256$. Both the NCM and JSCL methods are effective to learn a projection metric for instance level retrieval and outperform the unsupervised projection matrix obtained by PCA.

Note that it is curial to use early stopping based on the retrieval performance. From the results of NCM-class we observe that using a metric optimized for classification performance, has a retrieval performance which is worse than the using the original 4K descriptors. We thus conclude, that the classification objective determines a good "path" through the space of projection matrices, yet it is crucial to regularize for retrieval performance. We explain this discrepancy by the fact that instance level retrieval does not require the suppression of the within class variations which is needed for good classification. This observation also suggests that even better metrics might be learned by training NCM on a large set of queries with corresponding matches.

5.6 Conclusions

In this chapter we have evaluated techniques to learn low-rank, class-independent Mahalanobis distances to support k-NN and NCM classifiers for large scale image classification. We employ high-dimensional dense Fisher Vectors that lead to the current state-of-the-art results using a one-vs-rest SVM baseline approach. Surprisingly we found that the NCM classifier outperforms the more flexible k-NN approach. Moreover, using a learned metric, the performance of the NCM classifier is comparable to that of SVM baseline (even better with 4K dimensional features, but somewhat worse when using the 64K dimensional features), while projecting the data to as few as 256 dimensions. In practice, this means that the 1,2M training images from the ILSVRC'10 data set could be stored in about 1GB, instead of 20GB (4K) and 320GB (64K).

An advantage of both the k-NN and NCM classification methods is that they allow for extensions at (near) zero cost to new classes not used for training. A feature not shared by the SVM baseline, but which is essential for the use on real-life open-ended data sets where new images and classes are continuously added.

We have also introduced the non-linear NCMC classifier, which extent the NCM classifier by using multiple centroids to represent each class. Interestingly the used number of centroids offers a complexity trade off: from the linear NCM classifier to the non-linear and non-parametric k-NN classifier in the case when each image in the data set is used as a class centroid. Experimentally we have shown that the NCMC classifier, with 10 centroids per class, significantly improves over the NCM and the k-NN classifiers.

We have experimentally shown that our learned metrics generalize well to unseen classes. We have considered learning a metric on a subset of 800 classes and evaluated it on the 200 held out classes. Furthermore we have evaluated our metrics learned on the ILSVRC'10 data set, on the ImageNet-10K data set. For this experiment we have only computed the means of the 10,000 classes which has a negligible cost compared to the feature extraction process. We have shown to obtain competitive performance, and a 8,500 fold speed-up as compared to training 10,000 binary one-vs-rest classifiers. In addition, we have shown that our NCM classifiers can be used in a zero-shot setting where no training images are available for novel classes, and that the zero-shot model significantly boosts performance when combined with a class mean estimated from a limited amount of training images.

Finally, we have shown that NCM provides a unified way to treat classification and retrieval problems, since query-by-example image retrieval can be seen as a classification problem where only a single positive sample per class is provided. We have evaluated the NCM metric for image retrieval and found performance that is comparable to the current state-of-the-art on two public benchmarks.

6

Conclusion and Discussion

The important thing is not to stop questioning;
curiosity has its own reason for existing.
— Albert Einstein

Image classification and retrieval are essential image understanding tasks, required to grant computers the ability to see and interpret the visual world. Both tasks aim to relate low-level image features to a semantic concept of similarity. In this dissertation we have explored several statistical models for both tasks, that are learned in a supervised manner.

Next, we summarize our contributions and conclusions presented in the previous chapters. In Section 6.2 we outline prospective directions for further research.

6.1 Summary of conclusions

Below we review each of the goals stated in the introduction, and summarize our contributions and conclusions with respect to them.

Exploiting multi-modal data Motivated by the availability of large scale multi-modal image databases, in Chapter 3 we introduced parametrized versions of transmedia relevance feedback models. These models allow us, for example, to relate a textual query to purely visual content, by exploiting the intrinsic connection between the visual and textual modality given by the multi-modal documents in the data set. We have considered two different learning objectives to estimate the parameters for multi-modal retrieval functions from training data. Also, we have extended TagProp and applied these multi-modal distances for image annotation. Experimentally we have shown that using these parametrized relevance feedback models outperforms the current state-of-the-art on the data sets used for both tasks.

Modeling the structure of the output In Chapter 4, we introduced a series of graphical models, which explicitly encode dependencies among the labels. The key idea is that the prediction of a specific label will have positive correlations with some labels, while it will have negative correlations with other labels. For example, *bike* could have a positive correlation with *car*, but a negative correlation with *airplane*. In that case, predicting both *bike* and *airplane* to be relevant for the same image is implausible. In contrast to the dominant approach for image annotation, which uses independent label predictors, the proposed structured models can successfully encode these correlations.

For image annotation and attribute-based classification we have used generalized tree-structured models. Each node in the graphical model represents one or a few labels from the annotation vocabulary. While these models take into account a large number of label interactions, they still allow for tractable inference. We have shown that mixture-of-trees models, using trees with a different number of labels per node, and using unary functions based on pre-trained SVM scores, consistently outperform the state-of-the-art.

Furthermore, we have shown that learning score functions, with pairwise interactions optimized for ranking measures, like precision-at-k or mean average precision, is closely related to the NP-hard problem of quadratic assignment. This holds also for the tree-based models when optimized for ranking measures. However, we have identified c-star structured models as a class which is solvable in polynomial time, and still allows for modeling a substantial number of label interactions. When combined with scoring functions specially designed for the precision-at-k measure, these models are efficiently solvable and outperform an independent model. Nonetheless, we have observed that the mixture-of-trees model, which encodes more pairwise interactions but is optimized for classification accuracy, outperforms the c-star model.

Leveraging user interaction We have considered, in Chapter 4, an interactive labeling scenario, where a user is asked to set the value of a small number of labels at test time. We believe that such a scenario offers an interesting trade-off between prediction accuracy and labeling effort.

The tree-structured models we proposed for image annotation and attribute-based classification are able to transfer user input effectively to predictions of other image labels. After the user has set the value for a label, the inference in the tree changes significantly. Instead of propagating visual information, the state of the label becomes observed and the node shares information about its true state to the connected nodes. This new information translates to better predictions on the unobserved labels in the tree. The benefit from user input is even greater when the labels are selected following our entropy based criterion, which aims to reduce the remaining uncertainty of the other labels.

Classifying unseen classes Given that labels and concepts of interest in real-world data sets evolve over time, we have explored two strategies for classification of classes not seen during training.

In Chapter 4, we have followed the attribute-based classification paradigm, in which attribute predictors are combined with attribute-to-class specifications to classify test images from unseen classes. We have shown that our structured label models applied on the attribute level can improve over independent attribute prediction models. The improvement becomes especially significant in the interactive scenario, where a user provides a few attribute labels.

In Chapter 5, we have considered k-nearest neighbors and nearest class mean classification methods. These models, by design, allow for generalization to new classes at near-zero cost. However, the performance of these models depends critically on the distance metric used. For both methods we have explored and proposed efficient methods to learn distance metrics optimized for classification, and have shown the generalization performance of these methods to classes not seen during training. For the NCM method, we showed the generalization on a data set containing over ten thousand ImageNet classes, while using a metric learned on only thousand classes. Surprisingly, the NCM method performs competitively with classifiers specifically learned for these ten thousand classes, while it only computes the mean of each class.

Scaling to large data sets During the last five years the definition of large scale data sets has changed from data sets containing over ten thousand images to data sets containing over a million images. This trend reflects the need for efficient image understanding methods that can handle the large data sets encountered in today's practice.

For the distance-based classification methods discussed in Chapter 5, we have learned metrics on the ILSVRC'10 data set containing over one million images. Surprisingly, we have found that the NCM classifier with a learned metric performs similarly to SVMs, and outperforms the more flexible k-NN classifier. The NCM classifier allows for efficient classification, since it is a linear classifier. Also, the NCM classifier allows a reduction of the dimensionality of the image representation to as little as 256 dimensions, while still obtaining performance similar to SVM classifiers. As a result of its performance, combined with the possibility to classify classes not seen during training, we feel that NCM classifiers are an interesting alternative to SVMs.

6.2 Discussion of directions for further research

There are numerous ways to extend and build further on the methods described in this dissertation. For example aiming at improving the image representations, *e.g.* using the transmedia relevance models for combining multiple local features, or designing models for integrating the learning of image features in the classification objectives. Another example is to apply the discussed models in different settings or for different cost functions, *e.g.* the NCM classifier could be used for a multi-label task, or trained for a ranking loss.

In this section we detail three specific ideas for future research inspired directly by the work presented in this dissertation.

Active and interactive learning The current state-of-the-art image recognition systems are still unable to match humans in accuracy, robustness and the use of context between objects. As shown in Chapter 4, interactive image labeling obtains significantly higher performance compared to fully automatic label prediction, even when only a few labels are set by the user.

The interactive test scenario is very closely related to the active learning paradigm. Both aim to label images with a limited amount of user input. In practice both methods are also likely to be integrated: the labels set by a user, during an interactive session to obtain the best labeling for a specific image, could be used to update the classification models of the different labels, as is done in an active learning strategy.

