HAL open science

Contributions to large-scale learning for image classification

Zeynep Akata

To cite this version:

Zeynep Akata. Contributions to large-scale learning for image classification. Signal and Image Processing. Université de Grenoble, 2014. English. NNT : 2014GRENM003 . tel-00873807v2

HAL Id: tel-00873807

https://theses.hal.science/tel-00873807v2

Submitted on 22 May 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

CONTRIBUTIONS TO LARGE-SCALE LEARNING FOR IMAGE CLASSIFICATION

Zeynep Akata

PhD Thesis

l'École Doctorale
Mathématiques, Sciences et Technologies de
l'Information, Informatique
de Grenoble

UNIVERSITÉ DE GRENOBLE

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE GRENOBLE

Spécialité : **Mathématiques et Informatique**

Arrêté ministériel : 7 août 2006

Présentée par

Zeynep Akata

Thèse dirigée par **Cordelia Schmid**
et codirigée par **Florent Perronnin**

préparée au sein **Laboratoire Jean Kuntzmann**
dans **l'École Doctorale Mathématiques, Sciences et Technologies de l'Information, Informatique de Grenoble**

Contributions à l'apprentissage grande échelle pour la classification d'images

Contributions to large-scale learning for image classification

Thèse soutenue publiquement le **6 janvier 2014**,
devant le jury composé de :

Dr. Georges Quenot
CNRS Grenoble, France, Président
Prof. Christoph Lampert
IST Vienna, Austria, Rapporteur
Prof. Matthieu Cord
UPMC-Sorbonne Paris, France, Rapporteur
Prof. Vittorio Ferrari
University of Edinburgh, United Kingdom, Examinateur
Dr. Cordelia Schmid
INRIA Grenoble, France, Directeur de thèse
Dr. Florent Perronnin
XRCE Grenoble, France, Co-Directeur de thèse

The research described in this thesis was carried out at the LEAR team of INRIA-Grenoble and the CV 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

Building algorithms that classify images on a large scale is an essential task due to the difficulty in searching massive amount of unlabeled visual data available on the Internet. We aim at classifying images based on their content to simplify the manageability of such large-scale collections. Large-scale image classification is a difficult problem as datasets are large with respect to both the number of images and the number of classes. Some of these classes are fine grained and they may not contain any labeled representatives. In this thesis, we use state-of-the-art image representations and focus on efficient learning methods. Our contributions are (1) a benchmark of learning algorithms for large scale image classification, and (2) a novel learning algorithm based on label embedding for learning with scarce training data.

Firstly, we propose a benchmark of learning algorithms for large scale image classification in the fully supervised setting. It compares several objective functions for learning linear classifiers such as one-vs-rest, multiclass, ranking and weighted average ranking using the stochastic gradient descent optimization. The output of this benchmark is a set of recommendations for large-scale learning. We experimentally show that, online learning is well suited for large-scale image classification. With simple data rebalancing, One-vs-Rest performs better than all other methods. Moreover, in online learning, using a small enough step size with respect to the learning rate is sufficient for state-of-the-art performance. Finally, regularization through early stopping results in fast training and a good generalization performance.

Secondly, when dealing with thousands of classes, it is difficult to collect sufficient labeled training data for each class. For some classes we might not even have a single training example. We propose a novel algorithm for this zero-shot learning scenario. Our algorithm uses side information, such as attributes to embed classes in a Euclidean space. We also introduce a function to measure the compatibility between an image and a label. The parameters of this function are learned using a ranking objective. Our algorithm outperforms the state-of-the-art for zero-shot learning. It is flexible and can accommodate other sources of side information such as hierarchies. It also allows for a smooth transition from zero-shot to few-shots learning.

Keywords

Large-Scale Image Classification • Linear SVMs • Stochastic Gradient Descent • Label Embedding • Attributes • Zero-Shot Learning • Few-Shots Learning.

Résumé

La construction d'algorithmes classifiant des images à grande échelle est devenue une tâche essentielle du fait de la difficulté d'effectuer des recherches dans les immenses collections de données visuelles non-etiquetées présentes sur Internet. L'objetif est de classifier des images en fonction de leur contenu pour simplifier la gestion de telles bases de données. La classification d'images à grande échelle est un problème complexe, de par l'importance de la taille des ensembles de données, tant en nombre d'images qu'en nombre de classes. Certaines de ces classes sont dites "fine-grained" (sémantiquement proches les unes des autres) et peuvent même ne contenir aucun représentant étiqueté. Dans cette thèse, nous utilisons des représentations à l'état de l'art d'images et nous concentrons sur des méthodes d'apprentissage efficaces. Nos contributions sont (1) un banc d'essai d'algorithmes d'apprentissage pour la classification à grande échelle et (2) un nouvel algorithme basé sur l'incorporation d'étiquettes pour apprendre sur des données peu abondantes.

En premier lieu, nous introduisons un banc d'essai d'algorithmes d'apprentissage pour la classification à grande échelle, dans un cadre entièrement supervisé. Il compare plusieurs fonctions objectifs pour apprendre des classifieurs linéaires, tels que "un contre tous", "multiclasse", "classement", "classement avec pondération" par descente de gradient stochastique. Ce banc d'essai se conclut en un ensemble de recommandations pour la classification à grande échelle. Avec une simple repondération des données, la stratégie "un contre tous" donne des performances meilleures que toutes les autres. Par ailleurs, en apprentissage en ligne, un pas d'apprentissage assez petit s'avère suffisant pour obtenir des résultats au niveau de l'état de l'art. Enfin, l'arrêt prématuré de la descente de gradient stochastique introduit une régularisation qui améliore la vitesse d'entraînement ainsi que la capacité de régularisation.

Deuxièmement, face à des milliers de classes, il est parfois difficile de rassembler suffisamment de données d'entraînement pour chacune des classes. En particulier, certaines classes peuvent être entièrement dénuées d'exemples. En conséquence, nous proposons un nouvel algorithme adapté à ce scénario d'apprentissage dit "zero-shot". Notre algorithme utilise des données parallèles, comme les attributs, pour incorporer les classes dans un espace euclidien. Nous introduisons par ailleurs une fonction pour mesurer la compatibilité entre image et étiquette. Les paramètres de cette fonction sont appris en utilisant un objectif de type "ranking". Notre algorithme dépasse l'état de l'art pour l'apprentissage "zero-shot", et fait preuve d'une grande flexibilité en permettant d'incorporer d'autres sources d'information parallèle, comme des hiérarchies. Il permet en outre une transition sans heurt du cas "zero-shot" au cas où peu d'exemples sont disponibles.

Mots-clés

Classification d'image à grande échelle, Séparatrices à Vastes Marges linéaires, Descente de gradient stochastique, Incorporation d'étiquettes, apprentissage "Zero-shot", apprentissage "few-shots".

Acknowledgements

I would like to thank my supervisors Dr. Cordelia Schmid and Dr. Florent Perronnin for their tremendous help and patience in every stage of my PhD. Apart from the ocean of knowledge they generously shared with me, I am grateful for all their insights and efforts that lead to all the good work we produced together. I am leaving this institution with a huge backpack of knowledge, tools and hope that will guide me in the future.

I would also like to thank Dr. Zaid Harchaoui for his valuable contribution to all the papers we published in these three years, to Dr. Georges Quenot, Dr. Christoph Lampert, Dr. Vittorio Ferrari and Dr. Matthieu Cord for agreeing to be a part of my jury, to Nathalie Guillot for helping me with the administrative work, to Mattis Paulin for translating the abstract to French, to all my friends and colleagues in INRIA and XRCE for all the nice moments we shared in these three years. It was a privilege to be around so many smart, motivated, understanding, helpful and awesome people.

Finally I would like to thank my parents Aynur, Erol and my sister Elif for their constant support and love. I am very lucky to have them beside me.

Contents

Abstract v

Résumé vii

Acknowledgements ix

List of Figures xiii

List of Tables xv

1 Introduction 1
- 1.1 Goals . . . 3
- 1.2 Context . . . 5
- 1.3 Contributions . . . 9

2 Related Work 13
- 2.1 Image Description . . . 14
  - 2.1.1 Local Features . . . 16
  - 2.1.2 Coding and Pooling . . . 17
  - 2.1.3 Improving High-Level Image Descriptors . . . 22
  - 2.1.4 Compression . . . 25
- 2.2 Learning . . . 26
  - 2.2.1 Large-Scale Learning . . . 27
  - 2.2.2 Learning with Scarce Training Data . . . 29

3 Good Practice in Large Scale Learning 33
- 3.1 Introduction . . . 34
- 3.2 Related Work . . . 36
- 3.3 Objective Functions . . . 37
- 3.4 Optimization . . . 40
- 3.5 Experiments . . . 43
  - 3.5.1 Fine Grained Experiments . . . 45
  - 3.5.2 Large-Scale Experiments . . . 57

3.6 Conclusion 63

4 Label Embedding for Image Classification **65**
4.1 Introduction 66
4.2 Related Work 68
4.3 Label Embedding with Attributes 70
4.3.1 Framework 71
4.3.2 Parameter learning 73
4.3.3 Beyond attributes 75
4.4 Experiments 76
4.4.1 Zero-Shot Learning 78
4.4.2 Few-Shots Learning 84
4.4.3 All Data Experiments 86
4.5 Conclusion 92

5 Conclusion **95**
5.1 Contributions 95
5.2 Future Directions 97

Publications **I**

Bibliography **III**

List of Figures

1.1 Accessing Big Data 2
1.2 The Image Classification Pipeline 4
1.3 Evolution of Large Scale Datasets 6
1.4 Large Scale Learning Cube 7
1.5 The ILSVCR10 Dataset for Large Scale Image Classification 8

2.1 Global Image Histograms as image descriptors 14
2.2 Image Descriptor Extraction Pipeline based on Patch-Level Statistics . . . 15
2.3 The Scale Invariant Feature Transform (SIFT) 17
2.4 The Bag of Visual Words 18
2.5 ℓ_2 Normalization of FV 22
2.6 Power Normalization of FV 23
2.7 The Spatial Pyramid Matching (SPM) 24
2.8 Product Quantization (PQ) 26
2.9 Zero-Shot and Few-Shots Learning 30

3.1 Comparison of SGD and LibSVM for w-OVR 46
3.2 Comparison of SGD, SVM^{light} and LibLinear for MUL 47
3.3 Influence of data rebalancing in w-OVR 49
3.4 Impact of regularization on w-OVR 50
3.5 w-OVR, MUL, RNK and WAR on Fine Grained Datasets 52
3.6 Synthetic dataset with $\sigma = 0.25$ 54
3.7 Experiments on dataset density for Fungus134 55
3.8 Influence of Image Descriptor Dimensionality 56
3.9 Impact of Imbalance on ILSVRC 2010. 58
3.10 Impact of regularization on the ILSVRC 2010 dataset 59
3.11 Impact of Dictionary Size on the ILSVRC 2010 dataset 60
3.12 Qualitative results on ImageNet10K 61
3.13 Per class Top-1 Accuracy on ImageNet10K 62

4.1 Demonstration of label embedding using output codes 67
4.2 Accuracy as a function of the label embedding dimensionality 82
4.3 Qualitative results with attribute classification accuracy on ALE 83

4.4 Accuracy as a function of the number of training samples per class 85
4.5 Incremental learning using 1/4, 2/4, 3/4 and all the training data 90
4.6 Comparison of OVR, OVA, ALE, GLE, HDLE embeddings on full data . 91
4.7 Sparsity experiments using ridge regression formulation with Gaussian and Hadamard embeddings 92

List of Tables

3.1 Sampling and update equations 41
3.2 Datasets used in the experiments 44
3.3 Average training time for batch and SGD solvers 47
3.4 Sampling with and without replacement for w-OVR. 51
3.5 Influence of Training Data Size 53
3.6 RNK method in comparison with MUL,WAR and w-OVR 53
3.7 w-OVR, MUL, RNK and WAR on Syntethic Data 54
3.8 MUL with SGD, LibLinear and SVM^{light} on Syntethic Data 55
3.9 u-OVR, MUL, RNK and MUL on the ILSVRC 2010 dataset 59
3.10 u-OVR, w-OVR, MUL, RNK and WAR on ImageNet10K. 61

4.1 Comparison of the DAP and ALE approach on zero-shot learning 79
4.2 Comparison of the continuous and binary embeddings on zero-shot learning 80
4.3 Comparison of different learning algorithms for ALE in zero-shot learning 80
4.4 Comparison of ALE and HLE 82
4.5 Comparison of the DAP and ALE approach on zero-shot learning 83
4.6 Comparison of binary and continuous attributes on all data 87
4.7 Comparison of different learning algorithms on the full datasets 87
4.8 Incremental learning using 1/4, 2/4, 3/4 and all the training data 89

1

Introduction

Contents

1.1 Goals 3

1.2 Context 5

1.3 Contributions 9

Before the digital age, the data was produced as textual documents and a collection of these documents were stored in libraries, official and private archives Russell (1946). With the emergence of computers, firstly the text moved to digital format Arms (1996) and this brought down the search time to seconds. Over the past decade, the rapid decrease in the cost of digital cameras produced large amounts of multimedia data in the form of personal multimedia collections. With the emergence of the Internet and social networks such as Facebook, Flickr or Youtube, this visual data was easily shared with others which led to gigantic visual repositories. However, how to store, process and access such big data (see Figure 1.1) are challenging problems that need to be addressed.

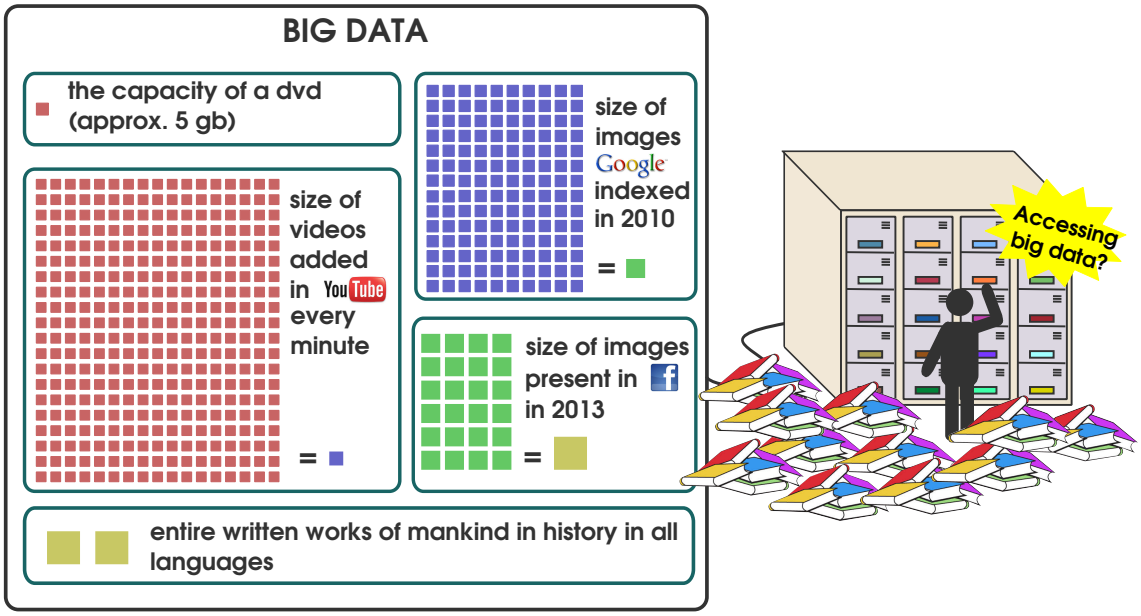


Figure 1.1 *Accessing Big Data?: The capacity of a DVD is taken as 5GB in our calculations (each red box represents 5GB of storage capacity). Accordingly, one blue box represents 1750 GB of storage, one green box represents 250 TB and one yellow box represents 6 thousand TB (6 petabytes). It is difficult to manage such exponential growth since the traditional approaches can not scale to the level needed to be able to ingest all the available data.*

In order to account for the speed of growth in such visual repositories, it is worth to note some recent statistics. For instance, when Google Images[1] was first launched in 2001, it indexed 250 million images. In 2005 this number was over 1 billion and in 2010 it increased to 10 billion [2]. This rapid increase in the available multimedia data brought along a huge storage requirement (see Figure1.1). Therefore, a popular photo sharing

[1]Google Images: http://images.google.com/

[2]http://googleblog.blogspot.fr/2010/07/ooh-ahh-google-images-presents-nicer.html

website Flickr[3] which contained 6 billion images in August 2011 has extended its per user storage space to 1TB in May 2013[4].

The same trend holds for Facebook[5] which is one of the most popular social networking websites. In May 2012, approximately 58 photos were being uploaded to Facebook through Instagram[6] every second[7]. The 250+ billion images available in Facebook today takes ≈6 petabytes of memory (see Figure 1.1). This corresponds to half of the written works that had been produced by mankind in all languages throughout the history.

Managing this rapidly increasing data manually is impossible. This necessitates automatic data organization methods since computers can efficiently process large quantities of data. Computer vision as a scientific discipline develops methods for acquiring, processing, analyzing, and understanding visual data from the real world to produce information in the form of decisions. This is an inverse problem, which seeks to recover some unknowns (*i.e.* classification parameters) given insufficient information (*i.e.* lack of labeled data) to fully specify the solution (*i.e.* scene semantics) Szeliski (2010).

Such a problem is even more complex in the large scale setting, as the number of images is becoming prohibitive, *i.e.* in order of millions. Moreover, the number of the semantic categories we want to analyze is equally high, *i.e.* in order of thousands. The exponential growth in data makes the traditional classification approaches difficult to be applied to today's data collections. They can not scale to the level needed to ingest all of the data, analyze it at the speed it arrives, and store the relevant information for extended periods of time. However, time-to-information is critical to derive maximum value from the unclassified data. It is not feasible to employ a system that takes weeks to analyze the data, since in real case scenarios, information is needed immediately [8]. Therefore, the focus of this thesis is to classify the data in such large scale image collections efficiently.

1.1 Goals

Indexing multimedia data based on its content simplifies its manageability and leads to efficiency in search. According to the type of the query, content based image search is grouped into two categories: (1) query-by-example and (2) query-by-text image search. In query-by-example search, given an image the task is to retrieve similar images Flickner et al. (1995); Smeulders et al. (2000); Datta et al. (2008). The image index associated

[3] Flickr: http://www.flickr.com/
[4] http://blog.flickr.net/en/2013/05/20/a-better-brighter-flickr/
[5] https://www.facebook.com/
[6] http://instagram.com/
[7] http://www.digitalbuzzblog.com/infographic-instagram-stats/
[8] See the August 31, 2012 event of FCW Executive Briefings http://semanticommunity.info/AOL_Government/Big_Data_and_the_Government_Enterprise focusing on the emerging challenges, trends, directions, architectures, and solutions for big data.

with this task can be a compact, *e.g.* binary representation such as the hash index. The distance between two hash indexes, *e.g.* measured with the Hamming distance, reflects the perceptual similarity between images. In query-by-text image search, a set of keywords that describe the visual content are used as image indexes. The image search engines rely mostly on textual keywords Grangier et al. (2006); Grangier and Bengio (2008); Li et al. (2011) found in captions and nearby text, augmented by user click-through data Joachims (2002); Craswell and Szummer (2007). Given this set of keywords, *e.g.* an object name, the aim is to retrieve images containing that particular object in the scene. The process of learning the mapping between the set of keywords and the images is referred to as image classification, annotation or categorization which is the focus of this thesis.

The objective of image classification is to assign one or multiple labels to an image that describe its content. More formally, given an image descriptor $\mathbf{x} \in X$ and a set of labels $Y = \{y_1, ..., y_c\}$ the goal is to learn a prediction function $f : X \rightarrow [0, 1]^c$ that predicts the presence or absence of each label. The standard image classification pipeline (Figure 1.2) has two major steps (1) image description and (2) image classification.

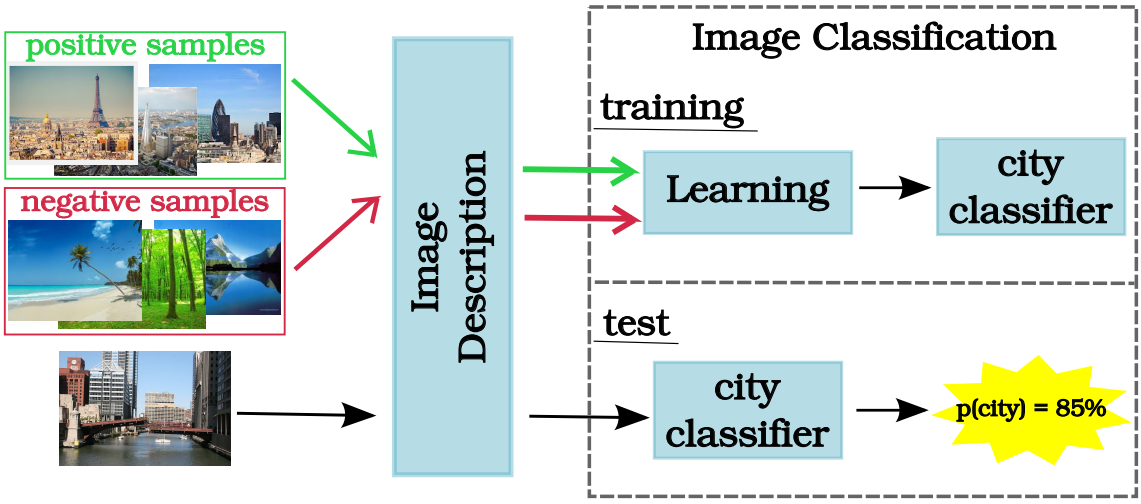


Figure 1.2 *Image classification pipeline consists of two steps: (1) The first step, image description, involves mapping the image content into a descriptor and (2) the second step, image classification, involves learning a classification function which differentiates between positive and negative examples.*

Image description transforms the low-level pixel information into a representation that is more suitable for learning a decision boundary. The ideal image description should have several properties. It should be descriptive, *i.e.* it should be informative enough to reflect the richness of the visual content. It should be robust to scene variations, *i.e.* it should be able to handle changes in viewpoint or illumination, and to synthetic variations, *i.e.* image compression or a change of the resolution. Finally, it should be efficient, *i.e.* it should be cheap to compute and manipulate.

As for image classification, the aim is to learn a decision function in order to separate samples from different classes. We consider the supervised learning problem where labeled data is provided to learn the decision function. This decision function is then used to infer the label of an unknown image. We also consider the situation when the labeled training data is scarce. In this case, side information is used to learn an embedding space which links the images with the class labels and provides a means to predict the label of an unseen image.

In this thesis, we build on the state-of-the-art image descriptors of Csurka et al. (2004); Perronnin and Dance (2007); Jégou et al. (2012) and focus on image classification. Our main goal is to learn the parameters for the class decision functions efficiently for large scale image collections. We benchmark efficient learning methods for semantic understanding of images in large scale image collections and algorithms to predict image labels in presence of missing training data. The main difficulty arises from the large scale nature of the problem where the datasets are large, the image descriptor dimensionality is large and the annotated training data is restricted.

1.2 Context

The scale of a learning problem can be measured by three dimensions: the number of classes k, the number of images n and the image descriptor dimensionality d (see Figure 1.4). The number of classes k and the number of images n depend on the dataset size. Over time, the evolution of image datasets with respect to the number of classes k is shown in Figure 1.3. While 10 years ago, the available datasets contained $O(10 - 100)$ classes and $O(1,000 - 10,000)$ images, today millions of images and thousands of classes are becoming available.

Currently, the large scale image datasets contain thousands of different concepts (k). For instance one of the largest publicly available image datasets is The ImageNet[9] which today contains 21K classes and 14M images. As the number of classes increases to the order of $O(1,000 - 10,000)$, the annotation process becomes difficult even for humans. For generic classes that are common knowledge, crowdsourcing systems such as AMT[10] offer a fast and cheap solution. On the other hand, for rather specific classes that contain rare objects we often need an expert opinion. Consequently, due to the high cost of image annotation and the unlimited variability of such concepts, the large scale image datasets often lack annotations. Learning the decision boundaries with the shortage of labeled training data is one of the key challenges of computer vision that requires attention.

[9]The ImageNet project http://www.image-net.org/ aims to collect images from the internet and annotate them with the 80K synsets of the WordNet hierarchy Miller (1995).

[10]Amazon Mechanical Turk (AMT) tool uses the crowd knowledge to annotate images in large datasets such as the ImageNet: https://www.mturk.com/mturk/welcome

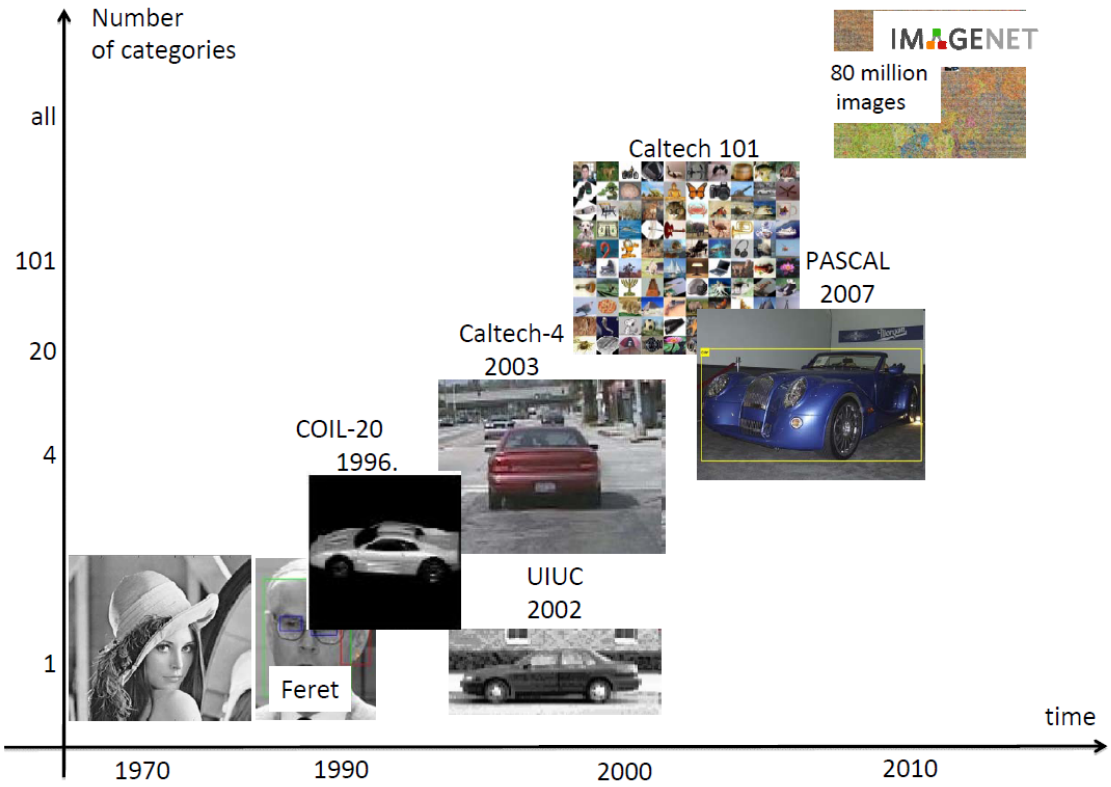


Figure 1.3 *Evolution of Large Scale Datasets over time. The limit of the vertical axis,* i.e. *"all" represents all the 80,000 nouns in the English language as it is indexed in the WordNet hierarchy. The ImageNet project aims to build an image database that contains representative images for all these classes. Figure from Forsyth et al. (2011)*

In order to avoid the complexity of non-linear classifiers, state-of-the-art large scale algorithms rely on high dimensional image descriptors. Such descriptors are constructed by mapping the local features extracted from an image to a high dimensional space in which their high order statistics are encoded with a visual vocabulary. The Fisher Vector (FV) Perronnin and Dance (2007) is a representative of high dimensional image descriptors which has been shown to yield excellent results in combination with linear classifiers Chatfield et al. (2011). Shortly, in combination with efficient linear classifiers, the dimensionality of image descriptors d is often in the order of $O(100K - 1000K)$ for large scale learning.

We now detail some of the challenges associated with large scale learning with respect to the number of classes k, the descriptor dimensionality d and the number of images n.

Number of classes (k). Increasing the number of classes k not only makes the problem computationally more intensive, but it also causes the problem to be more difficult. In order to illustrate this difficulty, Figure 1.5 shows the hierarchical relationships between

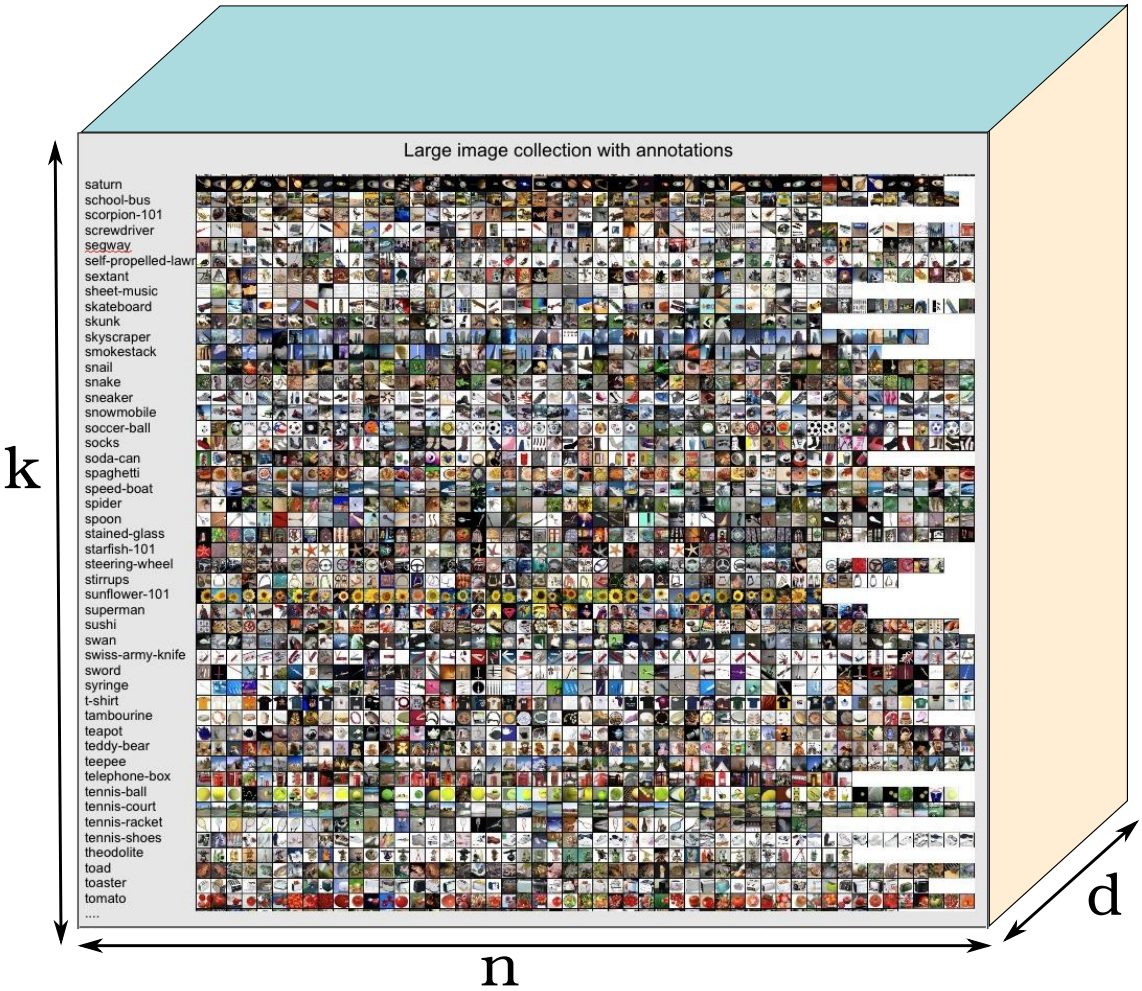


Figure 1.4 *Large Scale Learning Cube: The number of classes (k), the number of images (n) and the number of entries in image descriptors (d) determine the scale of the image classification problem.*

classes in The ImageNet Large Scale Visual Recognition Challenge dataset from 2010 (The ILSVRC 2010 dataset) [11]. In such datasets, the difficulty of recognition varies significantly for different parts of the semantic space. As shown in Figure 1.5 the large scale image datasets contain a wide variety of concept classes (*i.e.* $k = 1,000$) such as "vessel", "flower", "tree" *etc*. that are easy to distinguish. However, as the number of classes increases, the class density increases. As a result, the distance between the classes in the semantic space becomes smaller Deng et al. (2010). For instance, the lower branch of the ILSVRC10 dataset shown in Figure 1.5 has several "orchid" classes which can be distinguished by domain experts according to the details in their appearance. In this case, if the

[11] http://www.image-net.org/challenges/LSVRC/2010/index

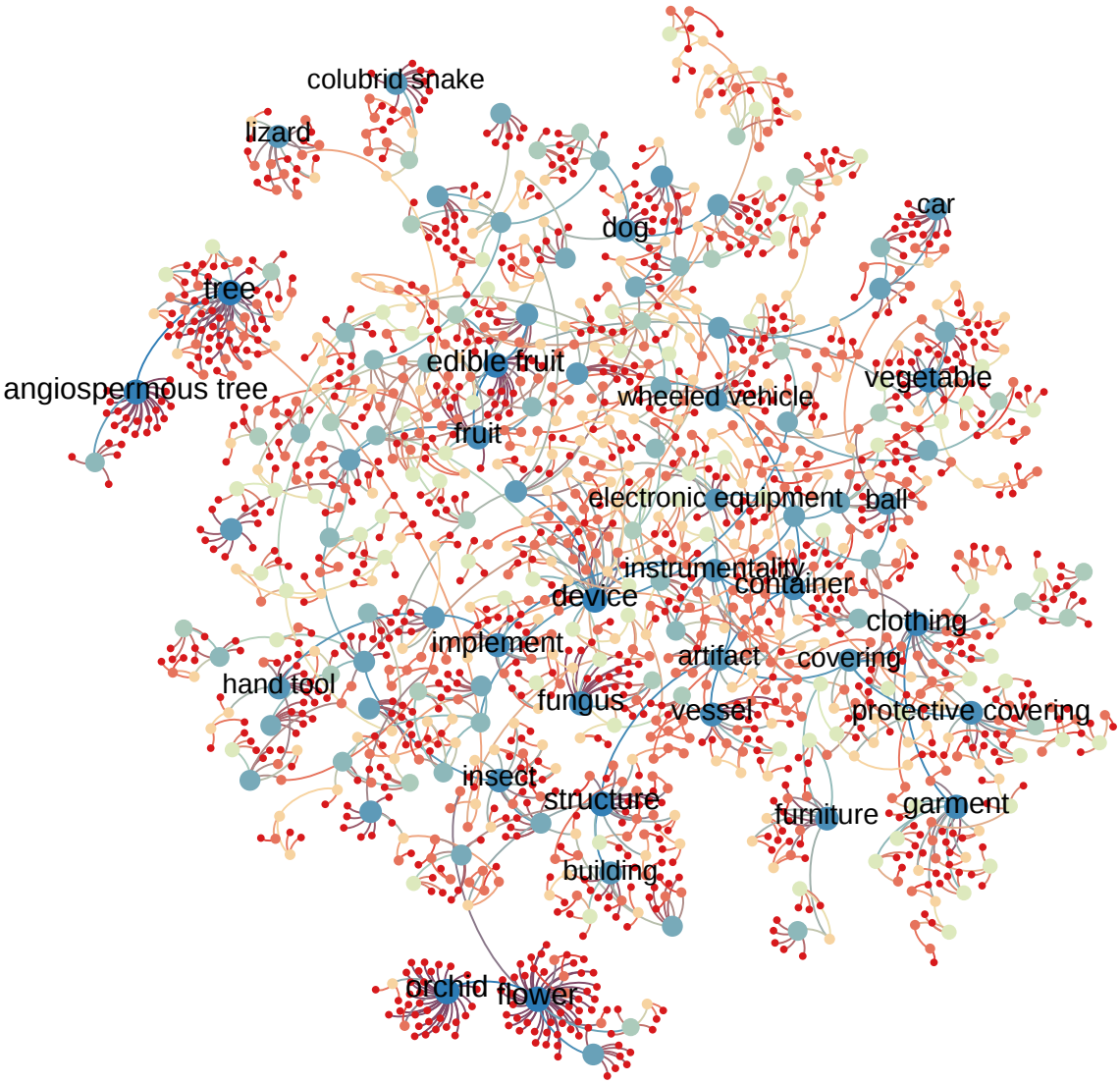


Figure 1.5 *Illustration of the hierarchical relations for the ImageNet Challenge Dataset of 2010 which contains $k = 1K$ classes and $n = 1.2M$ images. The Figure shows the hierarchical relationships between 1,000 classes with their ancestors upto the root node "entity".*

image classification system labels an image of "butterfly orchid" as "flower" or "orchid", that is evaluated as a mislabeled image.

Developing generic image classification algorithms that can cope with the difficulty caused by a large k at a reasonable computational cost is one of the challenges we address in this thesis.

Descriptor dimensionality (d). To build robust classifiers for large scale datasets, we need more expressive, *i.e.* higher dimensional descriptors Sánchez and Perronnin (2011),

especially when used in conjunction with linear classifiers. However, the increasing descriptor dimensionality d comes with high memory and computational cost. One possible research direction to address this challenge is through image descriptor compression Weinberger et al. (2009); Sánchez and Perronnin (2011); Jégou et al. (2011). Nevertheless, this is not the focus of this thesis.

Number of images (n). The predictive ability of supervised image classification systems increases with the number of labeled training images. Yet, processing all the training image descriptors for large scale datasets requires large computational resources. For instance, since the dimensionality of image descriptors is in the order of $O(100,000)$, and if each entry contains a floating point which is represented with 8 bytes, one image descriptor occupies $\approx$ 0.8MB in the memory. Consequently, the training images of the ILSVRC10 dataset would not fit in the RAM at once. For this reason, learning the decision boundaries using millions of images for training is a computational challenge, which we address in this thesis.

Another challenge that we face is the lack of labeled images for training the classifiers. For generic classes such as "flower" or "tree", crowdsourcing is an efficient method to obtain annotations. However, for more granular classes such as "spider orchid", the labeled training data is rare. Moreover, labeling the data is a costly process. In such cases, a sufficient amount of labeled training images is not always affordable.

In one extreme scenario, the aim is to predict the label(s) for the classes for which the system has not seen a single labeled example. This is known as zero shot learning Larochelle et al. (2008) problem. Given a semantic relationship between classes, the aim in zero-shot learning is to build a classifier to recognize classes that are omitted from the training set. In the context of computer vision, zero-shot learning studies the image classification problem when training and test classes are disjoint Lampert et al. (2009).

Despite the difficulty of obtaining the labeled data, the number of annotated images in the large scale image databases has been increasing rapidly. Accordingly, the image classification system should be able to adapt itself to the situation when some training data from the unseen classes becomes available. Such an incremental learning scheme which integrates new images in the learning process is called the few shots learning problem Tang et al. (2010). Zero-shot and few-shots learning are two of the challenges that are addressed in this thesis.

1.3 Contributions

The focus of this thesis is to classify images on large-scale and potentially fine grained datasets. We develop robust learning methods that are efficient for large amounts of data

as well as an approach that can cope with the lack of training data. In our setting, the number of classes (k) goes up to 10K, the number of images (n) goes up to 9M and the descriptor dimensionality (d) goes up to 130K. We provide experimental evidence to validate our solutions with respect to the following questions: (1) How do different learning methods perform for classification of a large scale dataset and what are the best strategies to improve the state-of-the-art image classification accuracy for large scale image datasets? (2) How can we cope with learning for classes for which no or minimal labeled training samples are provided? Our contributions towards answering these questions are described in the following paragraphs.