We could design a classification system, with the goal to quickly obtain a precise labeling for an image, at the lowest possible cost. The vocabulary of this system would be determined by the interest of the user, *i.e.* the user decides which labels should be used in the annotation process. The system should be allowed to either predict a label given the currently learned models, or to ask the user or an external resource to set the label. The objective should be to balance the accuracy versus the manual annotation cost.

An interesting research direction would be to model the labels and classes, and to model the user of the system. First, we should be able to learn a structure over the labels, while the vocabulary is evolving, *e.g.* classes could be added by the user. Secondly, it would be interesting if we could detect when new classes are necessary, *e.g.* by using anomaly detection methods to decide that none of the current classes is sufficient to describe the image; or to detect when a class should be divided into sub-classes, *e.g.* to recognize a specific instance of the class instead of a general member of the class. Thirdly, since users might make mistakes, the system should determine a confidence score for each label set by the user to allow for erroneous labels. Finally, the system should also evaluate the users input to ensure that he/she produces high quality labels, especially when the systems pays an external user to label images. In this latter setting the external user might have an interest in earning money from the system and therefore to mislead the system to obtain more images to label. In that case the classification system becomes an adversarial machine learning problem.

Structured-prediction using local-learning approaches In Chapter 4 we introduced a series of parametric structured models and in Chapter 2, 3 and 5 we showed the success of local-learning approaches such as k-NN and TagProp. It would be interesting to develop a combined method, which allow for local-learning of structured prediction.

Let us assume that we are interested in multi-label image classification to jointly predict the value of a set of labels. We could apply local learning methods by assigning a weight to each neighbor, based on a visual distance, and propagate the annotation of the neighbors to a test image. As we have discussed in this dissertation, both the multi-class strategy as well as the independent labeling strategy have unsatisfying properties for joint estimation of a set of labels.

An interesting approach could be to use local learning method to propagate pairwise marginals from the neighbors to the test image. Thus instead of using a weighted count of the state of a single label, as is done in *e.g.* TagProp, we use a weighted count for each of the possible configurations for a pair of labels. We could model the labels in a tree-structure and obtain pairwise and single label marginals from the neighbors of the test image. While the label marginals using such a model will be the same as those of an independent model, the model can make different predictions for the maximum a posteriori state, where the best scoring binary label configuration is selected. Also, in this pairwise marginal model we can trivially incorporate observed variables, *e.g.* the labels set by a user, by updating the relevant probability tables. Therefore, this model can transfer the observed labels to the predictions of the other labels in an interactive scenario.

Directions of future research include how to define a tree-structure, how to add prior-knowledge to the model, and how to trade-off between locally and globally learned pairwise marginals. Besides, it would also be interesting to investigate how to adapt the tree-structure to the local neighborhood of the test image, and which other non tree-structured models can be used, *e.g*. by imposing constraints on the pairwise marginals.

Class representations In many classification systems it is assumed that a class can be represented by a single weight vector (*e.g*. for SVMs), or by a single class mean (*e.g*. for NCM). However, given the high intra-class variability, combined with other complexities of the visual world such as viewpoint variations and complex scene layouts, it is in fact rather surprising that single class representations perform so well. Below we discuss several ideas to use richer class representations for image classification.

In Chapter 5 we have explored the possibility of enriching class representations by using multiple centroids per class, and we have shown a class representation which contains multiple centroids outperforms the NCM classifier. However, we have used a fixed set of class means, obtained by employing the k-means algorithm on Euclidean distances. It could be advantageous to iterate the clustering and the metric learning to obtain more precise and possible a variable number of class representatives.

Another improvement could be made by obtaining better class representatives for the task at hand. For example, the k-means algorithm might be sub-optimal, since it can regard outliers as noise, while the set of class centroids should represent all images from the class. It would be interesting to investigate different clustering algorithms to obtain a set of class centroids. In our work, each class is clustered independently from the other classes, to obtain more discriminative class centroids we could however rely on a clustering algorithm which takes all classes into account.

Using multiple weight vectors to represent a class is uncommon in the SVM framework for image classification. However, it could be included, for example by using a latent SVM model for classification, where the latent variable is used to assign a sample to a "sub-class". Each sub-class is then parametrized by its own weight vector and models a subset of the images belonging to that class. While latent SVMs have shown good performance in object detection, to the best of our knowledge, they have not been applied for image classification.

In the context of multi-label classification problems, often binary SVM classification models are used. These models represent each label by a single weight vector, and dependencies among image labels are not explicitly encoded. In order to include more label dependencies, one could learn "joint-class" classifiers, similar to the idea of using "visual-phrases" which is used for object-detection. For example, a joint classifier for "car-bike" versus the rest could allow for greater benefit from the context wherein both objects occur together, compared to combining the individual car and bike classifier. Research questions include studying which joint-classes to learn, how to merge the classifier scores for label predictions, and how to exploit the structure between the labels.

A

Report de Thèse

A.1 Introduction

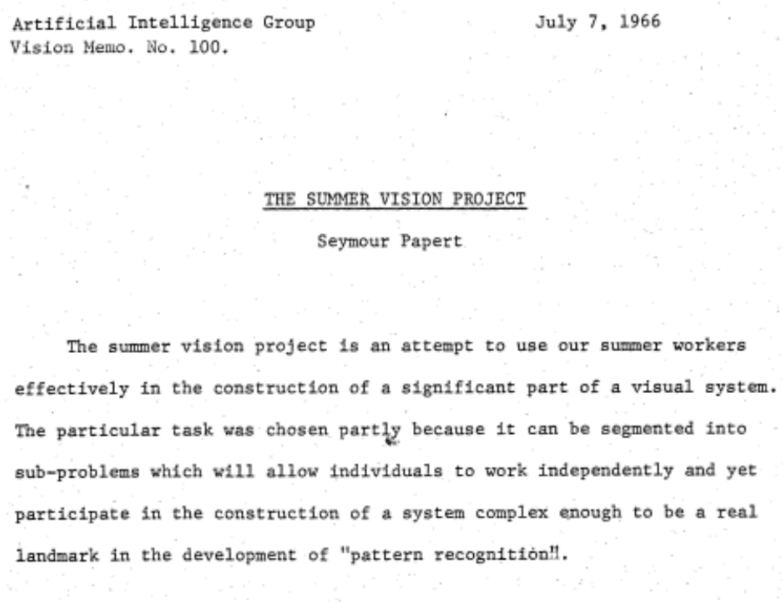
Photos et vidéos sont constamment prises, partagées, regardées, et recherchées. Ceci est la conséquence de la popularisation d'appareils de haute qualité permettant d'acquérir des images (appareils photos numériques et téléphones portables) combiné à la démocratisation des réseaux sociaux et des sites de partage de photos. L'omniprésence des photos et des vidéos sur internet devient évidente lorsqu'on considère les faits suivants :

- Le site de partage de photos Flickr héberge plus de 6 milliards de photos
- Les utilisateurs de Facebook y transfèrent 300 millions de nouvelles photos de façon quotidienne
- Le site de vidéo Youtube présente plus de 3 milliards d'heures de vidéo chaque mois

Ces données visuelles sont souvent accompagnées d'une description ou de méta-données. Ces descriptions se présentent sous différentes formes. Elles peuvent être générées automatiquement, comme les données EXIF, ou la localisation GPS. Elles peuvent être plus riches et haut-niveau, car fournies par les utilisateurs, comme par exemple des légendes, des étiquettes (ou tag). Ces données peuvent prendre la forme de l'insertion d'une image dans une hiérarchie de connaissance comme Wikipedia.

Les méthodes d'interprétation d'images peuvent ajouter de la valeur à ces données visuelles, en permettant notamment de trouver, chercher et utiliser ces images d'une façon sémantiquement pertinente, et centrée sur l'utilisateur, en utilisant le contenu visuel.

Pour illustrer ces possibilités, nous listons quelques applications qui utilisent l'interprétation d'images. Tout d'abord en robotique, la reconnaissance automatique d'objets peut être utilisée lorsqu'un robot doit effectuer la tâche d'attraper un objet spécifique sur une table, tout en se déplaçant autour des meubles dans une pièce. Deuxièmement, dans la



Artificial Intelligence Group July 7, 1966
Vision Memo. No. 100.

THE SUMMER VISION PROJECT

Seymour Papert

The summer vision project is an attempt to use our summer workers effectively in the construction of a significant part of a visual system. The particular task was chosen partly because it can be segmented into sub-problems which will allow individuals to work independently and yet participate in the construction of a system complex enough to be a real landmark in the development of "pattern recognition".