- We propose a benchmark for learning algorithms for large scale image classification in the fully supervised setting. It compares several objective functions for learning the classifier parameters of linear SVMs such as one-vs-rest, multiclass, ranking and weighted average ranking. We compare online and batch methods to optimize these objectives. However, for computational efficiency reasons, we focus on online optimization using the Stochastic Gradient Descent.

  The goal of this benchmark is to analyze the difficulties of large scale image classification with respect to each dimension, namely k, n and d. Therefore, we compare the aforementioned objective functions for varying the number of classes k, the number of images n and the dimensionality of the image descriptors d. Moreover, we investigate the effects of the parameters, *e.g.* regularization and step-size on an online learning scheme. We conclude that, despite its theoretical suboptimality, one-vs-rest is a very competitive training strategy to learn SVMs. Furthermore, it is easy to implement and to parallelize. Our second conclusion is that stochastic, *i.e.* online training is very well suited to the large-scale setting. Moreover simple strategies such as implicit regularization with early stopping and fixed-step-size in online learning work well in practice.

  Our experimental validation is performed on two large scale datasets, ImageNet with 10K classes and ImageNet with 1K classes as well as three fine grained subsets of ImageNet, namely Fungus134, Ungulate183 and Vehicles262. To the best of our knowledge, there is no previous work in the literature comparing the linear SVMs with one-vs-rest, multiclass, ranking and weighted average ranking formulations in such a large scale setting. The approach obtained good results in the XRCE-INRIA participation of the large scale image categorization challenge of the ILSVRC2012. This work was published in Perronnin et al. (2012); Akata et al. (2013b) and is described in Chapter 3.

- Our second contribution is a novel approach for classifying images when the annotations are scarce; that is when for some classes there is little or no training data available. To overcome the difficulty of learning in such a context, *i.e.* zero-shot learning, the attributes act as intermediate representations that enable parameter

sharing between classes. Our approach, Attribute Label Embedding (ALE), consists in embedding the classes in a Euclidean space using side information such as attributes, hierarchies *etc*. ALE has several advantages with respect to the standard DAP approach of Lampert et al. (2009): (1) It gets a higher accuracy, (2) it is not restricted to attributes, any other sources of side information can be used, and (3) it allows a smooth transition from zero-shot learning to learning with large quantities of data.

The experimental validation of this contribution is performed on Animals with Attributes dataset which contains 40 classes with 85 attributes associated with each class and Caltech UCSD 2011 Birds dataset which contains 200 classes with 312 attributes. Apart from the attributes, we build side information using the hierarchical relationships between the synsets in the WordNet hierarchy and we use Error Correcting Output Codes as side information. The details of this project are discussed in Chapter 4 which summarizes the publications Akata et al. (2013a) and Akata et al. (2013c).

2

Related Work

Contents

2.1 Image Description **14**

2.1.1 Local Features 16

2.1.2 Coding and Pooling 17

2.1.3 Improving High-Level Image Descriptors 22

2.1.4 Compression 25

2.2 Learning **26**

2.2.1 Large-Scale Learning 27

2.2.2 Learning with Scarce Training Data 29

It is trivial for humans to classify up to 20,000 object classes accurately Biederman (1987). However, all the automatic image classification systems fail at reaching human accuracy when the number of classes goes up to and above 1,000 object classes. In this chapter, we investigate the components of an image classification system and provide the related work regarding our main goal of large scale image classification.

Image classification consists in assigning one or multiple labels to an image based on its content. The image classification pipeline is composed of two steps: (1) Image description and (2) image classification. In image description, we create a high-level descriptor $\mathbf{x} \in \mathcal{X}$ for an image that represents its content. In image classification, given a set of image descriptors $\mathcal{X} = \{\mathbf{x}_1, ..., \mathbf{x}_N\}$ and a set of labels $Y = \{y_1, ..., y_c\}$, we learn a prediction function $f : \mathcal{X} \rightarrow [0, 1]^c$ that predicts the presence or absence of each label for an image.

The outline of this chapter is as follows. Section 2.1 reviews the algorithms used in the image description step to translate images into image descriptors. Section 2.2 reviews the machine learning algorithms developed for learning classifier parameters and for predicting class labels of an unknown image.

2.1 Image Description

The choice of image descriptors is one of the main factors that impacts the accuracy of an image classification system Parikh and Zitnick (2010). In this section, a detailed analysis of the available image features and descriptors that are suitable for large scale image datasets is provided. The desired properties of the image descriptor extraction algorithms are robustness, computational efficiency and high level descriptiveness of the image content. The image itself, as humans perceive it, has all the essential information about the content of the image. However, as computers perceive it, the image itself contains only low-level information about the individual pixels. Image descriptors are extracted from an image to bridge the gap between low-level information and high level concepts. Image descriptors can be broadly classified into two categories: global and local image descriptors.

Pixel-Level (Global) Image Descriptors. Early works described the images using global signatures based on the aggregation of pixel-level statistics. The global gray-scale image histogram is an example of such an image representation. It counts the number of times a certain pixel value appears in an image. Therefore, image histograms discard the spatial location of the pixel values. Since the image histograms of two conceptually similar images may be very different due to the changes in the background, illumination conditions, viewpoint or the scale, position, orientation of the objects (see Figure 2.1), this representation is not discriminative enough for image classification even at a small scale.

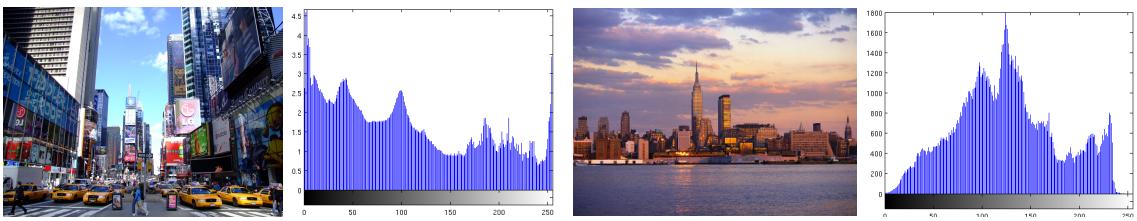


Figure 2.1 *Global Image Histograms are not discriminative enough to describe the image content. This figure shows two completely different histograms for two images that belong to the same class ("city" or "New York City").*

Another example of global image descriptors is the GIST descriptor introduced by Oliva and Torralba (2001). The idea in GIST is to develop a low-dimensional description of the whole image through a set of perceptual dimensions such as naturalness, openness, roughness, expansion and ruggedness. Because it is efficient to compute and because it is low-dimensional – and therefore efficient to match and classify – it has come back in fashion for large-scale visual recognition as shown in Deng et al. (2010). It has also been shown Douze et al. (2009) to perform well in some specific large-scale image classification applications such as commercial copy detection where the aim is to determine

whether a certain image is original or fake. However, GIST descriptors are not invariant to many image transformations such as crops, rotation *etc*. Thus, the descriptive power of such pixel-level image descriptors is limited.

Aggregation of Local Image Descriptors. Patch-based image descriptors (see Figure 2.2) consist in aggregating per image statistics computed from local image patches. We first extract patch-level local features and then aggregate the statistics computed from the local features into a fixed length local image descriptor (see Figure 2.2). The additional steps include normalization and spatial pyramid matching for image descriptor enhancement, and quantization for compression.

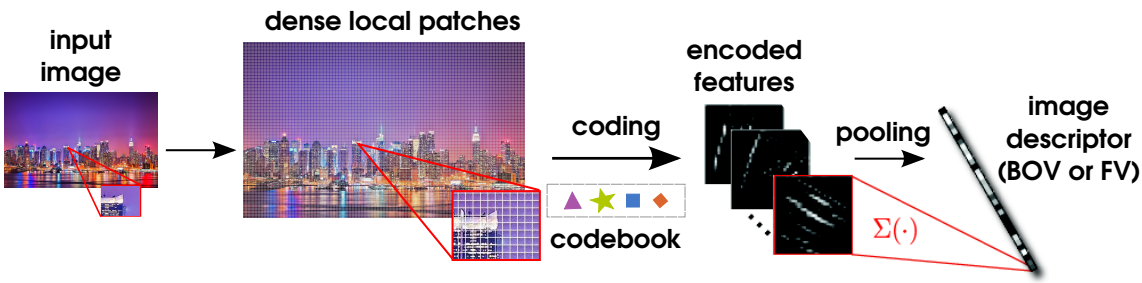


Figure 2.2 *In the state-of-the-art image description pipeline, we select a set of local patches densely from an image,* i.e. *detection. From each of these dense local patches, we extract local image features for a set of images,* i.e. *description. We then cluster these local features to form a visual vocabulary and we assign each local feature to one or more visual words,* i.e. *coding. Finally, we aggregate these encoded local features using the average or max operations,* i.e. *pooling.*

The patch-based image descriptors are used in state-of-the-art image classification systems since: (1) they are more informative about the image content than the individual pixels and (2) the feature detection and description at a patch level enables image descriptors to inherit the invariance properties obtained from the local features. The patch-level image descriptors are used in many applications such as object recognition Ferrari et al. (2006); Fergus et al. (2003), image stitching Brown and Lowe (2003), automatic image Mikolajczyk and Schmid (2001) and video Tuytelaars and Van Gool (2004) annotation *etc*. In this thesis, the experiments have been conducted using such state-of-the-art image descriptors. We present them in detail in the following.

Section 2.1.1 presents the detection and description steps of the local feature extraction pipeline. Section 2.1.2 details the algorithms for coding the local features and pooling them into high level image descriptors. Section 2.1.3 provides the methods for increasing the descriptive power of the high level image descriptors. Section 2.1.4 details the quantization methods that are used to compress the image descriptors.

2.1.1 Local Features

Local features are preferred in practice due to their invariance to certain image transformations such as translation and scaling, *etc*. Local feature extraction is composed of two steps: (1) detection and (2) description.

Detection. The feature detection step determines the number, the size and the location of the patches that are extracted in an image. There are three main methods used for feature detection in the literature: (1) sparse detection based on the interest points Mikolajczyk and Schmid (2002), (2) detection on a dense grid Tuytelaars (2010) and (3) random sampling of the patches Marée et al. (2005). The interest points are detected by searching the descriptive key-points that are invariant to certain image transformations in all scales of an image. Dense sampling extracts patches on a regular grid within the image. In order to have scale invariance, dense patches are extracted at different scales of every image. In random sampling, the patches are sampled at random locations in an image. Among these, dense sampling is the detection method that is used in the state of the art.

Description. In this step, local features are extracted from patches or interest points that were detected in the previous step. Local features can be simple as the intensity or RGB values. However, more descriptive features that have some level of invariance against illumination change or geometric distortions are usually preferred. The description step is briefly reviewed as follows, with an emphasis on the SIFT and color features which are used in the experimental evaluation of this thesis.

- **SIFT descriptor.** The SIFT descriptor Lowe (2004) builds a histogram of image gradients within each patch as illustrated on Figure 2.3. It computes 8 orientation directions over a 4×4 grid which results in a $4 \times 4 \times 8 = 128$-dimensional feature vector. Through a Gaussian window function that gives more weight to the gradients computed near the center of the patch, the SIFT descriptor offers robustness to some level of geometric distortion and noise. Also, for robustness to illumination changes, the SIFT descriptor is normalized to one. It is shown to outperform other descriptors in several tasks Mikolajczyk and Schmid (2005).

- **Local RGB Color Descriptors.** In large-scale datasets even a small variation in color plays an important role in distinguishing between classes. There are many ways of extracting color descriptors depending on the color coding used in the images. In this thesis, a patch is divided into $4 \times 4 = 16$ sub-regions, the mean and standard deviation of R,G and B channels are computed in each sub region, *i.e*. 2×3 which results in a 96-dim local RGB color feature.

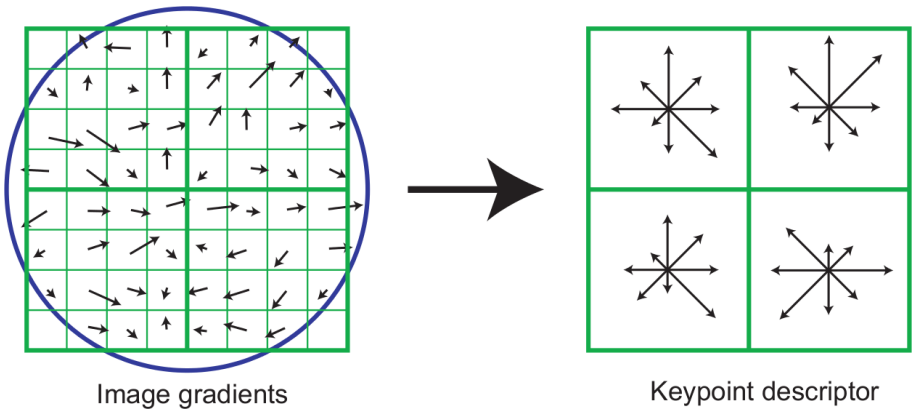


Figure 2.3 *Scale Invariant Feature Transform (SIFT) by Lowe (2004): describes each image patch as a histogram of image gradients measured by the magnitude and orientations of intensity changes in the pixel level. The* 8 *image gradients are calculated at the pixel level (left) and they are accumulated into orientation histograms over* 4×4 *subregions (right) which results in a* 128*-dimensional vector.*

In this thesis, we use the popular SIFT and the local RGB color descriptors. However, apart from them, many other local descriptors exist. Now we review some of the other type of descriptors introduced in the computer vision literature.

- Color SIFT van de Sande et al. (2010) computes SIFT descriptors separately for R,G and B channels.
- Local Self Similarity (LSS) Features Shechtman and Irani (2007) describes an interest point by computing the sum of squared distances between a small patch whose center is the sampled point and other patches from a bigger region.
- Speeded up Robust Features (SURF) Bay et al. (2006) is another scale and rotation invariant local feature extraction algorithm that computes gradients in only two orientations, *i.e.* x and y, and relies on image integral masks to approximate the gradient computation.

2.1.2 Coding and Pooling

The first step of the image description is to compute a set of local features, *e.g.* SIFT. The next step is to cluster this set of local features into visual words that create a visual vocabulary. The local features of an image are then assigned to one or multiple visual words using a similarity measure. The coding step maps an input descriptor to a higher dimensional space through a nonlinear operation. After the coding step, spatial pooling using average or max operations are carried out to aggregate the codes into an image level descriptor.

In the remainder of this chapter, we discuss the two coding techniques to build the global image descriptors that are used in the experimental evaluation of this thesis, (1) Bag of Words and (2) Fisher Vectors.

1) Bag of Visual Words (BOV)

Since it is first mentioned by Harris (1954) in the context of text processing, the Bag of Words model has been widely used in building a systematic description of textual documents and consecutively adapted to computer vision problems by Sivic and Zisserman (2003); Csurka et al. (2004). The Bag of Visual Words (BOV) is essentially a frequency count (*e.g.* histogram) of the local image features that are assigned to the closest visual word (see Figure 2.4). Each image is described as a collection or "bag" of these visual words that make up the visual dictionary. The contribution of each visual word in the BOV descriptor of each image is determined by the number of visual words that are present in the bag. Now we explain the details of BOV algorithm.

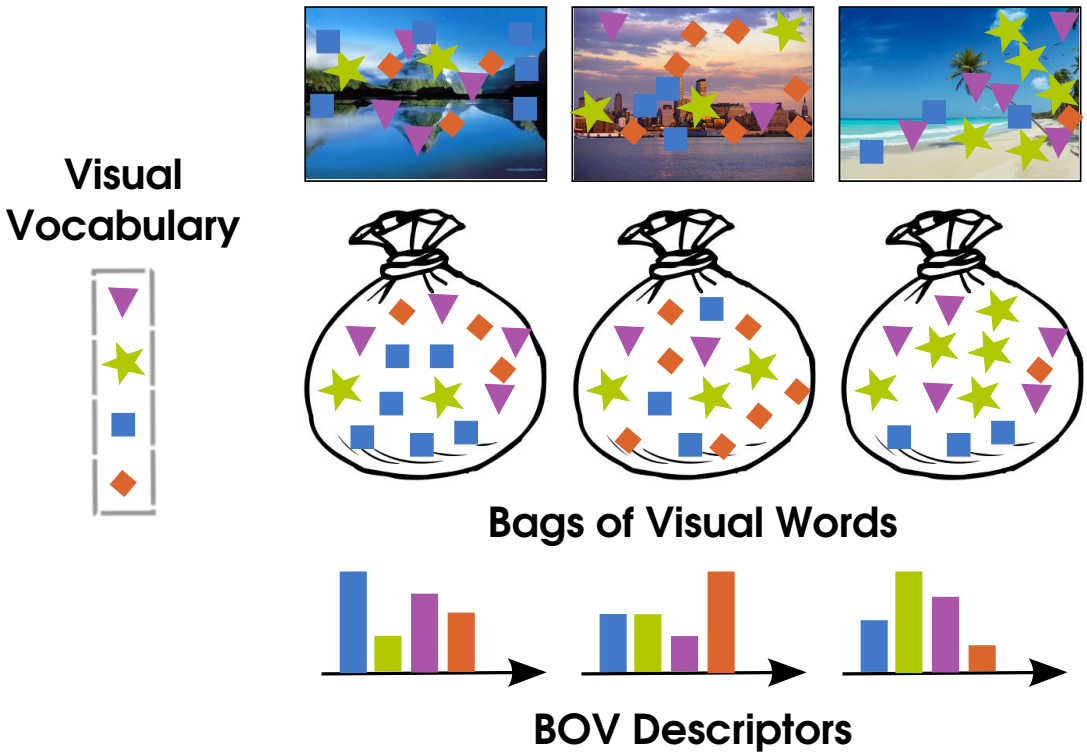


Figure 2.4 *Bag of Visual Words: After extracting local features, a visual vocabulary is created using k-means clustering. Each image is then described as a collection or "bag" of these visual words that make up the visual dictionary. The contribution of each visual word in the BOV descriptor of each images is determined by number of visual words that are present in the bag,* i.e. *a histogram of visual words.*

Creating the Visual Vocabulary. As mentioned earlier, the local features that are extracted from a set of images are clustered to build a visual vocabulary. The most popular clustering method used with BOV is k-means clustering Sivic and Zisserman (2003); Csurka et al. (2004). Given a set of N local image features $\{\mathbf{x}_1, ..., \mathbf{x}_N\} \in \mathbf{R}^D$, to learn a dictionary $\boldsymbol{D} = \{\boldsymbol{d}_1, ..., \boldsymbol{d}_k\}$ of size k, the k-means algorithm aims at minimizing the squared Euclidean distance between each local feature and its nearest visual word. This objective can be formulated as:

$$\min_{\boldsymbol{D}} \sum_{i=1}^{N} \|\mathbf{x}_i - \boldsymbol{d}_{q_i}\|^2 \text{ where } q_i = \arg\min_k \|\mathbf{x}_i - \boldsymbol{d}_k\|^2 \tag{2.1}$$

The clusters must be composed of similar amount of sample points in the end. Here, the cluster assignments of the local features are determined using Vector Quantization (VQ) method of Lloyd (1982).

In the next step where we compute the global image descriptor, *i.e.* creating the histogram of local features, the encoded local features can be pooled in one of two ways: (1) sum-pooling or (2) max-pooling. In the case of sum-pooling, the local encoded features are additively combined into a histogram, which then could be normalized by the number of samples. In the case of max-pooling, each histogram bin is assigned the highest value throughout all of the encoded local features.

The BOV has been the major feature descriptor in many computer vision applications Csurka et al. (2004); Farquhar et al. (2005); Sivic and Zisserman (2009); van Gemert et al. (2010); Bourreau et al. (2010) as it is computationally efficient, simple and easy to implement. As an improvement over the BOV representation, Farquhar et al. (2005); van Gemert et al. (2010) suggest to soft-assign the local descriptors using a generative model built on the descriptors. Another approach is based on sparse coding, which enforces a descriptor to be assigned to a small number of visual words Yang et al. (2009); Bourreau et al. (2010). While sparse-coding has been shown to yield excellent results, especially in combination with max-pooling Bourreau et al. (2010), it involves a costly iterative optimization process. Kernel Codebook Encoding Gemert et al. (2008) proposes to use kernel density estimation algorithm to allow a degree of ambiguity in assigning visual word centroids to image features. Another example to Visual Word Encoding methods is Locality Constrained Linear Encoding (LLC) Wang et al. (2010); Lin et al. (2011). LLC projects each image feature onto its local coordinate system and integrates the projected coordinates by max pooling to generate the final image descriptors.

The descriptive power of the high-level image descriptors increases with the increasing number of dimensions. Therefore, image descriptors such as BOV provides good results in combination with linear classifiers only when they are high dimensional. In order to implicitly increase the dimensionality of the high-level image descriptors, one way is to increase the size of the visual vocabulary. Another way to obtain good results without

using high dimensional descriptors is to use kernels, *e.g.* see the results of the successive PASCAL VOC competitions organized by Everingham et al. (2010). However, one of the limitations of non-linear SVM classifiers is that they do not scale well with the number of training samples. Therefore, several works have proposed to perform an explicit embedding of the image representations in a high dimensional space where the BOV histograms are more linearly separable. Maji and Berg (2009) proposed mappings for the Intersection Kernel (IK) and Wang et al. (2009) then proposed efficient algorithms to learn IK SVMs. Vedaldi and Zisserman (2010) and Perronnin et al. (2010a) subsequently generalized this principle to additive classifiers. Gong and Lazebnik (2011) benchmarked several feature mapping techniques and showed that the data-dependent mappings have an edge over the data-independent ones in large-scale scenarios (see the part on power normalization in the following sections). Other examples of high-level image descriptors are the Vector of Locally Aggregated Descriptors (VLAD) Jégou et al. (2012), the Super Vector (SV) Zhou et al. (2010) and the Fisher Vectors (FV) Perronnin and Dance (2007). Chatfield et al. (2011) benchmarked five local descriptor encoding techniques (BOV with hard coding, soft coding and sparse coding as well as SV and FV coding); the FV Perronnin et al. (2010a) yielded the best results on two standard datasets. Therefore, we mostly focus on the FV but also use the BOV in comparison with FV to show the generality of our conclusions. We explain the FVs in the following.

2) Fisher Vectors (FV)

While the BOV is composed on only the count of visual word occurrences for each local descriptor, other approaches have been proposed that introduce higher-order statistics. This includes the Fisher Vector (FV) Perronnin and Dance (2007) which consists in computing the deviation of a set of local descriptors from an average Gaussian Mixture Model (GMM). In the following, we explain the steps of the FV algorithm.

Creating the Visual Vocabulary. If the codebook is based on a Gaussian Mixture Model (GMM), the posterior probabilities of each Gaussian can be used as weight in the soft-assignment of image features to the visual words. A GMM is a probability density function calculated on the space of image features. The parameters can be learned with the Expectation Maximization (EM) algorithm Dempster et al. (1977). Given a training set of image features $\{\mathbf{x}_1, ..., \mathbf{x}_N\} \in \mathbf{R}^D$, the aim is to learn a dictionary $\boldsymbol{D} = \{d_1, ..., d_K\}$ of size k. Let $\mathbf{p}_\lambda$ to be the probability density function of a GMM:

$$\boldsymbol{p}_\lambda(\mathbf{x}) = \sum_{i=1}^{k} \pi_i \boldsymbol{p}_i(\mathbf{x}) \text{ with } \lambda = \{\pi_i, \boldsymbol{\mu}_i, \Sigma_i, i = 1, ..., k\} \tag{2.2}$$

where π_i, $\boldsymbol{\mu}_i$ and Σ_i are mixture weight, mean vector and covariance matrix of Gaussian $\boldsymbol{d}_i$. With the increasing number of clusters, the vocabulary size increases which leads to a higher performance Chatfield et al. (2011); Perronnin et al. (2012).

Fisher Vector Coding. The Fisher kernel is based on the gradient of the log-likelihood of a set of features on the GMM with respect to its parameters. The gradient of the log-likelihood describes how the parameters contribute to the process of generating a particular example. A local feature extracted from an image, denoted as $\mathbf{x}$ can be described by the gradient vector:

$$G_\lambda^{\mathbf{x}} = \nabla_\lambda \log p(\mathbf{x}|\lambda) \tag{2.3}$$

where p is the probability density function of a GMM and λ are its parameters. The gradients of the likelihood with respect to each parameter measures the contribution of each parameter to the generation process. The Fisher kernel on these gradients Jaakkola and Haussler (1998) is defined as:

$$K(\mathbf{x}, \boldsymbol{y}) = G_\lambda^{\mathbf{x}^T} F_\lambda^{-1} G_\lambda^{\boldsymbol{y}} \tag{2.4}$$

where F_λ is defined as the Fisher information matrix:

$$F_\lambda = E_{x\sim p}[\nabla_\lambda \log p(x|\lambda) \nabla_\lambda \log p(x|\lambda)^T]. \tag{2.5}$$

Since F_λ is symmetric and positive definite, it can be written as $F_\lambda = L_\lambda^T L_\lambda$ and accordingly the Fisher Vectors are defined as:

$$\mathcal{G}_\lambda^{\mathbf{x}} = L_\lambda G_\lambda^{\mathbf{x}}. \tag{2.6}$$

For each $K = 1, ..., k$ cluster centroids (also known as visual words) the following Fisher Vectors with respect to the mixing weights π_i, the means μ_i and standard deviations σ_i of Gaussian i are defined as:

$$\mathcal{G}_{\pi_i}^{\mathbf{x}} = \frac{1}{N\sqrt{\pi_i}}(\gamma(i) - \pi_i), \tag{2.7}$$

$$\mathcal{G}_{\boldsymbol{\mu}_i}^{\mathbf{x}} = \frac{1}{N\sqrt{\pi_i}}\gamma(i)\left(\frac{\mathbf{x} - \boldsymbol{\mu}_i}{\sigma_i}\right), \tag{2.8}$$

$$\mathcal{G}_{\sigma_i}^{\mathbf{x}} = \frac{1}{N\sqrt{2\,\pi_i}}\gamma(i)\left[\frac{(\mathbf{x} - \boldsymbol{\mu}_i)^2}{\sigma_i^2} - 1\right]. \tag{2.9}$$

where $\gamma(i) = p(i|x)$ is the soft assignment of feature x to Gaussian i. To aggregate the codes of different patches, Perronnin et al. (2010b) proposes to average them which corresponds to an independence (iid) assumption. For instance, when we consider the set of all the local features extracted from an image (*i.e.* $X = \{x_t, t = 1, ..., N\}$) and average them based on the mixing weights (*i.e.* $\mathcal{G}_{\pi_i}^{\mathbf{x}}$) the equation can be written as:

$$\mathcal{G}_{\pi_i}^{X} = \frac{1}{N\sqrt{\pi_i}}\sum_t(\gamma_t(i) - \pi_t), \tag{2.10}$$

which represents the 0-order word count that shows the number of features to be assigned to a certain Gaussian. This is the soft BOV descriptor minus the mixing weights and normalized by the frequency. Accordingly, the final FV descriptors are defined as the concatenation of the gradients calculated using the mean μ_i and standard deviation σ_i for $i = 1, ..., k$. The FV leads to a high dimensional descriptor[1] despite few mixing Gaussians are used at low cost.

2.1.3 Improving High-Level Image Descriptors

This section explains ℓ_2 normalization, power normalization and spatial pyramid matching for improving the FV. Our discussion is based on the FV but these improvements can be extended to other image descriptors.

ℓ_2 Normalization. Normalizing the FV improves its descriptive power Perronnin et al. (2010b). Here we provide two explanations supporting ℓ_2 normalization. As discussed in Perronnin et al. (2010b), the Fisher Vectors approximately discard the image-independent (background) information but focus on the image specific (foreground) information. Consequently, two images containing the same object with different sizes (see Figure 2.5) will have different FV signatures. To remove the dependence of the proportion of the image specific information, we ℓ_2 normalize the FV.

Figure 2.5 *Although the background information is approximately discarded from the the FV of Perronnin et al. (2010b), it is depends on the foreground information. In this figure, all the images contain horses but the proportion of the information that depicts the "horse" is different for each image. Consequently, they would have different FV signatures. To remove the dependence on the image specific information, we ℓ_2 normalize the FV.*

The second interpretation Sánchez et al. (2013) is valid for any high dimensional vector. Under the assumption that high dimensional vectors are uniformly distributed over the unit sphere, the marginals over the ℓ_2 normalized coordinates are approximately Gaussian.

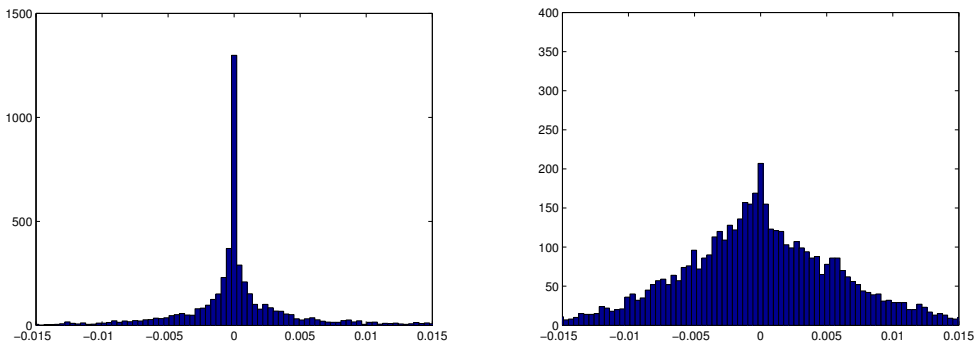
[1]The final dimensionality of FV is $2kD$ where D is the dimensionality of the local features (*e.g.* $D = 64$ in the setting with 128dim SIFT + PCA) and k is the number of Gaussians (*e.g.* typically $k = 256$ for the large scale experiments reported in the following chapters).

Moreover, it has been suggested that the ℓ_2 metric is a good metric between data points if they are distributed according to a generalized Gaussian. Since in large scale learning we rely on linear SVMs where the similarity between the samples is measured using simple dot-products and the dot-product is equivalent to ℓ_2 metric for ℓ_2 normalized vectors, ℓ_2 normalization of high dimensional vectors such as the FVs is a natural choice.

Power Normalization. As the number of cluster centroids that construct the visual vocabulary increases, fewer local features are assigned to some Gaussians, therefore the resulting Fisher Vectors become sparser. As the number of Gaussians K increases, the distribution of the features in a given dimension becomes more peaky around zero (see Figure 2.6). In other words, some of the visual words will not be represented in the Fisher Vectors. Hence, Perronnin et al. (2010b) proposed to perform a power normalization:

$$f(z) = sign(z)|z|^{\alpha} \text{ with } 0 < \alpha < 1 \tag{2.11}$$

for each dimension of the FV independently. In our experiments, we follow the choice of Perronnin et al. (2010b) and take $\alpha = 0.5$, therefore we can refer to this normalization as "square-rooting". This operation can be viewed as an explicit data embedding Sánchez et al. (2013).



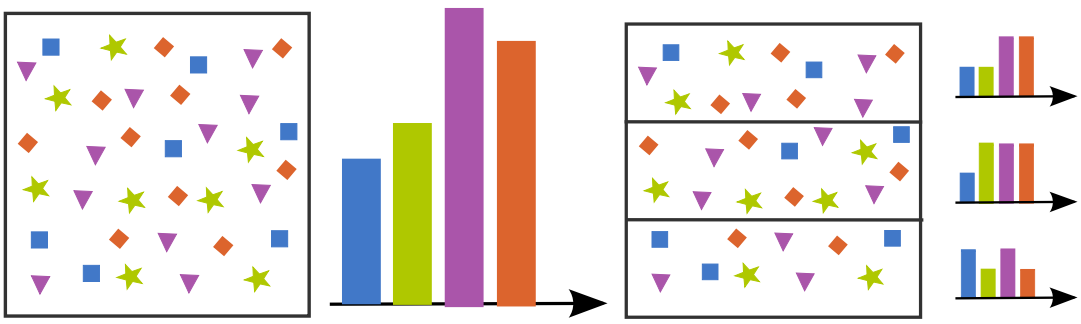
(a) no normalization (b) with power normalization

Figure 2.6 *This figure shows the distribution of the values in the first dimension of the L2 normalized FV constructed using 256 Gaussians (Perronnin et al. (2010b)). Figure on the left the values without power normalization and Figure on the right shows the effect of power normalization. Note that power normalization makes the Fisher Vector sparser.*

A more formal justification was provided in Jégou et al. (2012). The FVs can be viewed as emissions of a compound distribution whose variance depends on the mean. However, when using metrics such as the dot-product or the Euclidean distance, the implicit assumption is that the variance is stabilized, i.e. that it does not depend on the mean. It

was shown in Jégou et al. (2012) that the square-rooting had such a stabilization effect. All of the above papers acknowledge the incorrect patch independence assumption and try to correct a posteriori for the negative effects of this assumption. In contrast, Cinbis et al. (2012) proposed to go beyond this independence assumption by introducing an exchangeable model which ties all local descriptors together by means of latent variables that represent the GMM parameters. Such a model leads to discounting transformations in the FV similar to the simpler square-root transform, and with a comparable positive impact on performance. We finally note that the use of the square-root transform is not specific to the FV and is also beneficial to the BoV as shown for instance by Perronnin et al. (2010a), Vedaldi and Zisserman (2010), Winn et al. (2005).

Spatial Pyramid Matching. A weakness of the FV or the BOV descriptors is that, they are agnostic to the geometric correspondences between images. To incorporate more information regarding the structure of the scene, Spatial Pyramid Matching (SPM) is proposed by Lazebnik et al. (2006). The SPM partitions the image into increasingly fine sub regions and computes the image descriptors (*e.g.* BOV, FV *etc.*) for each sub-region, *i.e.* the local statistics are pooled at a region level. These image descriptors that encode the spatial structure of the image (see Figure 2.7) are then concatenated to form a global image representation.



(a) no SPM (b) with SPM

Figure 2.7 *In the figure, four feature types,* i.e. *squares, diamonds, stars and triangles are extracted from the image. In the figure on the left, no spatial pyramids are used. In the figure on the right, the image is subdivided into three sub-regions. The FVs extracted separately from these sub-regions encode the spatial structure of the image. The final FV is the concatenation of the FVs extracted from all these sub-regions + the FV extracted over the whole image.*

In this thesis the general practice is dividing the image in three horizontal sub regions and concatenate the FV obtained in all the three horizontal stripes as well as the FV obtained using the whole image ($S = 4$).

2.1.4 Compression

High dimensional image descriptors such as the FV lead to good results in combination with linear classifiers. However, a weakness of high-dimensional image descriptors is that they have large memory footprints Sánchez and Perronnin (2011); Lin et al. (2011)[2]. Consequently, they are difficult to scale to large datasets. Several works have been proposed to use more compact descriptors for classification. One line of work consists in describing an image with a set of high-level concepts based on object classifiers Wang et al. (2009); Torresani et al. (2010); Bergamo et al. (2011) or object detectors Li et al. (2010). Coding techniques such as Product Quantization (PQ) Jégou et al. (2011) have also been used to compress the data Sánchez and Perronnin (2011); Rastegari et al. (2011); Vedaldi and Zisserman (2012).

Descriptor quantization is used to reduce the cardinality of the descriptor space. It maps a large set of samples to a smaller set so that the distance between the original and the quantized signal is minimized. In the source coding literature, this quantitative measure of quantization quality is referred to as quantization error. An example to quantization is the JPEG image format which is used to reduce the memory footprint of an image. Formally, a quantizer is a function that maps a D-dimensional vector $\mathbf{x} \in \mathbf{R}^D$ to a finite set.

Product Quantization (PQ) For large scale problems where the sample set size often scales to million of samples, using large codebooks is computationally intensive. At the same time, an approximate search brings high cost at both training and test time for high dimensional vectors. Product Quantization (PQ) technique of Jégou et al. (2011) aims at resolving this issue. The input vector $\mathbf{x}$ is split into m distinct sub-vectors $\mathbf{u}_j$ of dimension $D^* = D/m$ where $1 \leq j \leq m$. The sub vectors are quantized separately using m distinct sub-quantizers as follows (see Figure 2.8)

$$\underbrace{\mathbf{x}_1, ..., \mathbf{x}_{D^*}}_{\mathbf{u}_1}, ..., \underbrace{\mathbf{x}_{D-D^*+1}, ..., \mathbf{x}_D}_{\mathbf{u}_\mathrm{m}} \rightarrow q_1(\mathbf{u}_1), ..., q_m(\mathbf{u}_m) \tag{2.12}$$

where q_j is a sub-quantizer associated with the jth subvector. The index set I_j is association with the codebook C_j and the corresponding reproduction values $c_{i,j}$ is done through q_j. The reproduction value $c_{i,j}$ is identified by an element of the product index set ($I = I_1 \times ... \times I_m$) where the codebook is defined as the Cartesian product $C = C_1 \times ... \times C_m$. A centroid of the index set is the concatenation of centroids of the m subquantizers.

PQ is an efficient method to increase the codebook size at an affordable cost. It splits the input vector into a set of m sub-vectors. In training, each sub-vector is clustered and

[2]As an example, the PASCAL VOC 2009 winners mention in Gong et al. (2009) (slide 27) that their GMM-based Zhou et al. (2010) and sparse-coding-based Yang et al. (2010) representations have respectively 655K and 2M dimensions.

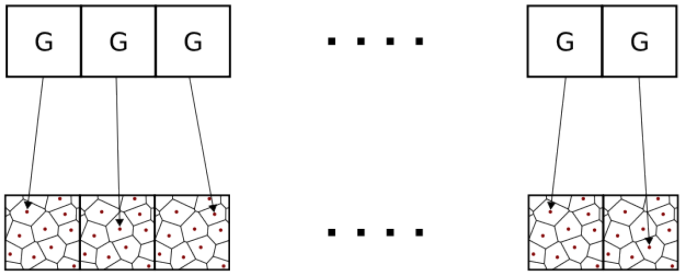


Figure 2.8 *Product Quantization: Firstly the image descriptor is split into small sub-vectors. Training is done by clustering each sub-vector and the compression is done by encoding each sub-vector to its closest codebook index (Image courtesy of Jorge Sánchez).*

quantized separately using m sub-quantizers. If we denote b as the average number of bits per dimension (assuming that the bits are equally distributed across the codebooks), the cost of learning and storing the codebook are around $O(D2^{bD^*})$ Jégou et al. (2012). It is also worth to note that $D^* = 1$ corresponds to no-compression and $D^* = D$ corresponds to Vector Quantization (VQ). PQ has been previously used with the FV in large-scale image classification Sánchez and Perronnin (2011); Sánchez et al. (2011). In our setting, the plain FVs are encoded with 32 bits per dimension (4-byte floating point arithmetic) whereas the compressed FVs are encoded with 1 bit per dimension.

2.2 Learning

Once the image descriptors are extracted, we need to train the classifiers to learn the parameters of the decision functions that separate different classes. Supervised image classification uses a set of image descriptors with their corresponding labels to build such decision functions. The aim is to minimize the empirical risk with regularization. The empirical risk corresponds to the cumulative misclassification loss for a set of samples. In testing, the decision function that is learned in training is used to assign one or multiple labels to an unseen image using a similarity score. In Section 2.2.1 we review the related work on supervised image classification in the context of large-scale learning.

In this thesis, we also consider the opposite problem of learning the decision boundaries with the lack of labeled training images which is known as zero-shot learning. In this case, side information is used to learn an embedding space that connects the images and the classes which can be used as a means of predicting the label of an unknown image. In another scenario, we explore the learning problem when only few training images are available. This is also known as few-shots learning. In Section 2.2.2 we discuss the related work on zero-shot and few-shots learning.