FIGURE A.1 *Description d'un projet d'été en vision, «**The summer vision project**» par Seymour Papert, au MIT en 1966. Le but de ce projet était de développer, pendant l'été, un système de reconnaissance visuelle.*

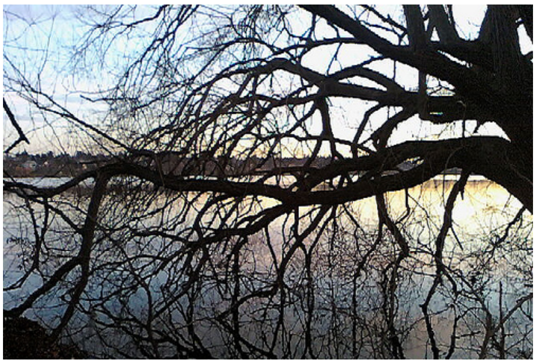
recherche d'image, la reconnaissance automatique d'actions permettrait de retrouver certaines scènes de film spécifiques, en se basant sur les actions effectuées par les acteurs. Troisièmement, lors de la création d'un livre photo, un utilisateur pourrait être assisté lors de la composition d'une livre à partir d'une collection de photos, en tenant compte de leur diversité, de leur mémorabilité, et de leur qualité esthétique.

Ces exemples d'applications, ainsi que l'explosion des données visuelles illustrent notre intérêt pour la compréhension d'images, soit dans le cadre de services web à grande échelle, soit à une plus petite échelle, pour aider les utilisateurs à organiser leur collections personnelles de photos.

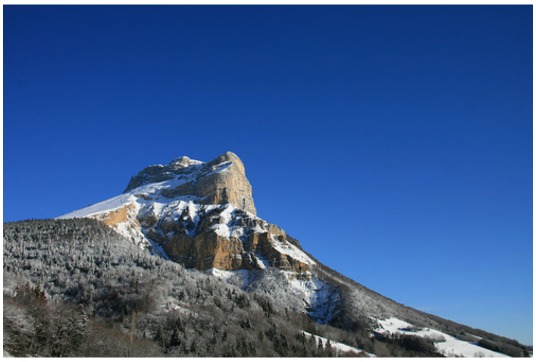
Dans cette thèse, nous nous focalisons sur l'interprétation des images, et plus précisément sur l'apprentissage supervisé de modèles statistiques pour la classification et la recherche d'images.

A.2 Contexte

À l'aube de l'intelligence artificielle, résoudre le problème de la vision par ordinateur semblait une tâche relativement simple. Ceci est illustré par le projet d'été «*The summer vision project*», proposé par Seymour Papert, au MIT en 1966, voir Figure A.1. Le but de ce projet était la création d'une bonne partie d'un système visuel, par un large groupe



Loved the colour of the sky and water against the dark tree reflections.



A long way to go to the top... Just above my house, close to Grenoble, in front of Belledonne mountains.

FIGURE A.2 *Exemples de légendes d'images produites par des humains. Les images sont extraites de la base de photos de "Stony Brooks University Captioned Photo Dataset" (Ordonez et al., 2011)*

d'étudiants. Ce système de vision artificielle aurait pu être utilisé comme entrée pour des tâches cognitives de haut niveau, comme le raisonnement et la planification. Il était donc un composant nécessaire pour reproduire l'intelligence humaine, ou pour créer un robot intelligent.

Quelques dizaines d'années plus tard, il s'avère qu'une compréhension complète du système visuel humain est loin d'être atteinte. La recherche en vision par ordinateur n'est toujours pas capable de reproduire le niveau de compréhension des images qu'a un enfant de 2 ans (Szeliski, 2011).

Cette difficulté est partiellement expliquée par le fait que l'interprétation de scènes et la reconnaissance d'objets sont des problèmes inverses, à travers lesquels nous cherchons à construire une compréhension du monde, étant donné une ou plusieurs images. Il est surprenant de voir que les humains réalisent cette tâche avec si peu d'effort, et avec une telle précision, malgré sa complexité. Le monde visuel dans toute sa complexité est difficile à modéliser, à cause de la quantité énorme de concepts différents, le nombre infini de projection possibles entre une scène en 3 dimensions et une image plan en 2 dimensions, et la complexité des scènes qui peuvent contenir plusieurs niveaux d'occultation et des conditions d'illumination différentes. De plus, il existe une variabilité intrinsèque très grande entre différents objets de la même classe.

A.3 Objectifs

Dans cette thèse, nous nous intéressons au problème de l'interprétation du contenu visuel d'une façon pertinente sémantiquement. Le but ultime serait de créer un système qui puisse générer une description de l'image de la qualité de celle d'un être humain, décrivant les scènes, les objets, et les relations entre ces objets, (voir les deux exemples proposés Figure A.2).

Un tel système devrait être capable de lier les représentations bas-niveau des images, aux représentations haut-niveau des images, par une description de son contenu. Le manque d'alignement entre les descriptions bas-niveau et l'interprétation haut-niveau des images est souvent connu comme le problème du faussé sémantique (Smeulders et al., 2000).

Dans cette thèse, nous nous intéressons plus particulièrement à l'apprentissage de modèles statistiques pour la classification et la recherche d'images. Ces deux tâches ont pour but de relier des descriptions bas-niveau d'images à une similarité sémantique avec une étiquette de classe, ou avec une autre image, tirant parti des bases d'images étiquetées.

Nous définissons brièvement ces deux tâches ci-dessous :

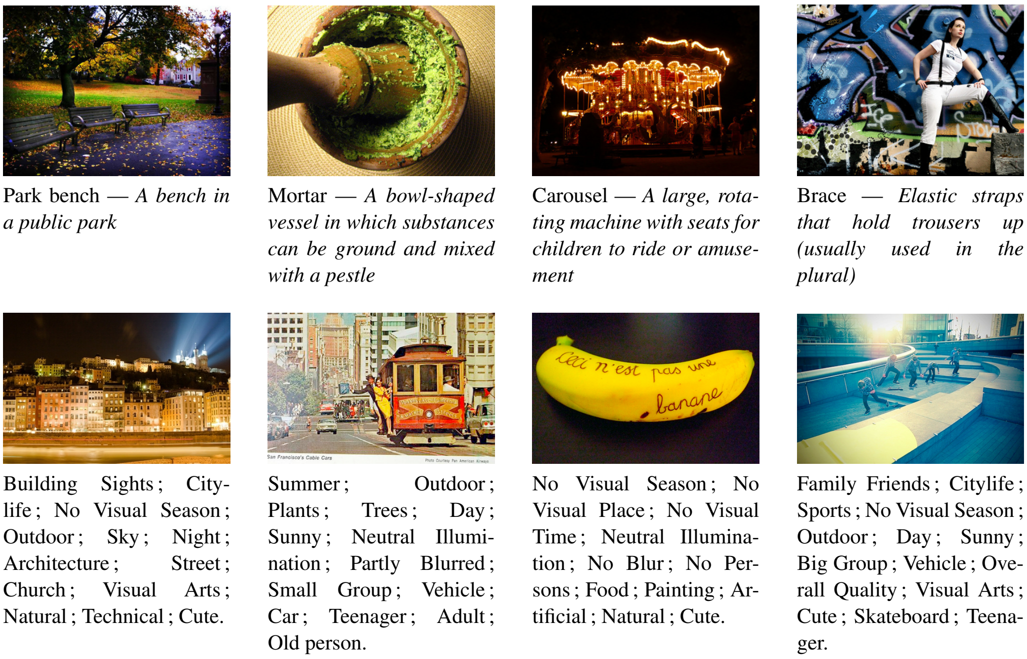
Classification d'images Le but de la classification d'images est de prédire la pertinence d'un ou plusieurs concepts sémantiques à partir d'un vocabulaire prédéfini. Nous distinguons la classification multi-classe, où les images sont associées à une seule étiquette sémantique, et la classification multi-étiquettes, où une image peut être associée à plus d'une étiquette. La diversité des concepts utilisés dans la classification d'images sont illustrés dans la Figure A.3.

Recherche d'images Le but de la recherche d'images est de retrouver, étant donnée une requête, les images les plus pertinentes d'une base. Il y a deux paradigmes classiques pour la recherche d'images : dans la *requête par texte*, la requête consiste en un ou plusieurs mots-clés, alors que dans la *requête par l'exemple*, la requête est une image, voir la Figure A.4 pour une illustration.

Ces deux tâches de compréhension d'images sont intiment liées. D'une part, les termes pertinents associés à une image par un système de classification d'image pourraient être utilisés dans un système de recherche basé sur une requête par le texte. D'autre part, la classification d'image peut être vue comme une tâche de recherche par l'exemple, pour laquelle l'image requête est annotée par les étiquettes de ses voisins visuels les plus proches.

Dans cette thèse, nous considérons cinq buts afin d'améliorer les techniques existantes pour ces tâches d'interprétation d'images.

1. **Utilisation de la multi-modalité des données** De grandes bases de données visuelles et multimodales sont actuellement disponibles, qui sont constituées, par exemple, d'images avec une description textuelle, ou de vidéos et leur sous-titre. L'exploitation de la relation entre ces différentes modalités présente un grand potentiel. Les différentes modalités d'un document peuvent être vues comme une forme de supervision mutuelle, bien que bruitée.



Park bench — *A bench in a public park*

Mortar — *A bowl-shaped vessel in which substances can be ground and mixed with a pestle*

Carousel — *A large, rotating machine with seats for children to ride or amusement*

Brace — *Elastic straps that hold trousers up (usually used in the plural)*

Building Sights ; Citylife ; No Visual Season ; Outdoor ; Sky ; Night ; Architecture ; Street ; Church ; Visual Arts ; Natural ; Technical ; Cute.