2.2.1 Large-Scale Learning

Along with the dimensionality of the image descriptors, the number of images and classes contained in the dataset are two of the properties that define the scale of the learning problem. Building such large datasets are mainly done in two steps. In the first step one simply queries image search engines such as Google Images[3] or Flickr[4] with a text query. In the second step one needs to post process the retrieved results manually to avoid the noisy results which is a costly process. Due to the high cost of manual labelling, the computer vision community used to focus on small-scale datasets such as Pascal VOC Everingham et al. (2010) that has 20 classes and 10,103 images, Caltech-101 Fei-Fei et al. (2006) that has 101 classes with 40 to 800 images per class, Caltech-256 Griffin et al. (2007) that has 256 classes and 30,607 images *etc*. However, with the emergence of crowdsourcing platforms such as the Amazon Mechanical Turk (AMT)[5], the image labeling process was simplified. This led to the development of truly large-scale datasets such as LabelMe Russell et al. (2008) and ImageNet Deng et al. (2009). Another large scale dataset, the Tiny-images Torralba et al. (2008) avoids the post processing step and keeps the noisy image labels. With the help of these recent large scale image datasets, the large-scale image classification problem gained popularity. In the following, we review the previous work on large-scale learning from the algorithms and solvers point of view. A detailed discussion about these algorithms and solvers is given in Chapter 3.

Supervised Large-Scale Learning Algorithms. In most previous works tackling the large-scale visual data sets, the objective function which is optimized is always the same: one binary SVM is learned per class in a one-vs-rest fashion Deng et al. (2010); Sánchez and Perronnin (2011); Lin et al. (2011); Rohrbach et al. (2011). Some approaches use simple classifiers such as the nearest-neighbor (NN) Torralba et al. (2008). While the exact NN can provide a competitive accuracy when compared to SVMs Weston et al. (2010); Deng et al. (2010), it is not straightforward to scale to large data sets. On the other hand, the Approximate Nearest Neighbor (ANN) performs poorly on high-dimensional image descriptors (significantly worse than one-vs-rest SVMs) while still being much more computationally intensive Weston et al. (2010). The one-vs-rest (OVR) strategies offer several advantages such as the speed and simplicity as compared to multi-class classifiers (see Rifkin and Klautau (2004) as a defense of OVR). Indeed, OVR classifiers are trained efficiently by decomposing the problem into independent per class problems.

On the other hand, several versions of multi-class SVMs were proposed. Weston and Watkins (1999) propose a multi-class SVM with a loss function that consists in summing the losses incurred by each class-wise score. Note that this version of multi-class SVM is statistically not consistent for large samples Tewari and Bartlett (2007). Lee et al. (2004)

[3] http://images.google.com/
[4] http://www.flickr.com/
[5] https://www.mturk.com/mturk/welcome

introduced a statistically consistent version of multi-class SVM, which can be thought of a "hinge-loss" counterpart of multinomial classification. In this work, the multi-class SVM formulation of Crammer and Singer (2002) is used, which is a computationally attractive variant which was proved to be consistent for large-scale problems.

Many alternative approaches for multi-class classification were proposed in the literature. Dietterich and Bakiri (1995) introduce error correcting codes as a basis of multi-class classification. Allwein et al. (2000) combine several multi-class classifiers using AdaBoost. Vural and Dy (2004) approach the problem from the decision trees perspective by partitioning the space in $N - 1$ regions. Platt (1999) uses sequential minimal optimization to speed up this process to polynomial time.

When the target loss function is not the classification accuracy but a more sophisticated performance measure such as mean-average-precision, a natural approach is to build ranking algorithms. Joachims (2002) proposed a ranking SVM, allowing to rank highly related documents on the higher ranks of the list. Grangier et al. (2006) improve the baseline ranking SVM by giving weights to classifiers. Usunier et al. (2009) penalize the loss encountered at the top of the list more than the bottom. Another ranking framework by Weston et al. (2010) uses the linear classifiers trained with SGD and a novel sampling trick to approximate the ranks.

In order to break the time-complexity of training to sub-linear in the number of classes, various approaches have been proposed which employ tree structures Marszalek and Schmid (2008); Bengio et al. (2010); Gao and Koller (2011); Chan and Stolfo (1996); Mehta et al. (1996); Gehrke et al. (2000). Beygelzimer et al. (2005) create a model for error-limiting reduction in the learning. According to this model, the tree (multi-class) reduction has a slightly larger loss rate than binary classifiers. The reader may refer to Rokach and Maimon (2005) for an extended survey of top down approaches for classification.

Large Scale Solvers. Here, we review the current publicly available solvers for the linear SVMs. Other algorithms will not be visited since in this thesis, they will not be experimentally evaluated. There are two main families of algorithms for optimizing the SVM objectives: batch algorithms, and online algorithms.

The state-of-the-art batch optimization algorithms for non-linear SVMs are based on a variant of coordinate-descent called sequential minimal optimization (SMO) Platt (1999). The main strength of batch approaches is the high robustness to the parameter setting, that are initialization, line-search, number of iterations, etc. Such algorithms are implemented in the most popular toolboxes LibSVM Chang and Lin (2011), SVM^{light} Joachims (1999), and Shogun Franc and Sonnenburg (2008). The state-of-the-art batch algorithms for linear SVMs are based on coordinate-descent approaches with the second-order acceleration. The widely used LibLinear toolbox Fan et al. (2008) provides an efficient implementation

of such algorithms. The weakness of the batch methods is the difficulty to scale up to large datasets.

The state-of-the-art optimization algorithms for linear SVMs are based on stochastic gradient descent (SGD) where the classifier parameters are determined by considering one sample at a time. Since the decision to determine the classification boundaries is made based on a single sample, the learning process is fast. The well known online SVM solver Pegasos Shalev-Shwartz et al. (2007) and libSGD Bottou (2003) provide an implementation of these algorithms. The main strength of SGD algorithms is their built-in ability to scale up to large datasets. However, without a careful setting of the parameters, SGD algorithms can have slow convergence and struggle to match the performance of their batch counterparts.

2.2.2 Learning with Scarce Training Data

One of the challenges of working with large-scale datasets containing thousand of classes is that labeled training data is hard to obtain. Especially for the classes which can be distinguished only by experts, the annotation is not accessible through crowdsourcing platforms such as AMT. Therefore, some of the classes remain unlabeled. Learning with the scarce training data, *e.g.* zero-shot learning and few-shots learning (see Figure 2.9) is a problem that we address in this thesis. In zero-shot learning, the training and the test classes are disjoint. In few-shots learning, the aim is to learn the classification boundaries in the presence of few training data. Following, we will review the related work that investigate the learning problem with the lack of training data.

Zero-Shot Learning. Zero-shot learning requires the ability to transfer knowledge from classes for which there is training data available to the classes for which there is no labeled training data available. There are two crucial choices when performing zero-shot learning: the choice of the prior information and the choice of the recognition model. The possible sources of prior information include attributes Lampert et al. (2009); Farhadi et al. (2009); Palatucci et al. (2009); Rohrbach et al. (2010a, 2011), semantic class taxonomies Rohrbach et al. (2011); Mensink et al. (2012a), class-to-class similarities Rohrbach et al. (2010a); Yu et al. (2013) or text features Palatucci et al. (2009); Rohrbach et al. (2010a, 2011); Socher et al. (2013). Rohrbach et al. (2011) compare different sources of information for learning with zero or few samples. However, since different models are used for the different sources of prior information, it is unclear whether the observed differences are due to the prior information itself or the model. In Chapter 4, the attributes and the class hierarchies are compared using the exact same learning framework. Therefore, we provide a fair comparison of different sources of side information. In addition to attributes and hierarchies, other sources of prior information have been proposed for special purpose problems. For instance, Larochelle et al. (2008) encode the characters with 7×5

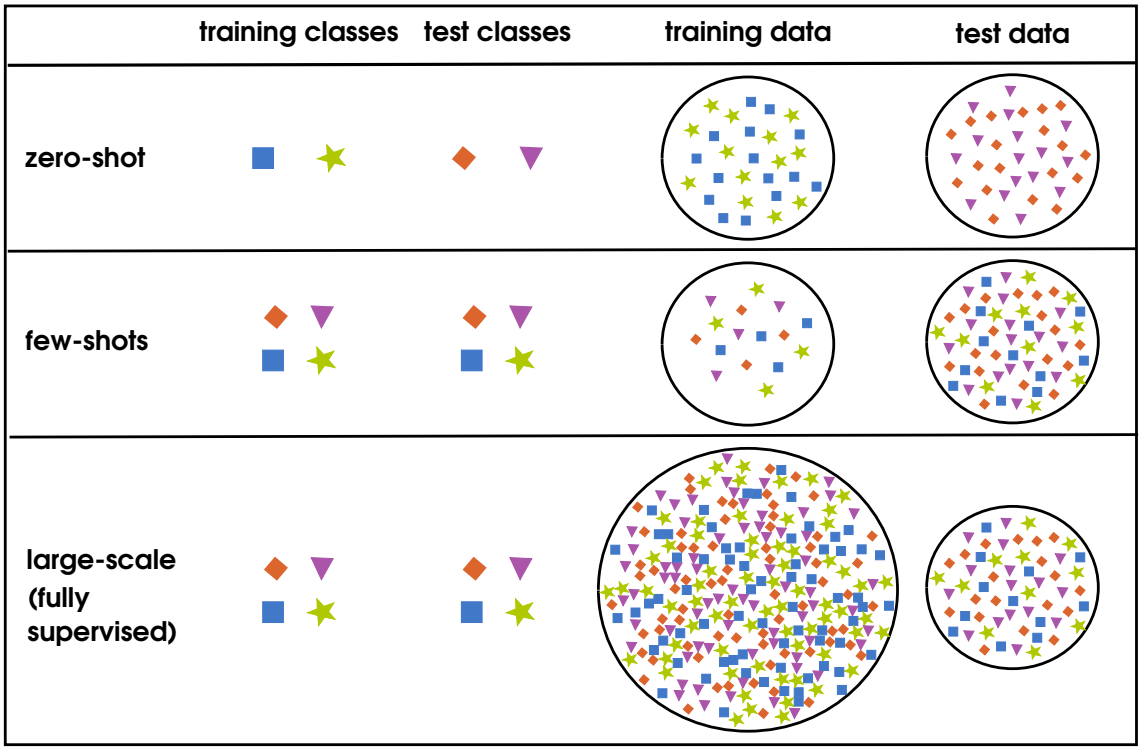


Figure 2.9 *In the figure we use four object classes: square, star, diamond and triangle. In the training and test sets, each symbol represents an image that belongs to one of those classes. In zero-shot learning, the training and the test classes are disjoint, accordingly, the training dataset does not contain any image from one of the classes that is included in the test dataset. In few-shots learning, the aim is to learn the classification boundaries with few training data. The training and the test classes are the same (as in large scale setting). In few-shots learning, the training dataset contains very few samples whereas in large scale learning (fully supervised setting) we have plenty of positive and negative samples from each class.*

pixel representations. It is unclear, however, how such an embedding can be extrapolated to the case of generic visual categories.

As for the recognition model, there are several alternatives. The state of the art DAP algorithm Lampert et al. (2009) uses a probabilistic model which assumes attribute independence. The ALE method described in Chapter 4 has been inspired by those works which perform zero-shot recognition by assigning an image to its closest class embedding. The distance between an image and a class embedding is generally measured as the Euclidean distance and a transformation is learned to map the input image features into the class embeddings Palatucci et al. (2009); Socher et al. (2013). ALE learns the input-to-output mapping features to optimize directly an image classification criterion: learning to rank the correct label higher than incorrect ones. In this thesis, it has been shown that this leads

to improved results compared to those works which optimize a regression criterion such as Palatucci et al. (2009); Socher et al. (2013).

Few-Shots Learning Few works have considered the problem of transitioning from zero-shot learning to learning with few shots Yu and Aloimonos (2010); Sharmanska et al. (2012); Yu et al. (2013). As mentioned earlier, Yu and Aloimonos (2010) is only applicable to the bag-of-words type of models. Sharmanska et al. (2012) proposes to augment the attribute-based representation with additional dimensions for which an autoencoder model is coupled with a large margin principle. While this extends DAP to learning with labeled data, this approach does not improve DAP for zero-shot recognition. In this thesis, we show that the ALE framework can admit transition from zero-shot to few-shots learning and improves the DAP in the zero-shot regime. As an alternative, Yu et al. (2013) learns separately the class embeddings and the input-to-output mapping which is suboptimal. On the contrary, when learning from few-shots, we learn the class embeddings and the input-to-output mappings jointly (using attributes as prior) to optimize classification accuracy.

3

Good Practice in Large Scale Learning

Contents

3.1 Introduction **34**

3.2 Related Work **36**

3.3 Objective Functions **37**

3.4 Optimization **40**

3.5 Experiments **43**

3.5.1 Fine Grained Experiments 45

3.5.2 Large-Scale Experiments 57

3.6 Conclusion **63**

In this chapter we benchmark several SVM objective functions for large-scale image classification. We consider the One-vs-Rest (OVR), Multiclass (MUL), Ranking (RNK) and Weighted Approximate Ranking (WAR) objective functions. A comparison of online and batch methods for optimizing these objectives shows that online methods perform as well as batch methods in terms of classification accuracy, with a significant gain in training speed. Using stochastic gradient descent, we can scale the training to millions of images and thousands of classes. Our experimental evaluation shows that ranking-based algorithms do not outperform the one-vs-rest strategy when a large number of training examples are used. Furthermore, the gap in accuracy between the different algorithms shrinks as the dimension of the features increases. Moreover, learning through cross-validation the optimal rebalancing of positive and negative examples can result in a significant improvement for the one-vs-rest strategy. Finally, early stopping can be used as an effective regularization strategy when training with online algorithms. Following these "good practices" we improve the state-of-the-art top-1 image classification accuracy on a large subset of 10K classes and 9M images of ImageNet from 16.7% to 19.1%.

3.1 Introduction

Image classification is the problem of assigning one or multiple labels to an image based on its content. This is a standard supervised learning problem: given a training set of labeled images, the goal is to learn classifiers to predict labels of new images.

Large-scale image classification has recently received significant interest from the computer vision and machine learning communities Torralba et al. (2008); Deng et al. (2010); Weston et al. (2010); Bengio et al. (2010); Sánchez and Perronnin (2011); Lin et al. (2011); Rohrbach et al. (2011); Deng et al. (2011); Zhao et al. (2011). This goes in hand with large-scale datasets being available. For instance, the ImageNet[1] dataset consists of more than 14M images labeled with almost 22K concepts Deng et al. (2009), the Tiny images dataset consists of 80M images corresponding to 75K concepts Torralba et al. (2008) and Flickr contains thousands of groups[2] with thousands (and sometimes hundreds of thousands) of pictures, which can be exploited to learn object classifiers Wang et al. (2009); Perronnin et al. (2010b).

Standard large-scale image classification pipelines use high-dimensional image descriptors in combination with linear classifiers Lin et al. (2011); Sánchez and Perronnin (2011). The use of linear classifiers is motivated by their computational efficiency which is a requirement when dealing with a large number of classes and images. High-dimensional descriptors allow separating the data with a linear classifier, i.e., they perform the feature mapping explicitly and avoid using non-linear kernels. As a rule of thumb, linear classifiers with high dimensional descriptors perform similarly to low dimensional bag-of-visual-words (BOV) with non-linear classifiers Chatfield et al. (2011). In the literature several high dimensional descriptors exist, i.e., the Fisher Vector (FV) Perronnin and Dance (2007); Perronnin et al. (2010b), local coordinate coding Wang et al. (2010) and supervector coding Zhou et al. (2010). One of the simplest strategies to learn classifiers in the multiclass setting is to train one-vs-rest binary classifiers independently for each class. Most image classification approaches have adopted this strategy not only because of its simplicity but also because it can easily be parallelized on multiple cores or machines. As an example, the two top systems at the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) 2010 Berg et al. (2010a) used such an approach Lin et al. (2011); Sánchez and Perronnin (2011).

Another approach is to view image classification as a ranking problem: given an image, the goal is to rank the labels according to their relevance. Performance measures such as the top-k accuracy reflect this goal and are used to report results on standard benchmarks, such as ILSVRC. While the one-vs-rest strategy is computationally efficient and yields competitive results in practice, it is – at least in theory – clearly suboptimal with respect to a strategy directly optimizing a ranking loss Bordes et al. (2007); Crammer and Singer

[1] http://www.image-net.org
[2] http://www.flickr.com/groups

RECOMMENDATIONS FOR LARGE-SCALE IMAGE CLASSIFICATION

1. *Stochastic training*: learning with the stochastic gradient descent (SGD) is well-suited for large-scale datasets
2. *Class imbalance*: optimizing the imbalance parameter in the one-vs-rest strategy is a must for competitive performance
3. *Early stopping*: regularizing through early stopping results in fast training and a good generalization performance
4. *Step-size*: a small-enough step-size w.r.t. the learning rate is often sufficient for state-of-the-art performance
5. *One-vs-rest*: the one-vs-rest strategy is a flexible option for large-scale image classification
6. *Capacity saturation*: for a sufficiently large image representation, all strategies lead to similar performance

(2002); Joachims (2002); Usunier et al. (2009); Yue et al. (2007); Xu et al. (2008); Weston et al. (2010).

In this chapter, we examine these ranking approaches to determine whether they scale well to large datasets and whether they improve the performance. We compare the one-vs-rest binary SVM, the multi-class SVM of Crammer and Singer (2002) that optimizes top-1 accuracy, the ranking SVM of Joachims (2002) that optimizes the rank as well as the recent weighted approximate ranking of Weston et al. (2010) that optimizes the top of the ranking list. The datasets we consider are large-scale in the number of classes (up to 10K), images (up to 9M) and feature dimensions (up to 130K).

For efficiency reasons we train our linear classifiers using Stochastic Gradient Descent (SGD) algorithms LeCun et al. (1998) with the primal formulation of the objective functions as in Bottou and Bousquet (2007); Shalev-Shwartz et al. (2007) for binary SVMs or in Nowozin and Lampert (2011) for structured SVMs. By using the exact same optimization framework, we truly focus on the merits of the different objective functions, not on the merits of the particular optimization techniques.

Our experimental evaluation confirms that SGD-based learning algorithms can work as well as batch techniques at a fraction of their cost. It also shows that ranking objective functions seldom outperform the one-vs-rest strategy. Only when a small amount of training data is available, did we observe a small-but-consistent improvement with ranking based methods. The gap between the accuracy of different learning algorithms reduces in case of high-dimensional data. We also experimentally show that for the one-vs-rest strategy carefully tuning the optimal degree of imbalance between the positive and the negative examples can have a significant impact on accuracy. This is observed especially when the feature dimensionality is "small" with respect to the problem complexity, in

particular with respect to the number of classes. Furthermore, early stopping can be used as an effective regularization strategy for fast training with SGD. Following these "good practices", we were able to improve the state-of-the-art accuracy on a large subset of 10K classes and 9M of images of ImageNet Deng et al. (2009) from 16.7% to 19.1%. We summarize our findings in the "recommendation box" at the top of the previous page.

Section 3.2 reviews the previous work that this chapter is based on. Section 3.3 presents the different objective functions for linear classification. Section 3.4 describes the SGD-based optimization framework. Experimental results are presented in Section 3.5 for five state-of-the-art datasets: three fine grained subsets of ImageNet (*i.e.* Fungus134, Ungulate183 and Vehicles262) as well as the ILSVRC 2010 and the very large-scale ImageNet10K dataset. Section 3.6 provides the conclusions and the future work. This chapter corresponds to the published works Perronnin et al. (2012) and Akata et al. (2013a).

3.2 Related Work

The related work on large-scale learning has been presented in Section 2.2.1. In this section we briefly review the related work that is the most relevant to this chapter.