Summer ; Outdoor ; Plants ; Trees ; Day ; Sunny ; Neutral Illumination ; Partly Blurred ; Small Group ; Vehicle ; Car ; Teenager ; Adult ; Old person.

No Visual Season ; No Visual Place ; No Visual Time ; Neutral Illumination ; No Blur ; No Persons ; Food ; Painting ; Artificial ; Natural ; Cute.

Family Friends ; Citylife ; Sports ; No Visual Season ; Outdoor ; Day ; Sunny ; Big Group ; Vehicle ; Overall Quality ; Visual Arts ; Cute ; Skateboard ; Teenager.

FIGURE A.3 *Illustration de la diversité des concepts utilisés en classification d'images, à l'aide d'images extraites de :* (haut) *la base ImageNet ILSVRC'10 (Berg et al., 2010), présentée avec les noms de classes pertinentes, et leur description, parmi un ensemble de 1,000 classes différentes, et* (bas) *la base ImageClef VCDT'10 (Nowak and Huiskes, 2010) ainsi que certains des concepts pertinents, parmi un ensemble de 93 étiquettes.*

2. **Modélisation d'une sortie structurée** Un grand nombre de méthodes de classification d'images ne modélisent pas explicitement les dépendances entre les étiquettes de classes, alors que bien souvent, ces étiquettes sont implicitement enchâssées dans une structure. Par exemple, les noms d'animaux pourraient être représentés par la classification classique (ou classification linnéenne), ou la présence de certains objets pourrait promouvoir la présence d'une autre classe avec laquelle ils sont positivement corrélés.

3. **Utilisation d'interactions avec l'utilisateur** L'interaction avec un utilisateur au moment du test offre une opportunité intéressante d'améliorer la performance des algorithmes de classification. La quantité d'interaction manuelle permet un compromis intéressant entre la précision des prédictions, et la quantité de documents qui sont annotés pendant une période de temps donnée.

4. **Classification de classes jamais observées** Les bases de donnée réalistes sont souvent indéterminées, avec des images de classes jusqu'alors jamais vues ajoutées à la base. Le défi à relever est la création de systèmes pour la classification capables de classifier des images pour des classes jamais vues pendant l'apprentissage. Nous considérons deux stratégies populaires, qui permettent la classification de classes

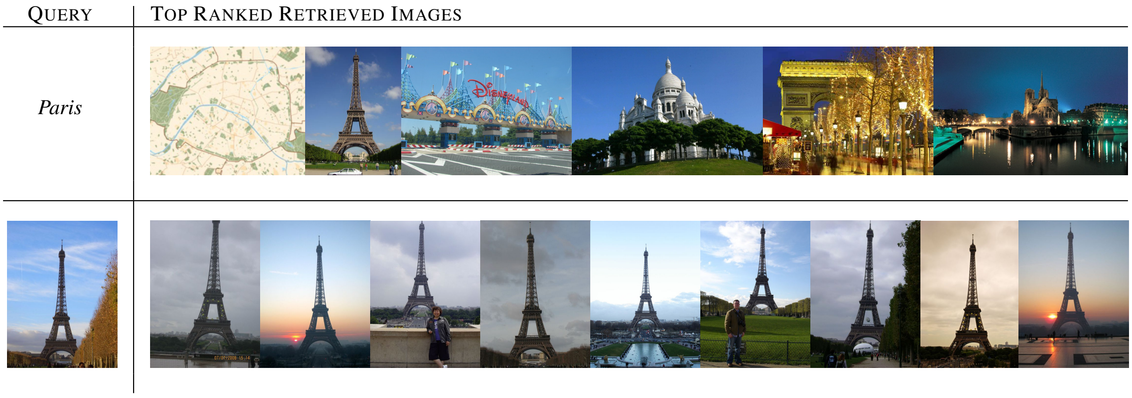


FIGURE A.4 *Exemple de systèmes de recherche, (*haut*), montrant les images les mieux classées par l'outil de recherche d'images de Googles pour la requête «*Paris*», et (*bas*) montrant les images les mieux classées pour un système similaire BigImbaz*[1]*, pour l'image requête montrées sur la gauche.*

jamais observées : la classification basée sur les attributs, et la classification basée sur la distance.

5. **Passage à l'échelle de grandes bases de données** Ces 5 dernières années, la définition d'une base à grande échelle a changé, passant de bases contenant un peu plus de dix mille images, à des bases contenant plus d'un million d'images. Cette tendance reflète un besoin pour des méthodes d'interprétation d'images efficaces, qui peuvent gérer les bases à grande échelle que l'on croise de nos jours. Idéalement, nous voudrions que nos algorithmes d'apprentissage soient efficaces selon plusieurs critères : le coût de calcul à l'apprentissage et au test, les besoins en stockage mémoire pour les représentation d'images, le nombre d'images d'apprentissage nécessaires à la généralisation, l'effort d'intervention manuelle nécessaire, par exemple pour l'annotation d'images d'apprentissage.

A.4 Plan de la thèse

Dans ce qui suit, nous décrivons le plan et les principaux contenus des chapitres présentés dans ce manuscrit.

Chapitre 2 Dans ce chapitre, nous décrivons de manière détaillée des représentations d'images utilisées pour la classification et la recherche d'images. Nous nous intéresserons plus particulièrement à la représentation par sac-de-mots (Csurka et al., 2004), et au vecteur de Fisher (Perronnin and Dance, 2007; Perronnin et al., 2010b).

[1] Demo disponible `http://bigimbaz.inrialpes.fr/`

Nous donnons aussi un aperçu des techniques de classification paramétriques, comme la régression logistique, les séparateurs à vaste marge (SVM) (Vapnik, 1995), mais aussi des techniques de classification non-paramètriques, comme les k plus proches voisins (k-PPV), ou le modèle TagProp (Guillaumin et al., 2009a). Bien que ce chapitre ne contienne aucune des contributions majeures de la thèse, il définit les concepts nécessaires à la compréhension des chapitres qui suivront.

Chapitre 3 Dans ce chapitre, nous considérons la recherche d'image, et la classification multi-étiquette d'image, utilisant des basées de données multimodales. Plus précisément dans ce cas, nous considérons des images accompagnées de mots-clés et/ou de légendes.

Dans les systèmes de recherche, une approche fructueuse pour l'exploitation de la mutlimodalité est l'utilisation de modèle de retour de pertinence entre média «*transmedia relevance feedback*» qui, pour une image requête donnée, utilisent les descriptions textuelles des plus proches voisins visuels de l'image requête, pour étendre la requête texte. Nous introduisons une paramétrisation pour apprendre les paramètres de ces modèles de retour de pertinence pour la recherche d'images.

Nous avons également étendu le modèle TagProp (Guillaumin et al., 2009a) afin d'exploiter ces distances de retour de pertinence pour la classification d'images multi-étiquettes. Nous montrons expérimentalement que l'utilisation de ces paramètres pour des modèles de retour de pertinence sont meilleurs que les modèles actuels de l'état de l'art sur les bases utilisées pour ces deux tâches. Le travail de ce chapitre a été publié en parti dans (Mensink et al., 2010b).

Chapitre 4 Dans ce chapitre, nous considérons les tâches d'annotation d'images, et de classification d'images basée sur les attributs. Pour ces tâches, nous introduisons une série de modèles structurés qui exploitent la corrélation entre les étiquettes de classe, et qui utilisent les interactions avec l'utilisateur. Dans ce scénario d'annotation interactive, un utilisateur doit donner la valeur d'un certain nombre d'étiquettes au moment du test. Nous pensons qu'un tel scénario propose un compromis intéressant entre la qualité de la prédiction, et les efforts d'annotation manuelle.

Nous avons utilisé des modèles graphiques à base de structure arborescente généralisée afin de modéliser les dépendances entre les étiquettes d'image et les attributs. Chaque nœud du modèle graphique représente une ou un petit nombre d'étiquettes, choisies parmi le vocabulaire d'annotation.

Alors que ces modèles prennent en compte un grand nombre d'interactions entre étiquettes, leur estimation est tout de même possible.

Nous avons appliqué ces modèles à base de structures arborescentes aux tâches d'annotation d'image, et à l'apprentissage sans exemples «*zero-shot learning*». Pour cette dernière, nous avons utilisé des paradigmes de classification basés sur les attributs. Dans ce cas, les prédictions d'attributs sont combinées avec des associations entre les attributs et les classes, et appliquées à la classification des images de test, pour des classes jamais vues. Des modèles structurés sont appliqués au niveau des attributes. Les modèles structurés que nous proposons dans ce manuscrit obtiennent de meilleurs résultats que les prédicateurs d'étiquettes ou d'attributs indépendants. Cependant, l'avantage majeur de ces modèles est leur capacité à bénéficier des interactions avec les utilisateurs, par rapport à des classifieurs indépendants.

Nous considérons aussi l'apprentissage de modèles spécifiques aux mesures de rangs utilisées pour l'évaluation. Malheureusement, l'optimisation de plusieurs mesures de performances basées sur le rang et intimement liée aux problèmes d'association quadratique qui sont NP-complets.