Most of today's approaches for image classification first extract visual image descriptors and then apply a classifier. In this work, we use the state-of-the-art image descriptors, namely FV Perronnin and Dance (2007), which has been shown to yield excellent results in large-scale classification while requiring reasonable computational resources Sánchez and Perronnin (2011); Chatfield et al. (2011). We use Product Quantization(PQ) Jégou et al. (2011) for compression. For training, we use different formulations of linear SVMs.

In the literature, the previous works tackling large-scale visual data sets use the same objective function: one binary SVM is learned per class in a one-vs-rest fashion Deng et al. (2010); Sánchez and Perronnin (2011); Lin et al. (2011); Rohrbach et al. (2011). In this chapter, we benchmark different objective functions for learning linear SVM parameters. We compare the One-vs-Rest(OVR) with the Multiclass(MUL) Crammer and Singer (2002), the Ranking(RNK) Joachims (2002) and the Weighted Average Ranking(WAR) Grangier et al. (2006); Usunier et al. (2009).

The above mentioned objective functions are optimized using Stochastic Gradient Descent (SGD) where the classifier parameters are determined by considering one sample at a time. Since the the decision to determine the classification boundaries is made based on a single sample, the learning process is fast. The experimental evaluation of this chapter also compares SGD and batch solvers such as LibSVM Chang and Lin (2011), SVM^{light} Joachims (1999) and LibLinear Fan et al. (2008). The main strength of SGD algorithms is their built-in ability to scale up to large datasets.

3.3 Objective Functions

In this section we will provide the formulations for the One-vs-Rest (OVR), Multiclass (MUL), Ranking (RNK), Weighted Approximate Ranking(WAR) objective functions.

Let $S = \{(\mathbf{x}_i, y_i), i = 1 \dots N\}$ be the training set where $\mathbf{x}_i \in \mathcal{X}$ is an image descriptor where $\mathcal{X} = \mathbb{R}^D$, $y_i \in \mathcal{Y}$ is the associated label and $\mathcal{Y}$ is the set of possible labels. Let $C = |\mathcal{Y}|$ denote the number of classes. We shall always take $\mathcal{X} = \mathbb{R}^D$.

The supervised learning corresponds to minimizing the empirical risk with a regularization penalty that is formulated as follows:

$$\underset{\mathbf{W}}{\text{Minimize}} \ \tfrac{\lambda}{2}\Omega(\mathbf{W}) + L(S; \mathbf{W}) \ , \tag{3.1}$$

where $\mathbf{W}$ is the weight matrix stacking the weight vectors corresponding to each subproblem. The objective decomposes into the empirical risk that is defined as:

$$L(S; \mathbf{W}) := \frac{1}{N} \sum_{i=1}^{N} L(\mathbf{x}_i, y_i; \mathbf{W}) \tag{3.2}$$

with $L(\mathbf{x}_i, y_i; \mathbf{W})$ a surrogate loss of the labeled example (x_i, y_i), and the regularization penalty that is defined as:

$$\Omega(\mathbf{W}) := \sum_{c=1}^{C} \|\mathbf{w}_c\|^2 \tag{3.3}$$

The parameter $\lambda \geq 0$ controls the trade-off between the empirical risk and the regularization. In the following, we first briefly review the classical binary SVM, then proceed with the multiclass, ranking and weighted approximate ranking SVMs. We finally discuss the issue of data re-weighting.

1) Binary One-vs-Rest SVM (OVR)

In the case of the one-vs-rest SVM, we assume that we have only two classes and $\mathcal{Y} = \{-1, +1\}$. Let $\mathbb{1}(u) = 1$ if u is true and 0 otherwise. The zero-one (0/1) loss $\mathbb{1}(y_i \mathbf{w}^T x_i < 0)$ is upper-bounded by:

$$L_{\text{OVR}}(\mathbf{x}_i, y_i; \mathbf{w}) = \max\{0, 1 - y_i \mathbf{w}^T \mathbf{x}_i\} \tag{3.4}$$

If there are more than two classes, then one transforms the C-class problem into C binary problems and trains independently C one-vs-rest classifiers. Since the classifiers are trained independently, this procedure can be easily parallelized.

2) Beyond Binary Classification

In the following sections, we treat the classes jointly and the label set contains more than two labels *i.e.* $\mathcal{Y} = \{1, \ldots, C\}$. Let $\{\mathbf{w}_c, c = 1 \ldots C\}$ denote the C classifiers corresponding to each of the C classes. In this case, $\mathbf{W}$ is a $C \times D$ dimensional vector obtained by concatenating the different w_c's. The loss incurred for assigning label $\bar{y}$ while the correct label was y is denoted by $\Delta(y, \bar{y})$. In this thesis, we focus on the 0/1 loss, *i.e.* $\Delta(y, \bar{y}) = 0$ if $y = \bar{y}$ and 1 otherwise. In the following, we assume to have one label per image to simplify the representation.

Multiclass SVM (MUL). There exist several flavors of the multiclass SVM such as the Weston and Watkins (1999) and the Crammer and Singer (2002) formulations (see Tewari and Bartlett (2007) for a comprehensive review). Both variants propose a convex surrogate loss to $\Delta(y_i, \hat{y}_i)$ with:

$$\hat{y}_i = \arg\max_y \mathbf{w}_y^T x_i, \tag{3.5}$$

i.e. the loss incurred by taking the highest score as the predicted label. The Weston and Watkins (1999) formulation uses:

$$L_{\text{WW}}(\mathbf{x}_i, y_i; \mathbf{w}) = \sum_{y \neq y_i} [\Delta(y_i, y) - \mathbf{w}_{y_i}^T \mathbf{x}_i + \mathbf{w}_y^T \mathbf{x}_i]_+ \tag{3.6}$$

This multiclass SVM formulation consists in summing the losses incurred by each class-wise score. In this thesis, we use the multiclass SVM formulation of Crammer and Singer (2002), which provides a tighter upper bound on the ideal misclassification loss and was proved to be consistent for large-scale problems. This formulation corresponds to:

$$L_{\text{MUL}}(\mathbf{x}_i, y_i; \mathbf{w}) = \max_y \left\{\Delta(y_i, y) + \mathbf{w}_y^T x_i\right\} - \mathbf{w}_{y_i}^T x_i. \tag{3.7}$$

Note that this is a particular case of the structured SVM introduced by Tsochantaridis et al. (2005).

Ranking SVM (RNK). As an alternative to MUL, Joachims (2002) considers the problem of ordering the pairs of documents. Adapting the ranking framework to our problem the goal is, given a sample $(\mathbf{x}_i, y_i)$ and a label $y \neq y_i$, to enforce $\mathbf{w}_{y_i}\mathbf{x}_i > \mathbf{w}_y^T \mathbf{x}_i$. The rank of label y for sample $\mathbf{x}$ can be written as:

$$r(\mathbf{x}, y) = \sum_{c=1}^{C} \mathbb{1}(\mathbf{w}_c^T \mathbf{x} \geq \mathbf{w}_y^T \mathbf{x}) \tag{3.8}$$

Given the triplet $(\mathbf{x}_i, y_i, y)$, $\mathbb{1}(\mathbf{w}_{y_i}^T\mathbf{x} \geq \mathbf{w}_y^T\mathbf{x})$ is upper-bounded by:

$$L_{\text{tri}}(\mathbf{x}_i, y_i, y; \mathbf{w}) = \max\{0, \Delta(y_i, y) - \mathbf{w}_{y_i}^T\mathbf{x}_i + \mathbf{w}_y^T\mathbf{x}_i\} \tag{3.9}$$

Therefore, the overall loss of $(\mathbf{x}_i, y_i)$ writes as:

$$L_{\text{RNK}}(\mathbf{x}_i, y_i; \mathbf{w}) = \sum_{y=1}^{C} \max\{0, \Delta(y_i, y) - (\mathbf{w}_{y_i} - \mathbf{w}_y)^T\mathbf{x}_i\} \tag{3.10}$$

Weighted Approximate Ranking SVM (WAR). An issue with RNK is that the misclassification loss is the same when going from rank 99 to 100 or from rank 1 to rank 2. However, in most practical applications, one is interested in the top of the ranked list. As an example, in ILSVRC the measure used during the competition is a loss at rank 5. Usunier et al. (2009) therefore proposed to minimize a function of the rank which gives more weight to the top of the list.

Let $\alpha_1 \geq \alpha_2 \geq \ldots \alpha_C \geq 0$ be a set of C coefficients. For sample $(\mathbf{x}_i, y_i)$ the loss is $\ell_{r(\mathbf{x}_i, y_i)}$ with ℓ defined as:

$$\ell_k = \sum_{j=1}^{k} \alpha_j \tag{3.11}$$

where the penalty incurred by going from rank k to $k+1$ is α_k. Hence, a decreasing sequence $\{\alpha_j\}_{j\geq 1}$ implies that a mistake on the rank when the true rank is at the top of the list incurs a higher loss than a mistake on the rank when the true rank is lower in the list. The objective function that is based on an ordered weighted averaging scheme is generic and admits as special cases the multiclass SVM of Crammer and Singer ($\alpha_1 = 1$ and $\alpha_j = 0$ for $j \geq 2$) and the ordered pairwise ranking SVM of Joachims ($\alpha_i = 1$, $\forall i$).

While Usunier et al. (2009) proposes an upper-bound on the loss, Weston et al. (2010) propose an approximation which is more amenable to large-scale optimization. We follow Weston et al. (2010) and write:

$$L_{\text{WAR}}(\mathbf{x}_i, y_i; \mathbf{w}) = \sum_{y=1}^{C} \ell_{r_\Delta(\mathbf{x}_i, y_i)} \frac{L_{\text{tri}}(\mathbf{x}_i, y_i, y; \mathbf{w})}{r_\Delta(\mathbf{x}_i, y_i)}, \tag{3.12}$$

where

$$r_\Delta(\mathbf{x}, y) = \sum_{c=1}^{C} \mathbb{1}(\mathbf{w}_c^T x + \Delta(y, c) \geq \mathbf{w}_y^T x). \tag{3.13}$$

is a regularized rank. Following Weston et al. (2010), we choose $\alpha_j = 1/j$. As opposed to other works Xu et al. (2008); Yue et al. (2007), this does not optimize directly standard

information retrieval measures such as Average Precision (AP). However, it mimics their behavior by putting an emphasis on the top of the list, works well in practice and is highly scalable.

Rebalancing the Positive and the Negative Samples

When the training set is unbalanced, *i.e.* when some classes are significantly more populated than others, it can be beneficial to reweight the data. For large scale datasets, this unbalance is extreme in the one-vs-rest case. In such a situation when one has to deal with a large number of classes C, the unbalance between the positive class and the negative class is on average $C-1$. In the binary case, the reweighting can be performed by introducing a parameter ρ and the empirical risk then writes as:

$$\frac{\rho}{N_+}\sum_{i\in I_+} L_{\text{OVR}}(\mathbf{x}_i, y_i; \mathbf{w}) + \frac{1-\rho}{N_-}\sum_{i\in I_-} L_{\text{OVR}}(\mathbf{x}_i, y_i; \mathbf{w}) \tag{3.14}$$

where I_+ (resp. I_-) is the set of indices of the positive (resp. negative) samples and N_+ (resp. N_-) is the cardinality of this set. Note that $\rho = 1/2$ corresponds to the natural rebalancing of the data, *i.e.* in such a case one gives as much weight to positives and negatives.

While training all classes simultaneously, to introduce one parameter per class is computationally intractable. It would require to cross-validate C parameters jointly, including the regularization parameter. In such a case, the natural re-balancing appears to be the most natural choice. In the multiclass case, the empirical loss becomes:

$$\frac{1}{C}\sum_{c=1}^{C}\frac{1}{N_c}\sum_{i\in I_c} L_{\text{MUL}}(\mathbf{x}_i, y_i; \mathbf{w}) \tag{3.15}$$

where $I_c = \{i : y_i = c\}$ and N_c is the cardinality of this set. One can perform a similar rebalancing in the case of L_{RNK} and L_{WAR}.

3.4 Optimization

We employ the Stochastic Gradient Descent (SGD) to learn linear classifiers in the primal [3] Shalev-Shwartz et al. (2007); Bottou and Bousquet (2007). Such an approach has recently gained popularity in the computer vision community for large-scale learning as

[3] Note that SGD algorithms have also been proposed for non-linear kernels which perform the optimization in the dual, such as LaRank for multiclass SVMs Bordes et al. (2007).

	Sampling	Update
R_{OVR}	Draw $(\mathbf{x}_i, y_i)$ from S.	$\delta_i = 1$ if $L_{\text{OVR}}(\mathbf{x}_i, y_i; \mathbf{w}) > 0, 0$ otherwise. $\mathbf{w}^{(t)} = (1 - \eta_t \lambda)\mathbf{w}^{(t-1)} + \eta_t \delta_i \mathbf{x}_i y_i$
R_{MUL}	Draw $(\mathbf{x}_i, y_i)$ from S.	$\bar{y} = \arg\max_y \Delta(y_i, y) + \mathbf{w}'_y \mathbf{x}_i$ and $\delta_i = \begin{cases} 1 & \text{if } \bar{y} \neq y_i \\ 0 & \text{otherwise.} \end{cases}$ $\mathbf{w}_y^{(t)} = \begin{cases} \mathbf{w}_y^{(t-1)}(1 - \eta_t \lambda) + \delta_i \eta_t \mathbf{x}_i & \text{if } y = y_i \\ \mathbf{w}_y^{(t-1)}(1 - \eta_t \lambda) - \delta_i \eta_t \mathbf{x}_i & \text{if } y = \bar{y} \\ \mathbf{w}_y^{(t-1)}(1 - \eta_t \lambda) & \text{otherwise.} \end{cases}$
R_{RNK}	Draw $(\mathbf{x}_i, y_i)$ from S. Draw $\bar{y} \neq y_i$ from $\mathcal{Y}$.	$\delta_i = 1$ if $L_{\text{tri}}(\mathbf{x}_i, y_i, \bar{y}; \mathbf{w}) > 0, 0$ otherwise. $\mathbf{w}_y^{(t)} = \begin{cases} \mathbf{w}_y^{(t-1)}(1 - \eta_t \lambda) + \delta_i \eta_t \mathbf{x}_i & \text{if } y = y_i \\ \mathbf{w}_y^{(t-1)}(1 - \eta_t \lambda) - \delta_i \eta_t \mathbf{x}_i & \text{if } y = \bar{y} \\ \mathbf{w}_y^{(t-1)}(1 - \eta_t \lambda) & \text{otherwise.} \end{cases}$
R_{WAR}	Draw $(\mathbf{x}_i, y_i)$ from S. For $k = 1, 2, \ldots, C-1$, do: $\begin{cases} \text{Draw } \bar{y} \neq y_i \text{ from } \mathcal{Y}. \\ \text{If } L_{\text{tri}}(\mathbf{x}_i, y_i, \bar{y}; \mathbf{w}) > 0, \text{ break.} \end{cases}$	$\delta_i = 1$ if $\bar{y}$ s.t. $L_{\text{tri}}(\mathbf{x}_i, y_i, \bar{y}; \mathbf{w}) > 0$ was sampled, 0 otherwise. $\mathbf{w}_y^{(t)} = \begin{cases} \mathbf{w}_y^{(t-1)}(1 - \eta_t \lambda) + \delta_i \ell_{\lfloor \frac{C-1}{k} \rfloor} \eta_t \mathbf{x}_i & \text{if } y = y_i \\ \mathbf{w}_y^{(t-1)}(1 - \eta_t \lambda) - \delta_i \ell_{\lfloor \frac{C-1}{k} \rfloor} \eta_t \mathbf{x}_i & \text{if } y = \bar{y} \\ \mathbf{w}_y^{(t-1)}(1 - \eta_t \lambda) & \text{otherwise.} \end{cases}$

Table 3.1 *The sampling for the OVR, MUL, RNK, WAR objective functions and the SGD based update equations for each objective function.*

shown by Perronnin et al. (2010a,b); Weston et al. (2010); Sánchez and Perronnin (2011); Lin et al. (2011); Rohrbach et al. (2011). In the following, we describe the training based on the SGD and our implementation details.

1) Stochastic Training

Training with stochastic gradient descent (SGD) consists in choosing a sample at random at each step and updating the parameters $\mathbf{w}$ using a sample-wise estimate of the regularized risk. In the case of R_{OVR} and R_{MUL}, the sample is simply a pair $(\mathbf{x}_i, y_i)$ while in the case of R_{RNK} and R_{WAR} it consists of a triplet $(\mathbf{x}_i, y_i, \bar{y})$ where $\bar{y} \neq y_i$.

Let z_t denote the sample drawn at step t (whether it is a pair or a triplet) and let $R(z_t; \mathbf{w})$ be the sample-wise estimate of the regularized risk. With η_t being the step size, the parameter vector $\mathbf{w}$ is updated as follows:

$$\mathbf{w}^{(t)} = \mathbf{w}^{(t-1)} - \eta_t \nabla_{\mathbf{w}=\mathbf{w}^{(t-1)}} R(z_t; \mathbf{w}). \tag{3.16}$$

Assuming there is no data rebalancing, the sampling and update procedures for the objective functions used in the experimental evaluation can be found in Table 3.1. For R_{OVR}, R_{MUL} and R_{RNK}, these equations are straightforward and they optimize the exact regularized risk. For R_{WAR} it is only approximation as it does not compute the value of $r_\Delta(\mathbf{x}_i, y_i)$

exactly, but estimates it from the number of samples k which were drawn before a violating sample $\bar{y}$ was found such that $L_{\text{tri}}(x_i, y_i, \bar{y}; \mathbf{w}) > 0$. If k samples were drawn, then:

$$r_{\Delta}(\mathbf{x}_i, y_i) \approx \lfloor \frac{C-1}{k} \rfloor. \tag{3.17}$$

For a large number of classes, this approximate procedure is significantly faster than the exact one (see Weston et al. (2010) for more details). Note also that the regularization can be implemented by penalizing the squared norm of $\mathbf{w}$, while Weston et al. (2010) actually bound the norm of $\mathbf{w}$. We observed that both strategies provide similar results in practice.

2) Implementation Details

The basis of the implementation in this work is the SGD library for binary classification that is available on Bottou (1997)'s website (version 1.3). This is an optimized code which includes fast linear algebra and a number of optimizations such as using a scale variable to update $\mathbf{w}$ only when a loss is incurred. In the following some of the implementation details are presented.

- **Bias:** Until now, we have not considered the bias in our objective functions. The bias corresponds to an additional parameter per class. Following the common practice, we add one constant entry to each observation, *e.g.* the image descriptor $\mathbf{x}_i$. As is the case in Bottou (1997)'s code, we do not regularize this additional dimension.

- **Stopping Criterion:** Since at each step in SGD there is a noisy estimate of the objective function, this value cannot be used for stopping the iterations. Therefore, in all our experiments, we use a validation set and stop iterating when the accuracy does not increase by more than a threshold θ.

- **Regularization:** While a vast majority of works on large-margin classification regularize explicitly by penalizing the squared norm of w or by bounding it, an alternative is regularizing implicitly by early stopping Bai et al. (2009). In such a case, one sets $\lambda = 0$ and iterates until the performance converges (or starts descreasing) on a validation set. In our experiments, applying this strategy yields competitive results.

- **Step Size:** In order to guarantee converge to the optimum, the sequence of step sizes should satisfy $\sum_{t=1}^{\infty} \eta_t = \infty$ and $\sum_{t=1}^{\infty} \eta_t^2 < \infty$. Assuming $\lambda > 0$, the usual choice is $\eta_t = 1/\lambda(t + t_0)$, where t_0 is a parameter. Bottou (1997) provides a heuristic to set t_0 in his code. Another option is to cross validate the t_0 but the empirical observation showed no significant improvements. However, in the experimental section the reader can find various experiments reporting competitive results with decreasing and a fixed step size η as in Bai et al. (2009); Weston et al. (2010).

- **Rebalancing Positive and Negative Samples:** All sampling/update equations in Table 3.1 are based on non-rebalanced objective functions. To rebalance the data, either the sampling or the update equations should be modified. We chose the first alternative, since in general it led to faster convergence. If we take the example of L_{OVR}, then the sampling is modified as follows: Draw $y = +1$ with probability ρ; $y = -1$ with probability $1 - \rho$ and x_i such that $y_i = y$.

- **Parallelization:** Since learning the classifiers in parallel speeds up the computation, here the C code has been parallelized using pthreads. For one-vs-rest SVMs, this is trivially done by independently training the classifiers on different CPUs. For the multiclass SVM, the most expensive step is the $\arg\max$ operation of equation (3.5) which requires computing C scores and which is distributed across CPUs. For pairwise ranking, the CPUs sample different triplets and a mutex avoids two CPUs working simultaneously on the same classes. The efficiency is high when the probability of "collision" of two CPUs on the same class is low, *i.e.* when the number of classes is significantly larger than the number of processing units. For weighted approximate ranking, the CPUs compete to find a violating class. Significant speed-ups can be obtained when a large number of classes has to be sampled, *i.e.* during later iterations.

Comparison with Batch Solvers. It is well known that SGD performs as well as batch solvers for OVR SVMs at a fraction of the cost Bottou (2003); Bottou and Bousquet (2007). However, to the best of our knowledge, the public SGD solvers do not exist for other SVM formulations such as MUL SVM. As a sanity check, several batch solvers have been compared to the SGD solvers in Section 3.5. For instance, the MUL SGD solver has been compared to SVM^{light} Joachims (1999) multiclass and LibLinear Fan et al. (2008) batch solvers. Moreover, the OVR SGD solver has been compared to LibSVM Chang and Lin (2011) batch solver. Because of the cost of running batch solvers, we ran experiments on Fine Grained subsets of ImageNet on small BOV vectors and on synthetic datasets.

3.5 Experiments

In this section, we first describe the datasets and image descriptors used in our experimental setup. In Section 3.5.1, we provide a detailed analysis of the different objective functions and parameters for three fine-grained subsets of ImageNet (*i.e.* Fungus134, Ungulate183, Vehicle262). We believe that these fine-grained datasets can provide useful insights about the different objective functions since they correspond to different class densities as shown in Deng et al. (2010). In Section 3.5.2, we provide our results on two large-scale datasets, the ILSVRC10 and the ImageNet10K.

Datasets. For our fine-grained image categorization experiments, we use three subsets of ImageNet: Fungus134, Ungulate183 and Vehicle262 as described in Deng et al. (2010). These datasets contain only the leaf nodes under the respective parent class (*i.e.* fungus, ungulate or vehicle) in the hierarchy of ImageNet. Unless stated otherwise, we use the following set-up: half of the data for training, 5K images for validation and the remainder for testing.

For the large-scale experiments, we use the ILSVRC 2010 Berg et al. (2010a) and the ImageNet10K Deng et al. (2010) datasets. For the ILSVRC2010, we follow the standard training/validation/testing protocol. For the ImageNet10K, we follow Sánchez and Perronnin (2011) and use half of the data for training, 50K images for validation and the remainder for testing. The properties of the datasets are summarized in Table 3.2.

	Total # of		Partition		
	images	classes	train	val	test
Fungus134	88K	134	44K	5K	39K
Ungulate183	183K	183	91.5K	5K	86.5K
Vehicle262	226K	262	113K	5K	108K
ILSVRC10	1.4M	1,000	1.2M	50K	150K
ImageNet10K	9M	10,184	4.5M	50K	4.45M

Table 3.2 *The datasets used in the experimental evaluation. For the fine-grained image classification experiments we use the three standard subsets of the ImageNet (*i.e. *Fungus134, Ungulate183 and Vehicle262) as described in Deng et al. (2010). These datasets contain only the leaf nodes under the respective parent class in the hierarchy of ImageNet. For the large-scale experiments, we use the ILSVRC 2010 fromBerg et al. (2010a) and the ImageNet10K from Deng et al. (2010).*

In all the experiments, we compute the flat top-1 or top-5 accuracy per class and report the average as in Deng et al. (2010); Sánchez and Perronnin (2011); Weston et al. (2010). We could also have used a hierarchical loss which takes into account the fact that some errors are more costly than others. The choice of a flat *vs.* a hierarchical loss is directly related to the choice of the cost function Δ (see Section 3.3). In our preliminary experiments, this choice had only very limited impact on the ranking of the different objective functions. This effect was also observed during the ILSVRC 2010 and 2011 competitions which might explain why the hierarchical measure was dropped in 2012. In what follows we report the accuracy with only the flat loss.

Image Descriptors. Before the feature extraction, we resize the images to 100K pixels (if larger) while keeping the aspect ratio. We extract approximately 10K SIFT descriptors Lowe (2004) at 5 scales for 24×24 patches every 4 pixels. The SIFT descriptors are reduced from 128-dim to 64-dim using PCA. They are then aggregated into an image-level

signature (*e.g.* BOV or FV) using a probabilistic visual vocabulary, such as a Gaussian Mixture Model (GMM). By default, we use Fisher Vectors (FV) with N =256 Gaussians. We also report results with the bag of visual words (BOV) given its popularity in large-scale classification Deng et al. (2010); Weston et al. (2010). Our default BOV uses $N = 1,024$ codewords. Note that our purpose is not to compare the BOV and FV representations in this work (see Chatfield et al. (2011) for such a comparison). Unless it is stated otherwise, we use spatial pyramids with $R = 4$ regions (*i.e.* the entire image and three horizontal stripes).

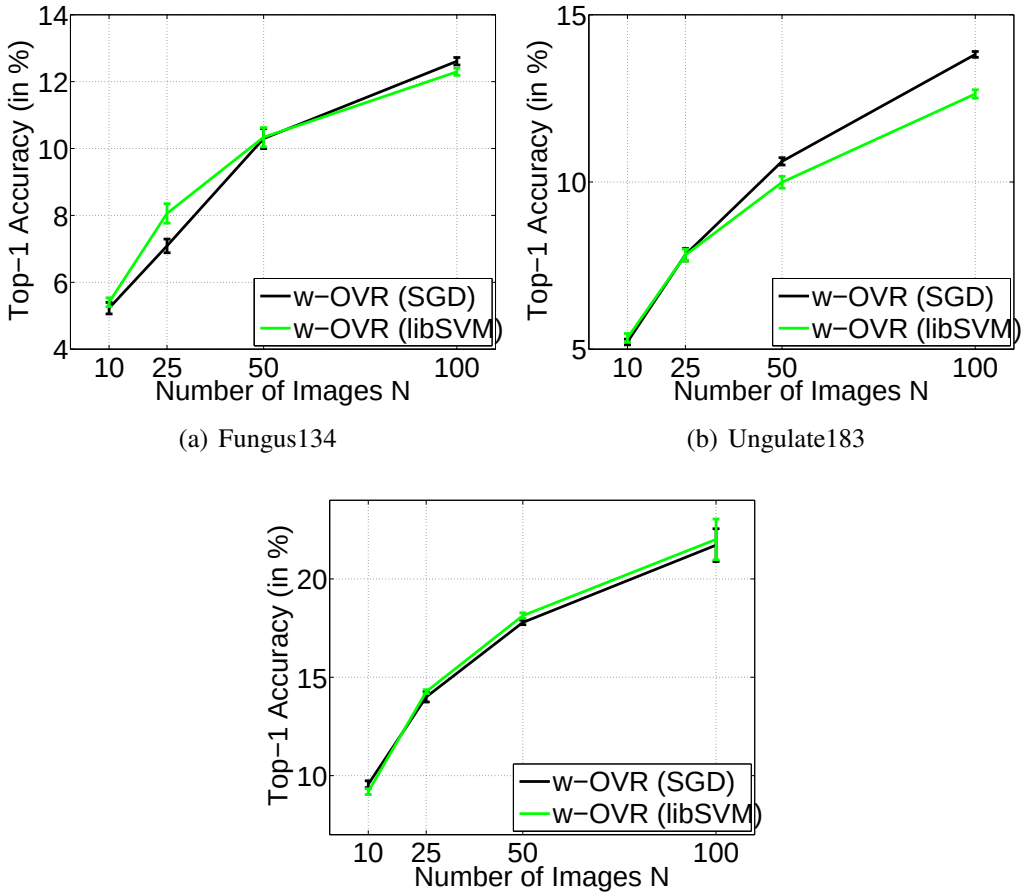
Considering that high dimensional image descriptors such as 130K-dim FV have large memory footprints, our experiments on the large-scale datasets require image signatures to be compressed. For the FV we employ Product Quantization (PQ) Jégou et al. (2011). We use sub-vectors of 8 dimensions and 8 bits per sub-vector. This setting was shown to yield minimal loss of accuracy in Sánchez and Perronnin (2011). For the BOV we use Scalar Quantization with 8 bits per dimension as suggested in Deng et al. (2010). Signatures are decompressed on-the-fly by the SGD routines Sánchez and Perronnin (2011).

3.5.1 Fine Grained Experiments

Distinguishing between the semantically related categories which are also visually very similar is referred to as fine-grained visual categorization (see Deselaers and Ferrari (2011) for a study of the relationship between semantic and visual similarity in ImageNet). It has several applications such as the classification of mushrooms for edibility, animals for environmental monitoring or vehicles for traffic related applications. In this section we analyze the performance of the objective functions described in Section 3.3 in the context of fine-grained visual categorization. We report the results on three subsets of ImageNet: Fungus134, Ungulate183 and Vehicle262 Deng et al. (2010).

Comparison between SGD and Batch Methods. It is known that SGD can perform as well as batch solvers for OVR SVMs at a fraction of the cost Bottou and Bousquet (2007); Shalev-Shwartz et al. (2007). However, public SGD solvers do not exist for other SVM formulations such as w-OVR, MUL, RNK or WAR. Hence, as a sanity check, we compared our w-OVR and MUL SGD solvers to publicly available batch solvers. We compared our w-OVR SGD to LibSVM Chang and Lin (2011) and MUL SGD to LibLinear Fan et al. (2008) and SVM^{light} Joachims (1999). Given the cost of training batch solvers, we restrict our experiments to the small dimensional BOV vectors ($(N = 1,024) \times (R = 4) \rightarrow 4,096$ dimensions).

Furthermore, we perform training on the subsets of the full training sets: for each class we use 10, 25, 50 and 100 random samples within the training set. We repeat the experiments



(a) Fungus134

(b) Ungulate183

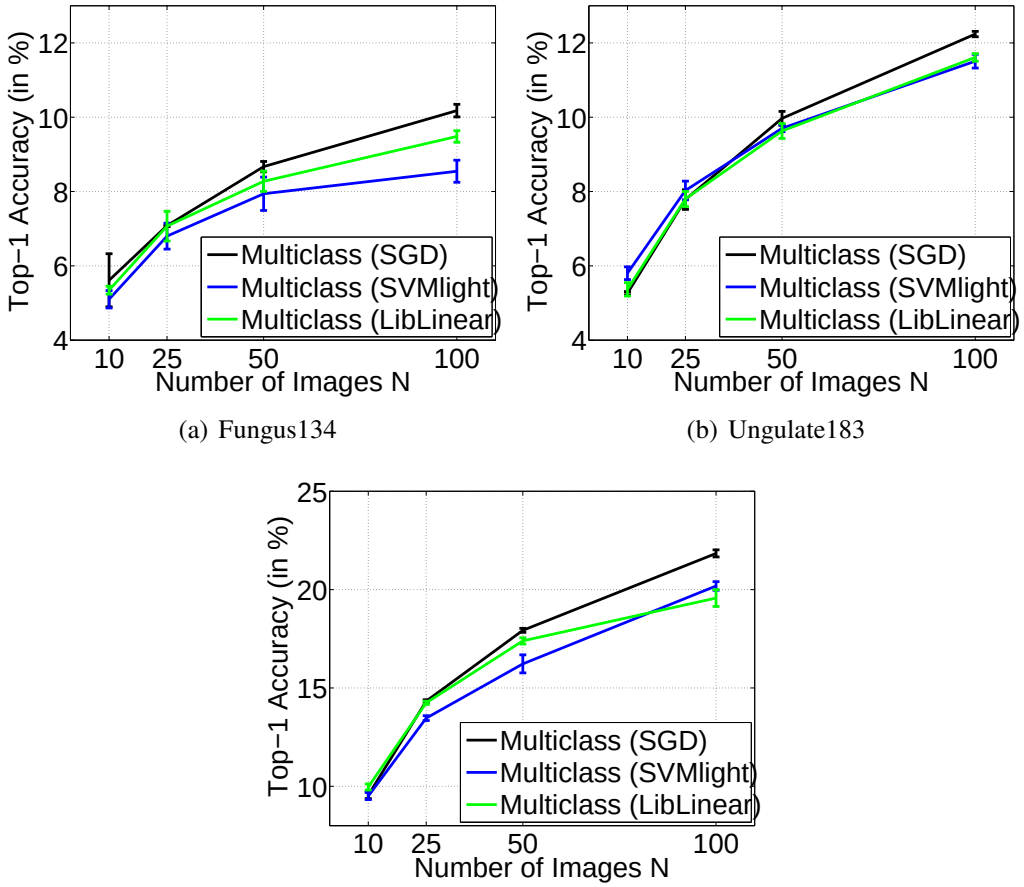
(c) Vehicle262

Figure 3.1 *Comparison of SGD and libSVM for optimizing the w-OVR SVM using 4,096-dim BOV descriptors. The number of images per class for training vary between 10, 25, 50 and 100. The experiments with 10 and 25 images per class are repeated 100 times, with 50 and 100 images are repeated 5 times. The SGD and batch methods perform similarly.*

5 different times for 5 random subsets of training images. With this setting, the top-1 classification accuracy is reported in Figure 3.1 and 3.2.

These experiments show that stochastic and batch solvers perform on par for w-OVR and MUL SVMs. Somewhat surprisingly, SGD solvers even seem to have a slight edge over batch solvers, especially for the MUL SVM. This might be because SGD uses as stopping criterion the accuracy on the validation set measured with the true target loss (top-1 loss in this case). In contrast, batch solvers stop upon convergence of a surrogate objective function on the training set.

The training times of SGD and batch solvers are reported in Table 3.3 (in CPU seconds on 32GBs RAM double quad-core multi-threaded servers using a single CPU). As ex-



(a) Fungus134

(b) Ungulate183

(c) Vehicle262

Figure 3.2 *Comparison of SGD, SVM^{light} and LibLinear solvers for MUL using using 4,096-dim BOV. In training, we use 10, 25, 50, 100 images from each class. The experiments are repeated for 5 times. The SGD and batch methods perform similarly. Only when the number of training images is high, the SGD method has an edge over the batch methods.*

	LibSVM (batch) / OVR SGD (online)			SVM^{light} (batch) / MUL SGD (online)		
	Fungus134	Ungulate183	Vehicle262	Fungus134	Ungulate183	Vehicle262
10	12 / 7	31 / 18	107 / 39	45 / 36	324 / 81	557 / 209
25	95 / 16	175 / 36	835 / 119	99 / 72	441 / 198	723 / 369
50	441 / 38	909 / 67	3,223 / 271	198 / 261	855 / 420	1,265 / 747
100	1,346 / 71	3,677 / 133	11,679 / 314	972 / 522	1,674 / 765	3,752 / 1,503

Table 3.3 *Average training time (in CPU seconds) for batch,* i.e. *LibSVM or SVM^{light} and SGD solvers on Fungus134, Ungulate183, Vehicle262 for 10, 25, 50, 100 training samples per class using 4,096-dim BOV. The SGD method works consistently faster than batch methods.*

pected, the CPU time of SGD solvers is significantly smaller than for batch solvers and the difference increases for larger training sets.

Importance of Data Rebalancing in OVR In this section we investigate the effect of data rebalancing for OVR SVMs. Note that more sophisticated strategies could also be used, as in Tao et al. (2006). Here we simply emphasize that in terms of the performance, it is always beneficial to cross-validate the imbalance parameter of one-vs-rest classifiers. We compare unweighted OVR (u-OVR) and weighted OVR (w-OVR). Since we do the reweighting by biasing the sampling, we introduce the imbalance parameter $\beta = (1-\rho)/\rho$ which is the (average) number of negatives sampled for each positive. $\beta = 1$ corresponds to the natural rebalancing of the data (giving the same weight to positives and negatives).

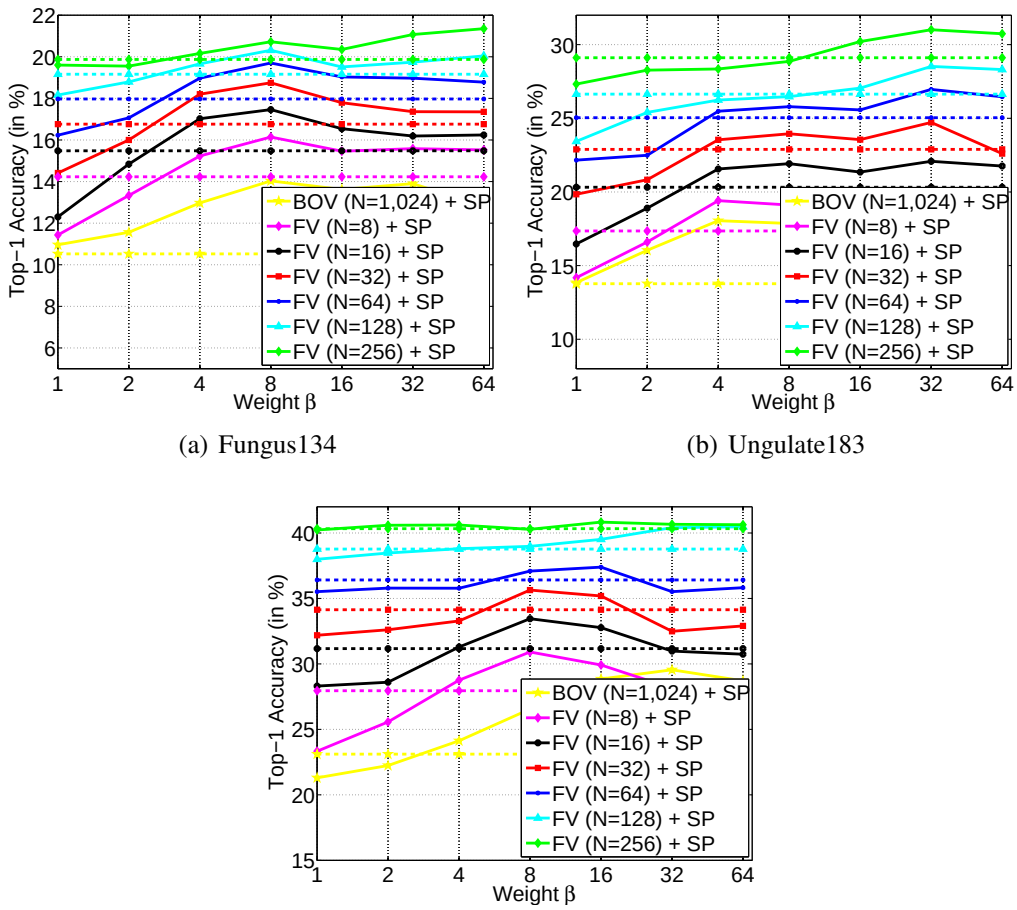
The solid lines in Figure 3.3 show the results with w-OVR and the dashed lines with u-OVR. The experiments are performed on BOV descriptors with $N = 1024$ Gaussians and FV descriptors with $N = 8, 16, 64$, and 256 Gaussians using a pyramid with $R = 4$ regions. Here, it is beneficial also to cross-validate step size and the regularization parameter for each configuration separately.

We can draw the following conclusions. First, rebalancing makes a significant difference for smaller dimensional features, but has less impact on high-dimensional features. Second, it is crucial to correctly set the parameter β, as natural rebalancing generally does not lead to the best results. Indeed, the optimal β depends on many factors including the dataset, the feature type and the feature dimensionality. We observe empirically that the optimal β typically increases with the number of classes. For instance, on the ILSVRC 2010 and the ImageNet10K respectively, the best w-OVR results are obtained with $\beta = 64$ and $\beta = 256$ (with the 4K-dim BOV features). In the following, we use the weighted version of OVR, *i.e.* w-OVR.