Ceci est valable aussi pour les modèles basés sur des arbres, lorsqu'ils sont optimisés pour des mesures de rang. Nous avons identifié une sous-classe de ces problèmes qui peut être résolue en complexité polynomiale, basée sur des modèles structurés c-étoile qui permettent la modélisation d'un grand nombre d'interactions binaires entre les étiquettes. Lorsqu'ils sont combinés avec des fonctions de score spécifiquement conçues pour la mesure précision-à-k, ces modèles peuvent être résolus efficacement, et produisent de meilleurs résultats que les modèles indépendants. Néanmoins, nous observons que les modèles de mélange d'arbres, qui encodent plus d'interactions binaires, mais sont optimisés pour la classification, sont meilleurs que le modèle c-étoile.

Ce travail a été publié dans (Mensink et al., 2011a, 2012b, 2011b).

Chapitre 5 Dans ce chapitre, nous considérons des classifiers par k plus proches voisins (k-PPV), et par plus proche moyenne (PPM), pour la classification d'images à grande échelle. Alors que ces méthodes permettent la généralisation à de nouvelles classes, elles ont tendance à obtenir des résultats peut satisfaisants lorsqu'une distance Euclidienne est utilisée. C'est pourquoi nous proposons d'explorer l'apprentissage de métriques partagées entre les classes.

Les méthodes proposées sont efficaces, et permettent un apprentissage sur la base «*ImageNet Large Scale Visual Recognition Challenge 2010*» qui contient plus de 1,2 millions d'images de 1 000 classes, tout en utilisant une représentation d'image de soixante-quatre mille dimensions. De façon surprenante, nous avons observé que le classifier PPM, avec une métrique apprise, obtient des performances similaires aux SVMs, et surpasse le classifier k-PPV, plus flexible. Puisque le classifier PPM est un classifier linéaire, il permet une classification efficace.

Pour la méthode PPM, nous montrons la généralisation à une base de données contenant plus de 10 millions de classes, tout en utilisant une métrique apprise sur la base ILS-VRC'10. De façon surprenante, la méthode PPM obtient des résultats compétitifs avec des classifieurs appris de façon spécifiques pour ces 10 mille classes, alors qu'il se contente de calculer la moyenne de ces classes. Ces bonnes performances, combinées avec la possibilité de classifier des classes qui n'ont pas été vues pendant l'apprentissage fait du classifier PPM une alternative intéressante au classifier SVM.

Ce travail a été publié dans (Mensink et al., 2012a, Submitted2012).

A.5 Perspectives

Ils existent de très nombreuses façon d'étendre les méthodes décrites dans cette thèse. Ici, nous détaillons trois idées spécifiques à poursuivre, directement inspirées des idées présentées dans ce manuscrit.

Apprentissages actif et interactif Les systèmes de reconnaissance d'images actuels sont toujours loin derrière les performances et la robustesse du système visuel humain, et n'ont pas la même capacité à utiliser le contexte entre les objets. Dans cette thèse, nous avons utilisé un système d'annotation interactif des images, qui a obtenu des performances significativement plus élevées que des systèmes de prédiction totalement automatiques, même si quelques étiquettes ont été fournies par un utilisateur. Ce scénario interactif est très proche du concept d'apprentissage actif. Ces deux scénarios visent à étiqueter des images avec une quantité limitée d'annotation manuelle.

Nous pourrions mettre un place un système de classification dont le but serait d'obtenir rapidement l'étiquetage précis d'une image, au coût le plus bas possible. Le vocabulaire de ce système serait déterminé par l'intérêt de l'utilisateur, en d'autres termes, l'utilisateur décide quelles étiquettes devraient être utilisées dans le processus d'annotation.

Nous proposons les questions de recherche suivantes :

- comment apprendre une structure entre les étiquettes, alors que le vocabulaire est en constante évolution ;
- comment détecter que de nouvelles classes sont nécessaires ;
- comment déterminer un score de confiance pour chaque étiquette fournie par l'utilisateur pour permettre de gérer des étiquettes erronées, et
- comment assurer que l'utilisateur produise des étiquettes de grande qualité.

Prédiction structurée, en utilisant l'apprentissage local Dans le chapitre 4, nous avons introduit une série de modèles structurés paramétriques, et dans les chapitres 2, 3 et 5, nous avons montré le succès des approches basées sur l'apprentissage local, comme k-PPV et TagProp. Il serait intéressant de développer une méthode combinée, qui permette l'apprentissage local pour la prédiction structurée. Supposons que nous sommes intéressés par la classification d'images multi-étiquette, c'est à dire prédire conjointement la valeur d'un ensemble d'étiquettes. Nous pourrions appliquer des méthodes d'apprentissage local, par l'affectation d'un point à chaque image de la base, basée sur des distances visuelles, et propager ces annotations aux images de test.

Pour permettre une prédiction structurée, nous pourrions propager les marginales binaires des plus proches voisins vers les images de test, c'est à dire utiliser un compte pondéré de chaque configuration possible pour les paires de labels. Nous pourrions modéliser les étiquettes dans une structure arborescente, et obtenir des marginales pour des paires d'étiquettes, et pour des étiquettes seules, à partir des voisins des images de test.

Parmi les directions de recherche possibles, citons :

- comment définir une structure arborescente et l'adapter au voisinage local d'une image de test
- comment ajouter de la connaissance a priori dans le modèle ; et
- comment modéliser un compromis entre des marginales binaires, apprises localement et globalement

Représentation des classes Dans beaucoup de systèmes de classification, on suppose qu'une classe peut être représentée par un simple vecteur de poids (par exemple pour les SVMs), ou par un simple vecteur moyen (par exemple pour NCM). Cependant, étant donnée la grande variabilité intra-classe, combinée avec d'autres difficultés inhérentes de la perception visuelle, comme les variations de points de vues, ou les scènes complexes, il est en fait surprenant qu'une représentation unique pour chaque classe fonctionne si bien.

Des représentations de classe plus riches pourraient être utilisées de différentes façons, par exemple

- Des moyennes par classe, utilisées pour le classifier NCMC (voir chapitre 5), auraient pu être obtenues en alternant un regroupement par les k moyennes, et un apprentissage de métrique, pour obtenir de meilleurs représentants par classe.
- Des représentants par classes plus discriminants pourraient être obtenus à l'aide d'une méthode de regroupement tenant compte de toutes les classes simultanément, et non indépendemment.
- Dans le cas de classifieurs SVM, nous pourrions utiliser des modèles de SVM à variables latentes pour la classification, ou les variables latentes seraient utilisées pour associer chaque image à une «sous-classe».
- Dans le contexte de la classification multi-étiquette, nous pourrions apprendre des classifieurs conjoints, de la même façon que les «phrases visuelles» sont utilisées pour la détection d'objet. Par exemple, un classifier conjoint pour «voiture-vélo» un-contre-tous serait d'un grand intérêt pour le cas où les deux objets apparaissent ensembles, par rapport à la combinaison de deux classifiers voiture et vélo indépendants.

Publications

Articles in peer-reviewed journals

- Large scale metric learning for distance-based image classification,
  T. Mensink, J. Verbeek, F. Perronnin, G. Csurka,
  Submitted to Transactions on Pattern Analysis and Machine Intelligence (PAMI).

- Tree-structured CRF models for interactive image labeling,
  T. Mensink, J. Verbeek, G. Csurka,
  IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 2012.

- Face recognition from caption-based supervision,
  M. Guillaumin, T. Mensink, J. Verbeek, C. Schmid,
  International Journal of Computer Vision (IJCV), 2012.

International peer-reviewed conferences

- Metric learning for large scale image classification:
  Generalizing to new classes at near-zero cost,
  T. Mensink, J. Verbeek, F. Perronnin, G. Csurka,
  European Conference on Computer Vision (ECCV), 2012. (Oral).

- Learning to rank and quadratic assignment,
  T. Mensink, J. Verbeek, T. Caetano,
  NIPS Workshop on Discrete Optimization in Machine Learning (DISCML), 2011.

- Learning structured prediction models for interactive image labeling,
  T. Mensink, J. Verbeek, G. Csurka,
  IEEE Conference on Computer Vision & Pattern Recognition (CVPR), 2011.

- Improving the Fisher kernel for large-scale image classification,
  F. Perronnin, J. Sánchez, T. Mensink,
  European Conference on Computer Vision (ECCV), 2010.

- Transmedia relevance feedback for image auto-annotation,
  T. Mensink, J. Verbeek, G. Csurka,
  British Machine Vision Conference (BMVC), 2010.

- EP for efficient stochastic control with obstacles,
  T. Mensink, J. Verbeek, Kappen B.,
  European Conference on Artificial Intelligence (ECAI), 2010.

- TagProp: Discriminative metric learning in nearest neighbor models
  for image auto-annotation,
  M. Guillaumin, T. Mensink, J. Verbeek, C. Schmid,
  IEEE International Conference on Computer Vision (ICCV), 2009. (Oral)

- Improving people search using query expansions:
  How friends help to find people,
  T. Mensink, J. Verbeek,
  European Conference on Computer Vision (ECCV), 2008. (Oral)

- Automatic face naming with caption-based supervision,
  M. Guillaumin, T. Mensink, J. Verbeek, C. Schmid,
  IEEE Conference on Computer Vision & Pattern Recognition (CPRV), 2008.

- Distributed EM learning for appearance based multi-camera tracking,
  T. Mensink, W. Zajdel, B. Kröse,
  IEEE/ACM International Conference on Distributed Smart Cameras (ICDSC), 2007.

National peer-reviewed conferences

- Apprentissage de distance pour l'annotation d'images par plus proches voisins,
  M. Guillaumin, J. Verbeek, C. Schmid, T. Mensink,
  Reconnaissance des Formes et Intelligence Artificielle (RFIA), 2010..

- Multi-observations newscast EM for distributed appearance based tracking,
  T. Mensink, W. Zajdel, B. Kröse,
  Belgian-Dutch Conference on Artificial Intelligence (BNAIC), 2007.

Other publications

- Weighted transmedia relevance feedback for image retrieval and auto-annotation,
  T. Mensink, J. Verbeek, G. Csurka,
  Technical Report, RT-0415, 2011.

- Face recognition from caption-based supervision,
  M. Guillaumin, T. Mensink, J. Verbeek, C. Schmid,
  Technical Report, RT-0392, 2010.

- LEAR and XRCE's participation to Visual Concept Detection Task - ImageCLEF 2010,
  T. Mensink, G. Csurka, F. Perronnin, J. Sánchez, J. Verbeek,
  Workshop Cross Language Image Retrieval (ImageCLEF), 2010.

- Image annotation with TagProp on the MIRFLICKR set,
  J. Verbeek, M. Guillaumin, T. Mensink, C. Schmid,
  ACM Conference on Multimedia Information Retrieval (MIR), invited paper, 2010.

- INRIA-LEARs participation to ImageCLEF 2009,
  M. Douze, M. Guillaumin, T. Mensink, C. Schmid, J. Verbeek,
  Working Notes for the CLEF 2009 Workshop, 2009.

- Multi-observations newscast EM for distributed multi-camera tracking,
  T. Mensink,
  Master Thesis, Universiteit van Amsterdam, 2007.

Bibliography

J. Ah-Pine, C. Cifarelli, S. Clinchant, G. Csurka, and J.-M. Renders. XRCE's participation to ImageCLEF 2008. In *Working Notes of the CLEF Workshop*. CLEF Campaign, 2008.

J. Ah-Pine, S. Clinchant, G. Csurka, F. Perronnin, and J.-M. Renders. Leveraging image, text and cross-media similarities for diversity-focused multimedia retrieval. In H. Müller, P. Clough, T. Deselaers, and B. Caputo, editors, *ImageCLEF - Experimental Evaluation in Visual Information Retrieval*. Springer, 2010.

N. Ailon. A simple linear ranking algorithm using query dependent intercept variables. In *European Conference on Information Retrieval*, 2009.

T. Arni, P. Clough, M. Sanderson, and M. Grubinger. Overview of the ImageCLEFphoto 2008 photographic retrieval task. In *Working Notes of the CLEF Workshop*. CLEF Campaign, 2008.

B. Bai, J. Weston, D. Grangier, R. Collobert, Y. Qi, K. Sadamasa, O. Chapelle, and K. Weinberger. Learning to rank with (a lot of) word features. *Information Retrieval – Special Issue on Learning to Rank*, 13:291–314, 2010.

K. Barnard, P. Duygulu, D. Forsyth, N. de Freitas, D. Blei, and M. Jordan. Matching words and pictures. *Journal of Machine Learning Research*, 3:1107–1135, 2003.

H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool. SURF: Speeded up robust features. *Computer Vision and Image Understanding*, 110:346–359, 2008.

S. Bengio, J. Weston, and D. Grangier. Label embedding trees for large multi-class tasks. In *Advances in Neural Information Processing Systems*, 2011.

A. Berg, J. Deng, and F.-F. Li. The ImageNet large scale visual recognition challenge 2010. http://www.image-net.org/challenges/LSVRC/2010, 2010.

T. Berg, A. Berg, J. Edwards, M. Maire, R. White, Y. Teh, E. Learned-Miller, and D. Forsyth. Names and faces in the news. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2004.

H. Bilen, V. P. Namboodiri, and V. P. Van Gool. Object and Action Classification with Latent Variables. In *Proceedings of the British Machine Vision Conference*, 2011.

C. Bishop. *Pattern recognition and machine learning*. Spinger-Verlag, 2006.

M. Blum, R. Floyd, V. Pratt, R. Rivest, and R. Tarjan. Time bounds for selection. *Journal of computer and system sciences*, 7(4), 1973.

O. Boiman, E. Shechtman, and M. Irani. In defense of nearest-neighbor based image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

L. Bottou. Large-scale machine learning with stochastic gradient descent. In *International Conference on Computational Statistics (COMPSTAT)*, 2010.

J. Bradley and C. Guestrin. Learning tree conditional random fields. In *Proceedings of the International Conference on Machine Learning*, 2010.

S. Branson, C. Wah, F. Schroff, B. Babenko, P. Welinder, P. Perona, and S. Belongie. Visual recognition with humans in the loop. In *Proceedings of the European Conference on Computer Vision*, 2010.

C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *Proceedings of the International Conference on Machine Learning*, 2005.

R. Burkard, M. Dell'Amico, and S. Martello. *Assignment Problems*. Society for Industrial and Applied Mathematics, 2009.

T. E. de Campos, G. Csurka, and F. Perronnin. Images as sets of locally weighted features. *Computer Vision and Image Understanding*, 2012.

J. Chai, H. Liua, B. Chenb, and Z. Baoa. Large margin nearest local mean classifier. *Signal Processing*, 90(1):236–248, 2010.

Y.-C. Chang and H.-H. Chen. Approaches of using a word-image ontology and an annotated image corpus as intermedia for cross-language image retrieval. In *Working Notes of the CLEF Workshop*, 2006.

K. Chatfield, V. Lempitsky, A. Vedaldi, and A. Zisserman. The devil is in the details: an evaluation of recent feature encoding methods. In *Proceedings of the British Machine Vision Conference*, 2011.

G. Checkik, V. Sharma, U. Shalit, and S. Bengio. Large scale online learning of image similarity through ranking. *Journal of Machine Learning Research*, 11:1109–1135, 2010.

M. Choi, J. Lim, A. Torralba, and A. Willsky. Exploiting hierarchical context on a large database of object categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467, 1968.

O. Chum, J. Philbin, J. Sivic, M. Isard, and A. Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *Proceedings of the International Conference on Computer Vision*, 2007.

R. Cinbis, J. Verbeek, and C. Schmid. Image categorization using fisher kernels of non-iid image models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

S. Clinchant, J.-M. Renders, and G. Csurka. XRCE's participation to ImageCLEFphoto 2007. In *Working Notes of the CLEF Workshop*, 2007.

T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21 –27, 1967.

K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2001.

G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *ECCV International Workshop on Statistical Learning in Computer Vision*, 2004.

C. Cusano, G. Ciocca, and R. Schettini. Image annotation using SVM. In *Proceedings of the SPIE Conference on Internet imaging*, volume 5304, 2004.

N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

R. Datta, D. Joshi, J. Li, and J. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys*, 40(2):5, 2008.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

J. Deng, A. Berg, K. Li, and L. Fei-Fei. What does classifying more than 10,000 image categories tell us? In *Proceedings of the European Conference on Computer Vision*, 2010.

A. Depeursinge and H. Müller. Fusion techniques for combining textual and visual information retrieval. In H. Müller, P. Clough, T. Deselaers, and B. Caputo, editors, *ImageCLEF - Experimental Evaluation in Visual Information Retrieval*. Springer, 2010.

C. Desai, D. Ramanan, and C. Fowlkes. Discriminative models for multi-class object layout. In *Proceedings of the International Conference on Computer Vision*, 2009.

I. Dimitrovski, D. Kocev, S. Loskovska, and S. Džeroski. Detection of Visual Concepts and Annotation of Images Using Predictive Clustering Trees. In *Workshop ImageCLEF*, 2010.

P. Duygulu, K. Barnard, N. de Freitas, and D. Forsyth. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *Proceedings of the European Conference on Computer Vision*, 2002.

N. M. Elfiky, F. S. Khan, J. van de Weijer, and J. Gonzalez. Discriminative compact pyramids for object and scene recognition. *Pattern Recognition*, 45:1627–1636, 2012.

G. Erdogan. *Quadratic assignment problem: linearizations and polynomial time solvable cases*. PhD thesis, Bilkent University, 2006.

M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop, 2007.

M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2008 Results. http://www.pascal-network.org/challenges/VOC/voc2008/workshop, 2008.

M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2009 Results. http://www.pascal-network.org/challenges/VOC/voc2009/workshop, 2009.

M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2010 Results. http://www.pascal-network.org/challenges/VOC/voc2010/workshop, 2010.

M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2011 Results. http://www.pascal-network.org/challenges/VOC/voc2011/workshop, 2011.

J. Farquhar, S. Szedmak, H. Meng, and J. Shawe-Taylor. Improving "bag-of-keypoints" image categorisation: Generative models and pdf-kernels. Technical report, University of Southamption, 2005.

L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):594–611, 2006.

A. Frome, J. Malik, and Y. Singer. Image retrieval and classification using local distance functions. In *Advances in Neural Information Processing Systems*, volume 19, pages 417–424, 2007.

T. Gao and D. Koller. Discriminative learning of relaxed hierarchy for large-scale visual recognition. In *Proceedings of the International Conference on Computer Vision*, 2011.

J.-L. Gauvain and C.-H. Lee. Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains. *IEEE Transactions on Speech and Audio Processing*, 2:291–298, 1994.

J. van Gemert, J.-M. Geusebroek, C. J. Veenman, and A. W. M. Smeulders. Kernel codebooks for scene categorization. In *Proceedings of the European Conference on Computer Vision*, 2008.

J. van Gemert, C. Snoek, C. Veenman, S. Smeulders, and J.-M. Geusebroek. Comparing compact codebooks for visual categorization. *Computer Vision and Image Understanding*, 114(4):450–462, 2010.

A. Gordo, J. Rodríguez, F. Perronnin, and E. Valveny. Leveraging category-level labels for instance-level image retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

D. Grangier and S. Bengio. A discriminative kernel-based model to rank images from text queries. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(8): 1371–1384, 2008.

R. Gray and D. Neuhoff. Quantization. *IEEE Transactions on Information Theory*, 44(6): 2325–2383, 1998.

G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical report, California Institute of Technology, 2007.

M. Grubinger, P. Clough, H. Müller, and T. Deselaers. The IAPR benchmark: A new evaluation resource for visual information systems. In *International Conference on Language Resources and Evaluation*, 2006.

M. Guillaumin, T. Mensink, J. Verbeek, and C. Schmid. TagProp: Discriminative metric learning in nearest neighbor models for image auto-annotation. In *Proceedings of the International Conference on Computer Vision*, 2009a.

M. Guillaumin, J. Verbeek, and C. Schmid. Is that you? Metric learning approaches for face identification. In *Proceedings of the International Conference on Computer Vision*, 2009b.

M. Guillaumin, T. Mensink, J. Verbeek, and C. Schmid. Face recognition from caption-based supervision. *International Journal of Computer Vision*, 96(1):64–82, 2012.

M. Guillaumin. *Exploiting Multimodal Data for Image Understanding*. PhD thesis, Institut National Polytechnique de Grenoble - INPG, 2010.

C. Harris and M. Stephens. A combined corner and edge detector. In *Alvey Vision Conference*, 1988.

T. Hertz, A. Bar-Hillel, and D. Weinshall. Learning distance functions for image retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2004.

M. Huiskes and M. Lew. The MIR Flickr retrieval evaluation. In *ACM International Conference on Multimedia Information Retrieval*, 2008.

T. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In *Advances in Neural Information Processing Systems*, 1999.

H. Jégou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *Proceedings of the European Conference on Computer Vision*, 2008.

H. Jégou, M. Douze, and C. Schmid. On the burstiness of visual elements. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

H. Jégou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128, 2011.

H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid. Aggregating local image descriptors into compact codes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012.

H. Jégou, M. Douze, and C. Schmid. Improving bag-of-features for large scale image search. *International Journal of Computer Vision*, 87(3):316–336, 2010.

H. Jégou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3304–3311, 2010.

T. Joachims. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In *Proceedings of the European Conference on Machine Learning*, 1998.

T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining*, 2002.

T. Joachims. A support vector method for multivariate performance measures. In *Proceedings of the International Conference on Machine Learning*, pages 377–384, 2005.

F. Jurie and B. Triggs. Creating efficient codebooks for visual recognition. In *Proceedings of the International Conference on Computer Vision*, 2005.

F. S. Khan, J. van de Weijer, and M. Vanrell. Top-down color attention for object recognition. In *Proceedings of the International Conference on Computer Vision*, 2009.

J. Kludas, S. Marchand-Maillet, and E. Bruno. Information fusion in multimedia information retrieval. In *Workshop on Adaptive Multimedia Retrieval*, 2007.

T. Koopmans and M. Beckmann. Assignment problems and the location of economic activities. *Econometrica*, 1957.

J. Krapac, M. Allan, J. Verbeek, and F. Jurie. Improving web-image search results using query-relative classifiers. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

J. Krapac, J. Verbeek, and F. Jurie. Learning tree-structured descriptor quantizers for image categorization. In *Proceedings of the British Machine Vision Conference*, 2011a.

J. Krapac, J. Verbeek, and F. Jurie. Modeling spatial layout with fisher vectors for image categorization. In *Proceedings of the International Conference on Computer Vision*, 2011b.

C. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

I. Laptev, M. Marszałek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

H. Larochelle, D. Erhan, and Y. Bengio. Zero-data learning of new tasks. In *AAAI Conference on Artificial Intelligence*, 2008.

S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

Q. Le, A. Smola, O. Chapelle, and C. H. Teo. Optimization of ranking measures. *Journal of Machine Learning Research*, 1:1–48, 2010.

Q. Le, M. Ranzato, R. Monga, M. Devin, K. Chen, G. S. Corrado, J. Dean, and A. Y. Ng. Building high-level features using large scale unsupervised learning. In *Proceedings of the International Conference on Machine Learning*, 2012.

D. Lewis. Applying support vector machines to the TREC-2001 batch filtering and routing tasks. In *Proceedings of Text REtrieval Conference (TREC)*, 2001.

D. Lewis. Naive (Bayes) at Forty: The Independence Assumption in Information Retrieval. In *Proceedings of the European Conference on Machine Learning*, 1998.

H. Li. *Learning to Rank for Information Retrieval and Natural Language Processing*. Morgan & Claypool Publishers, 2011.

X.-C. Lian, Z. Li, B.-L. Lu, and L. Zhang. Max-margin dictionary learning for multiclass image categorization. In *Proceedings of the European Conference on Computer Vision*, 2010.

Y. Lin, F. Lv, S. Zhu, M. Yang, T. Cour, K. Yu, L. Cao, and T. Huang. Large-scale image classification: Fast feature extraction and SVM training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

T. Lindeberg. Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30(2):79–116, November 1998.

J. Liu, M. Li, Q. Liu, H. Lu, and S. Ma. Image annotation via graph learning. *Pattern Recognition*, 42(2):218–228, 2009.

D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

N. Maillot, J.-P. Chevallet, V. Valea, and J. Lim. IPAL inter-media pseudo-relevance feedback approach to ImageCLEF 2006. In *Working Notes of the CLEF Workshop*, 2006.

J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Supervised dictionary learning. In *Advances in Neural Information Processing Systems*, 2008a.

J. Mairal, M. Leordeanu, F. Bach, M. Hebert, and J. Ponce. Discriminative sparse image models for class-specific edge detection and image interpretation. In *Proceedings of the European Conference on Computer Vision*, 2008b.

S. Maji and A. Berg. Max-margin additive models for detection. In *Proceedings of the International Conference on Computer Vision*, 2009.

S. Maji, A. C. Berg, and J. Malik. Classification using intersection kernel support vector machines is efficient. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

A. Makadia, V. Pavlovic, and S. Kumar. A new baseline for image annotation. In *Proceedings of the European Conference on Computer Vision*, 2008.

C. Manning, P. Raghavan, and H. Schütze. *Introduction to information retrieval*. Cambridge University Press Cambridge, 2008.

E. Mbanya, C. Hentschel, S. Gerke, M. Liu, A. Nürnberger, and P. Ndjiki-Nya. Augmenting Bag-of-Words - Category Specific Features and Concept Reasoning. In *Workshop ImageCLEF*, 2010.

T. Mensink, G. Csurka, F. Perronnin, J. Sánchez, and J. Verbeek. LEAR and XRCE's participation to visual concept detection task - ImageCLEF 2010. In *Working Notes of the CLEF Workshop*, 2010a.

T. Mensink, J. Verbeek, and G. Csurka. Trans media relevance feedback for image autoannotation. In *Proceedings of the British Machine Vision Conference*, 2010b.

T. Mensink, J. Verbeek, and G. Csurka. Learning structured prediction models for interactive image labeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011a.

T. Mensink, J. Verbeek, F. Perronnin, and G. Csurka. Metric learning for large scale image classification: Generalizing to new classes at near-zero cost. In *Proceedings of the European Conference on Computer Vision*, 2012a.

T. Mensink, J. Verbeek, F. Perronnin, and G. Csurka. Large scale metric learning for distance-based image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Submitted2012.

T. Mensink, J. Verbeek, and T. Caetano. Learning to rank and quadratic assignment. In *NIPS Workshop on Discrete Optimization in Machine Learning*, Granada, Spain, December 2011b.

T. Mensink, J. Verbeek, and G. Csurka. Tree-structured crf models for interactive image labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012b.

K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004.

K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, 2005.

N. Motohashi, R. Izawa, and T. Takagi. Meiji University at ImageCLEF2010 Visual Concept Detection and Annotation Task. In *Workshop ImageCLEF*, 2010.

H. Müller, P. Clough, T. Deselaers, and B. Caputo. *ImageCLEF - Experimental Evaluation in Visual Information Retrieval*. Springer, 2010.

S. Navarro, M. García, F. Llopis, M. Díaz, R. Muñoz, M. Martín, L. Ureña, and A. Montejo. Text-mess in the ImageCLEFphoto08 task. In *Working Notes of the CLEF Workshop*, 2008.

D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

E. Nowak and F. Jurie. Learning visual similarity measures for comparing never seen objects. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

E. Nowak, F. Jurie, and B. Triggs. Sampling strategies for bag-of-features image classification. In *Proceedings of the European Conference on Computer Vision*, 2006.

S. Nowak and M. Huiskes. New strategies for image annotation: Overview of the photo annotation task at ImageCLEF 2010. In *Working Notes of CLEF*, 2010.

S. Nowozin and C. Lampert. Structured learning and prediction in computer vision. *Foundations and Trends in Computer Graphics and Vision*, 6:185–365, 2011.

S. Nowozin, P. Gehler, and C. Lampert. On Parameter Learning in CRF-based Approaches to Object Class Image Segmentation. In *Proceedings of the European Conference on Computer Vision*, 2010.

N. O'Hare, P. Wilkins, C. Gurrin, E. Newman, G. Jones, and A. Smeaton. Dcu at imageclefphoto 2008. In *Working Notes of the CLEF Workshop*, 2008.

A. Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, 2001.

V. Ordonez, G. Kulkarni, and T. L. Berg. Im2text: Describing images using 1 million captioned photographs. In *Advances in Neural Information Processing Systems*, 2011.

J. Pan, H. Yang, C. Faloutsos, and P. Duygulu. Automatic multimedia cross-modal correlation discovery. In *Proceedings of the ACM international conference on Knowledge discovery and data mining (SIGKDD)*, 2004.

F. Perronnin and C. Dance. Fisher kernels on visual vocabularies for image categorization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

F. Perronnin, C. Dance, G. Csurka, and M. Bressan. Adapted vocabularies for generic visual categorization. In *Proceedings of the European Conference on Computer Vision*, 2006.

F. Perronnin, Yan Liu, and J.-M. Renders. A family of contextual measures of similarity between distributions with application to image retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

F. Perronnin, Y. Liu, J. Sánchez, and H. Poirier. Large scale image retrieval with compressed fisher vectors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010a.

F. Perronnin, J. Sánchez, and T. Mensink. Improving the Fisher kernel for large-scale image classification. In *Proceedings of the European Conference on Computer Vision*, 2010b.

F. Perronnin, Z. Akata, Z. Harchaoui, and C. Schmid. Towards good practice in large-scale learning for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

J. Petterson and T. Caetano. Submodular multi-label learning. In *Advances in Neural Information Processing Systems*, 2011.

J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

J. Platt. Probabilities for SV machines. In *Advances in Large Margin Classifiers*, 2000.

J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in kernel methods*, pages 185–208. MIT Press, 1999.

P. Pletscher, C. Ong, and J. Buhmann. Spanning tree approximations for conditional random fields. In *Proceedings of the International Workshop on Artificial Intelligenceand Statistics*, 2009.

J. Ponte and W. Croft. A language modelling approach to information retrieval. In *Proceedings of the International ACM Conference on Research and Development in Information Retrieval (SIGIR)*, 1998.

T. Qin, T.-Y. Liu, J. Xu, and H. Li. Letor: A benchmark collection for research on learning to rank for information retrieval. *Information Retrieval*, 13:346–374, 2010.

A. Rabinovich, A. Vedaldi, C. Galleguillos, and E. W. S. Belongie. Objects in context. In *Proceedings of the International Conference on Computer Vision*, 2007.

M. Rohrbach, M. Stark, and B. Schiele. Evaluating knowledge transfer and zero-shot learning in a large-scale setting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18:613–620, 1975.

G. Salton. *Automatic text processing: the transformation, analysis, and retrieval of information by computer*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

G. Salton and C. Buckley. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 1990.

J. Sánchez and F. Perronnin. High-dimensional signature compression for large-scale image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

K. van de Sande, T. Gevers, and C. G. M. Snoek. Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1582–1596, 2010.

B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, 2002.

F. Schroff, A. Criminisi, and A. Zisserman. Harvesting image databases from the web. In *Proceedings of the International Conference on Computer Vision*, 2007.

B. Settles. Active learning literature survey. Technical Report 1648, University of Wisconsin-Madison, 2009.

G. Sharma and F. Jurie. Learning discriminative spatial representation for image classification. In *Proceedings of the British Machine Vision Conference*, 2011.

E. Shechtman and M. Irani. Matching local self-similarities across images and videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2007.

J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Proceedings of the International Conference on Computer Vision*, 2003.

A. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content-based image retrieval at the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12):1349–1380, 2000.

C. Snoek, M. Worring, and A. Smeulders. Early versus late fusion in semantic video analysis. *ACM Multimedia*, 2005.

S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf. Large scale multiple kernel learning. *Journal of Machine Learning Research*, 7:1531–1565, 2006.

R. Szeliski. *Computer Vision: Algorithms and Applications*. Spinger-Verlag, 2011.

B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. In *Advances in Neural Information Processing Systems*, 2003.

S. Tollari, M. Detyniecki, M. Ferecatu, H. Glotin, P. Mulhem, M. Amini, A. Fakeri-Tabrizi, P. Gallinari, H. Sahbi, and Z. Zhao. Consortium AVEIR at imageCLEF photo 2008: on the fusion of runs. In *Working Notes of the CLEF Workshop*, 2008.

T. Tommasi and B. Caputo. The more you know, the less you learn: from knowledge transfer to one-shot learning of object categories. In *Proceedings of the British Machine Vision Conference*, 2009.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

K. van de Sande and T. Gevers. The University of Amsterdam's Concept Detection System at ImageCLEF 2010. In *Workshop ImageCLEF*, 2010.

V. Vapnik. *The Nature of Statistical Learning Theory*. Springer Verlag, 1995.

A. Vedaldi and A. Zisserman. Efficient additive kernels via explicit feature maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(3), 2011.

C. Veenman and D. Tax. LESS: a model-based classifier for sparse subspaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(9):1496–1500, 2005.

J. Verbeek, M. Guillaumin, T. Mensink, and C. Schmid. Image annotation with TagProp on the MIRFLICKR set. In *ACM International Conference on Multimedia Information Retrieval*, 2010.

S. Vijayanarasimhan and K. Grauman. Multi-level active prediction of useful image annotations for recognition. In *Advances in Neural Information Processing Systems*, 2009.

J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

A. Webb. *Statistical pattern recognition*. Wiley, New-York, NY, USA, 2002.

J. van de Weijer and C. Schmid. Coloring local feature extraction. In *Proceedings of the European Conference on Computer Vision*, 2006.

K. Weinberger and L. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10:207–244, 2009.

K. Weinberger, J. Blitzer, and L. Saul. Distance metric learning for large margin nearest neighbor classification. In *Advances in Neural Information Processing Systems*, 2006.

J. Weston and C. Watkins. Support vector machines for multi-class pattern recognition. In *Proceedings of the European Symposium on Artificial Neural Networks*, 1999.

J. Weston, S. Bengio, and N. Usunier. Large scale image annotation: Learning to rank with joint word-image embeddings. In *Proceedings of the European Conference on Machine Learning*, 2010.

J. Weston, S. Bengio, and N. Usunier. WSABIE: Scaling up to large vocabulary image annotation. In *International Joint Conference on Artificial Intelligence*, 2011.

J. Winn, A. Criminisi, and T. Minka. Object categorization by learned universal visual dictionary. In *Proceedings of the International Conference on Computer Vision*, pages 1800–1807, 2005.

J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

Y. Yue, T. Finley, F. Radlinski, and T. Joachims. A support vector method for optimizing average precision. In *Proceedings of the International ACM Conference on Research and Development in Information Retrieval (SIGIR)*, 2007.

H. Zhang, A. Berg, M. Maire, and J. Malik. SVM-KNN: Discriminative nearest neighbor classification for visual category recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2126–2136, 2006.

J. Zhang, M. Marszałek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: a comprehensive study. *International Journal of Computer Vision*, 73(2):213–238, 2007.

X. Zhang, Z. Li, L. Zhang, W. Ma, and H.-Y. Shum. Efficient indexing for large scale visual search. In *Proceedings of the International Conference on Computer Vision*, 2009.

Z. Zheng, H. Zha, T. Zhang, O. Chapelle, K. Chen, and G. Sun. A general boosting method and its application to learning ranking functions for web search. In *Advances in Neural Information Processing Systems*, 2008.

X. Zhou, X. Zhang, Z. Yan, S.-F. Chang, M. Hasegawa-Johnson, and T. Huang. Sift-bag kernel for video event analysis. In *ACM Multimedia*, 2008.

X. Zhou, K. Yu, T. Zhang, and T. Huang. Image classification using super-vector coding of local image descriptors. In *Proceedings of the European Conference on Computer Vision*, 2010.