Explicit *vs*. Implicit Regularization The experiments in this section are performed on high-dimensional features (the 130K-dim FV for $N = 256$ and $R = 4$), *i.e.* for which regularization is supposed to have a significant impact. The following experiments focus on w-OVR but we obtained very similar results for MUL, RNK and WAR. We compare three regularization strategies:

(i) Using explicit regularization ($\lambda > 0$) and a decreasing step size $\eta_t = 1/(\lambda(t + t0)$. Setting t_0 correctly is of paramount importance for fast convergence and Bottou (1997) proposes a heuristic to set t_0 automatically[4]. Cross-validation on t_0 showed that the optimal value is very close to the one predicted by Bottou's heuristic.

[4]This heuristic is based on the assumption that the input vectors are ℓ_2-normalized and uses the fact that the norm of the optimal w, denoted w^*, is bounded Shalev-Shwartz et al. (2007): $||w^*|| \leq 1/\sqrt{\lambda}$. t_0 is set so that the norm of w during the first iterations is comparable to this bound.



(a) Fungus134 (b) Ungulate183

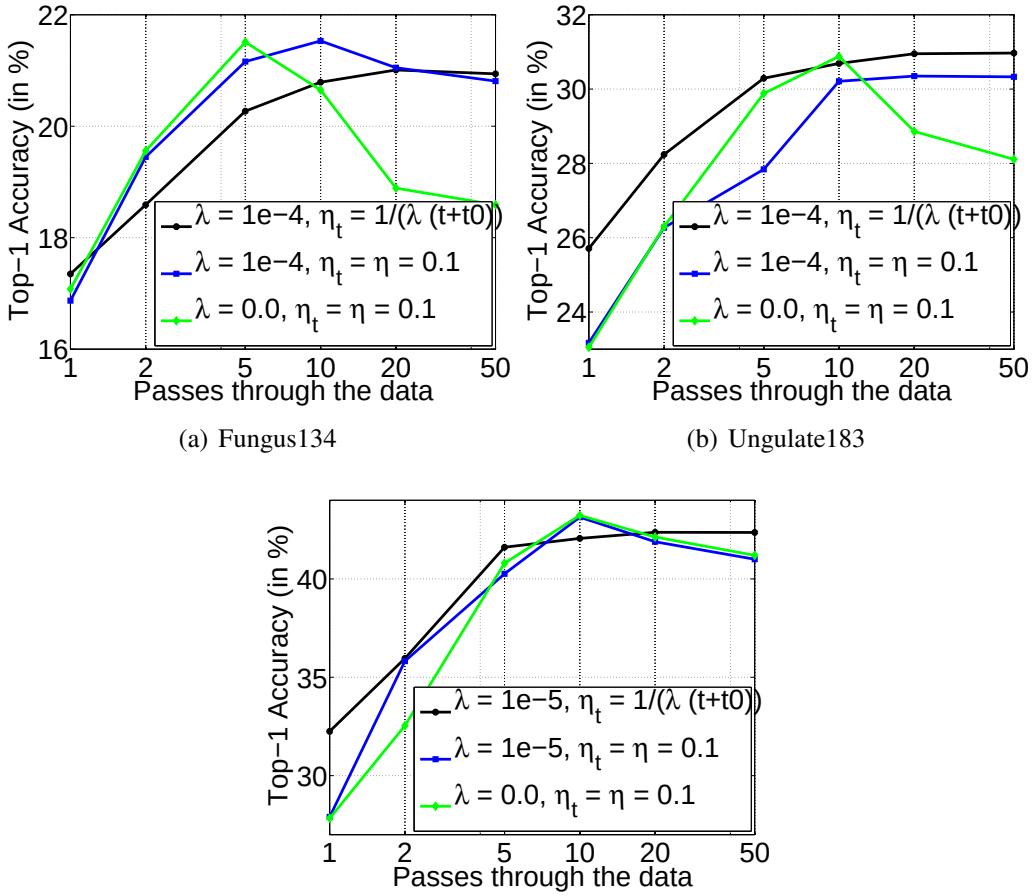
(c) Vehicle262

Figure 3.3 *Influence of data rebalancing in w-OVR. The value of the weight parameter,* i.e. *1, 2, 4, 8, 16, 32 and 64 indicates the number of negative training images sampled for each positive image (solid lines). The results of u-OVR are shown with dashed lines. The dimensionality of the BOV is ≈ 4K and FV varies between ≈ 4K, 8K 16K, 32K, 64K and 130K. When the FV dimensionality is high, rebalancing does not increase the accuracy. However, with small dimensional features, data rebalancing is beneficial.*

(ii) Using explicit regularization $\lambda > 0$ and a fixed step size $\eta_t = \eta$.

(iii) Using no regularization ($\lambda = 0$) and a fixed step size $\eta_t = \eta$, *i.e.* implicitly regularizing with the number of iterations.

The results of these experiments are provided in Figure 3.4 with β set by cross-validation. All three strategies perform similarly showing that implicit regularization with fixed step size can be an effective learning strategy. We observe that for smaller datasets it is important to stop early for optimal performance. This is consistent with regularization being less important when training data is plentiful. We also experimented with the implicit



(a) Fungus134

(b) Ungulate183

(c) Vehicle262

Figure 3.4 *Impact of regularization on w-OVR. Results on Fungus134, Ungulate183, Vehicle262 using 130K-dim FVs. The comparison is between (i) explicit regularization with variable step size, (ii) explicit regularization with constant step size, (iii) implicit regularization with constant step size through early stopping. One pass means visiting all the positives for a given class + (on average) β times more negatives. Implicit and explicit regularization gives similar results but we choose implicit regularization since it has one less parameter to tune.*

regularization with early stopping on the small-scale PASCAL VOC 2007 dataset Everingham et al. (2010) and observed a small drop of performance compared to explicit regularization (from 62.1% to 60.2%) which might indicate that this strategy is better suited to large-scale datasets.

As a conclusion, there is no significant difference between the performances of the experiments with respect to the regularization strategies. In strategy (i), which is implicit regularization and variable step size, in Fungus134 and Ungulate183 there is no improvement with the number of iterations. In Vehicle262, the strategy (i) gives slightly lower

results and converges more slowly in comparison to the others. The strategy (iii) which is the implicit regularization and fixed step size gives better results and converges faster than the strategies (i) and (ii). A major advantage of the strategy (iii) with respect to the strategies (i) and (ii) is that there is a single parameter to tune (the number of iterations $niter$) instead of two ($niter$ and λ). Given the training times in large-scale, the strategy (iii) is the approach we choose in our truly large scale experiments.

Analysis of Sampling in w-OVR In this section, we evaluate the effect of sampling the negative data points with or without replacement during SGD training for w-OVR. We experiment with our highest dimensional features, *i.e.* the 130K-dim FV for compatibility with our previous results. The results are reported in Table 3.4 in terms of top-1 accuracy.

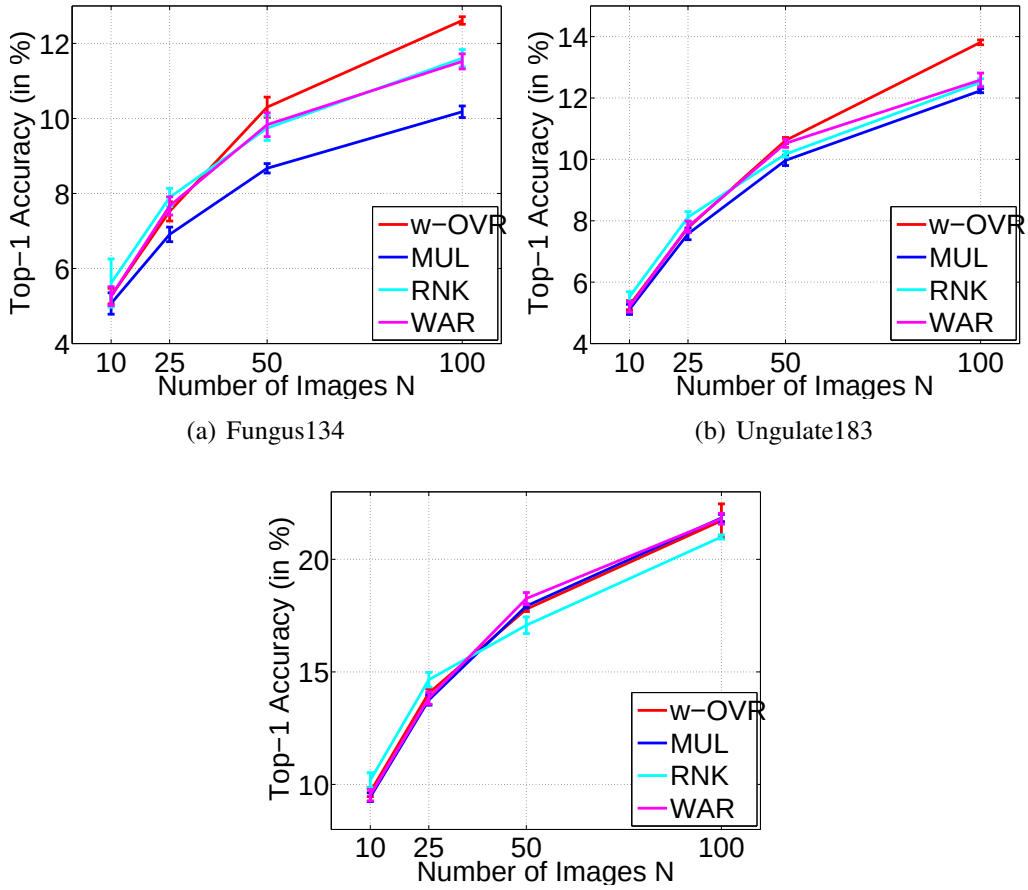
	Fungus134	Ungulate183	Vehicle262
no replacement	**19.75**	**27.86**	38.40
replacement	19.55	27.79	**39.53**

Table 3.4 *Comparison of sampling with and without replacement for w-OVR. Top-1 accuracy (%) on Fungus134, Ungulate183, Vehicle262 using 130K-dim FVs. The results show that there is not a big difference in terms of accuracy between these different sampling strategies.*

The results show that there is not a big difference between the accuracies obtained with sampling without replacement and sampling with replacement. The conclusion is that sampling without replacement does not improve the accuracy significantly. In what follows, we draw negative samples with replacement.

Influence of Training Data Size The aim of the experiments in this section is to understand whether the amount of training data has an impact on the relative performance of the four different objective functions: w-OVR, MUL, RNK and WAR. We sample a fraction of data from the whole training set. The experiments with 10 or 25 training images sampled per class are repeated 100 times. For the configuration where we select 50 or 100 images per class, we repeat the experiments 5 times. The image descriptors are 4,096 dimensional BOV descriptors and the results can be seen in Figure 3.5.

Despite its simplicity and its supposed sub-optimality, w-OVR provides a competitive performance. It seems that RNK has an edge over the three other objective functions for a very small number of training samples. In Table 3.5 the numerical results for 10 and 25 training samples are also provided. To understand whether the observed differences are significant, we performed two types of significance tests. The sign test Sheskin (2007); Salzberg (1997) counts the number of times an algorithm performs better than the other one and the paired t-test determines whether the observed differences in accuracies are



(a) Fungus134 (b) Ungulate183

(c) Vehicle262

Figure 3.5 *Comparison of w-OVR, MUL, RNK and WAR as a function of the number of training images,* i.e. *10, 25, 50, 100 on Fungus134, Ungulate183, Vehicle262 using 4,096-dim BOV. Despite its simplicity and its supposed sub-optimality, w-OVR provides a competitive performance. It seems that RNK has an edge over the three other objective functions for a very small number of training samples.*

purely due to random errors. Table 3.6 shows the probability for both tests to reject the null hypothesis, *i.e.* the probability that the observed differences happen by chance. In all cases the probabilities are close to zero, *i.e.* the null hypothesis is rejected with high confidence, thus indicating that the modest improvement of RNK with respect to w-OVR, MUL and WAR for a small number of training samples is significant.

The conclusions that can be drawn from these experiments are as follows. The OVR objective function for a given class can be written as a sum of losses over samples (x_i, y_i). In contrast, using the weighted ranking framework of Usunier et al. (2009) the MUL, RNK and WAR objective functions can be written as sums of losses over triplets (x_i, y_i, y). Since the number of triplets is C times larger than the number of samples, one could expect the latter objective functions to have an edge over OVR in the small sample regime.

	Mean (μ) / Std deviation (σ)					
	10 training images per class			25 training images per class		
	Fungus134	Ungulate183	Vehicle262	Fungus134	Ungulate183	Vehicle262
w-OVR	5.26 / 0.20	5.24 / 0.14	9.66 / 0.20	7.52 / 0.25	7.77 / 0.15	14.05 / 0.19
MUL	5.06 / 0.28	5.11 / 0.16	9.43 / 0.19	6.90 / 0.19	7.57 / 0.19	13.75 / 0.23
RNK	**5.61 / 0.64**	**5.52 / 0.26**	**10.18 / 0.33**	**7.89 / 0.24**	**8.11 / 0.18**	**14.65 / 0.32**
WAR	5.26 / 0.25	5.19 / 0.17	9.52 / 0.25	7.66 / 0.24	7.80 / 0.18	13.83 / 0.27

Table 3.5 *The mean and standard deviation of the top-1 accuracy in on Fungus134, Ungulate183, Vehicle262 with 10 and 25 training images per class (repeated 100 times) on w-OVR, MUL, RNK and WAR using 4,096-dim BOV. The statistical consistency is measured with t-test and sign-tests. RNK performs consistently better than all the other methods.*

		p-values (t-test, sign test)		
		RNK vs MUL	RNK vs WAR	RNK vs w-OVR
Fungus134	10	9.82e-12, 5.07e-20	5.48e-6, 5.52e-17	4.81e-7, 9.44e-20
	25	9.32e-51, 3.15e-28	1.39e-10, 2.49e-08	3.55e-20, 1.42e-13
Ungulate183	10	3.16e-34, 5.10e-25	5.97e-25, 3.31e-18	1.03e-20, 3.77e-21
	25	2.69e-36, 1.23e-23	1.84e-18, 1.04e-16	4.49e-26, 3.31e-18
Vehicle262	10	1.36e-35, 5.58e-19	2.06e-26, 5.58e-19	1.23e-26, 3.31e-18
	25	8.34e-35, 3.31e-18	2.74e-30, 3.31e-18	3.02e-33, 3.31e-18

Table 3.6 *RNK method in comparison with MUL,WAR and w-OVR on Fungus134, Ungulate183, Vehicle262 using 10 and 25 training images per class. The image descriptors are 4,096-dim BOV. This table shows that the high accuracy obtained with RNK over the other methods are statistically consistent.*

However, while the RNK objective function is an unweighted sum over triplets, the MUL and WAR objective functions correspond to weighted sums that give more importance to the top of the list. In other words, because of the weighting, the effective number of triplets is much smaller than the actual number of triplets in the objective function. For instance, in the case of MUL, only the top triplet has a non-zero weight which means that the effective number of triplets equals the number of samples. This may explain why eventually MUL does not have an edge over OVR in the small sample regime.

Similarly, the WAR weighting can be viewed as a smooth interpolation between the MUL weighting and the RNK weighting. Although the WAR weighting – which is inversely proportional to the triplet rank – makes sense for our classification goal, it is by no means guaranteed to be optimal. While Usunier et al. (2009) propose alternative weight profiles (see Section 6 in Usunier et al. (2009)), we believe that such weights should be learned from the data in order to give full strength to the WAR objective function. However, it is

not straightforward to devise a method to learn these weights from the data. Furthermore, from our experiments, the optimal weighting profile seems to depend on the number of training samples.

Influence of Class Density The density of a dataset Deng et al. (2010) is defined as the mean distance between all pairs of categories, where the distance between two categories in a hierarchy is the height of their lowest common ancestor. Small values imply dense datasets and indicate a more challenging recognition problem. Note that the Fungus134 is the densest of all three fine-grained datasets that are used in our experimental validation.

In order to better understand the effect of dataset density, we create 6 synthetic datasets that are composed of 1,000 sample points drawn from 5 Gaussians with the same mean and 6 different standard deviations *e.g.* $\sigma \in \{0.1, 0.25, 1, 2, 3, 5\}$. Figure 3.6 shows the synthetic dataset for the standard deviation of 0.25. As the standard deviation increases, the overlap between classes increases significantly.

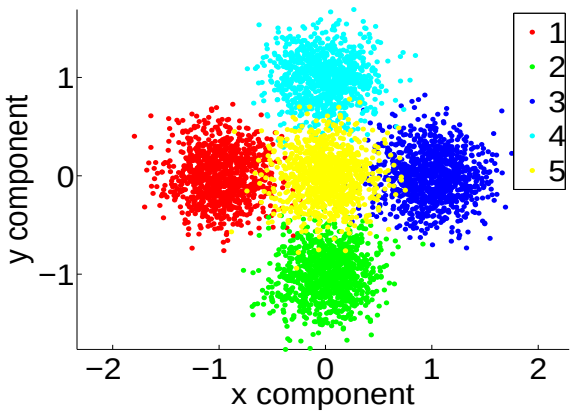


Figure 3.6 *Illustration of the synthetic dataset with standard deviation 0.25. Each color represents a different class and each point represents a sample in the dataset.*

$\sigma =$	0.1	0.25	1.0	2.0	3.0	5.0
w-OVR	100.00	96.30	47.04	33.08	27.78	24.48
MUL	100.00	95.70	44.96	26.70	23.32	22.36
RNK	100.00	96.28	47.76	33.24	27.66	24.30
WAR	100.00	96.32	47.48	32.98	27.62	24.62

Table 3.7 *Comparison of the top-1 accuracies (%) between w-OVR, MUL, RNK and WAR on synthetic data with 5 classes.* $\sigma = \{0.1, 0.25, 1.0, 2.0, 3.0, 5.0\}$ *are the standard deviation of the Gaussians that determine the sample distribution and the difficulty of the synthetic dataset.*

Table 3.7 shows that all classifiers yield perfect results when the data are well separable, but MUL performs worse than all the other methods in case of overlapping distributions. These results are in agreement with teh results in Figure 3.5 where MUL performs poorly, especially on the dense Fungus134 dataset. As a sanity check for the MUL implementation that uses the SGD oprimization, Table 3.8 shows the comparison between LibLinear and SVM^{light} solvers in comparison with MUL SGD solver.

$\sigma =$	0.1	0.25	1.0	2.0	3.0	5.0
MUL (SGD)	100.00	95.70	44.96	26.70	23.32	22.36
MUL (LibLinear)	100.00	96.16	46.72	27.46	23.46	22.26
MUL (SVM^{light})	100.00	96.24	46.74	24.96	23.36	19.56

Table 3.8 *Comparison of the accuracy (in %) with MUL formulation optimized with the online SGD method and the batch methods LibLinear and SVM^{light} on the synthetic data. σ is the standard deviation of the Gaussians that determine the sample distribution and the difficulty of the dataset.*

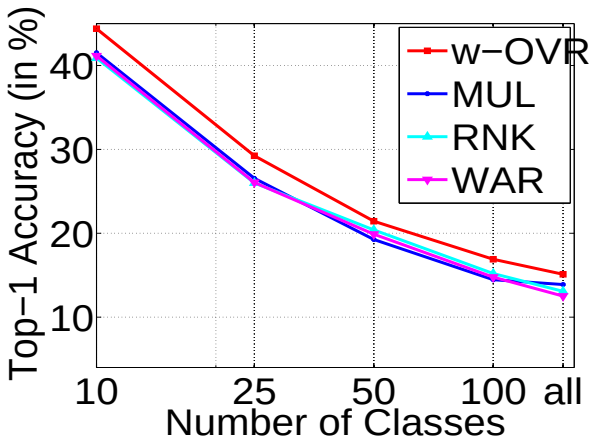
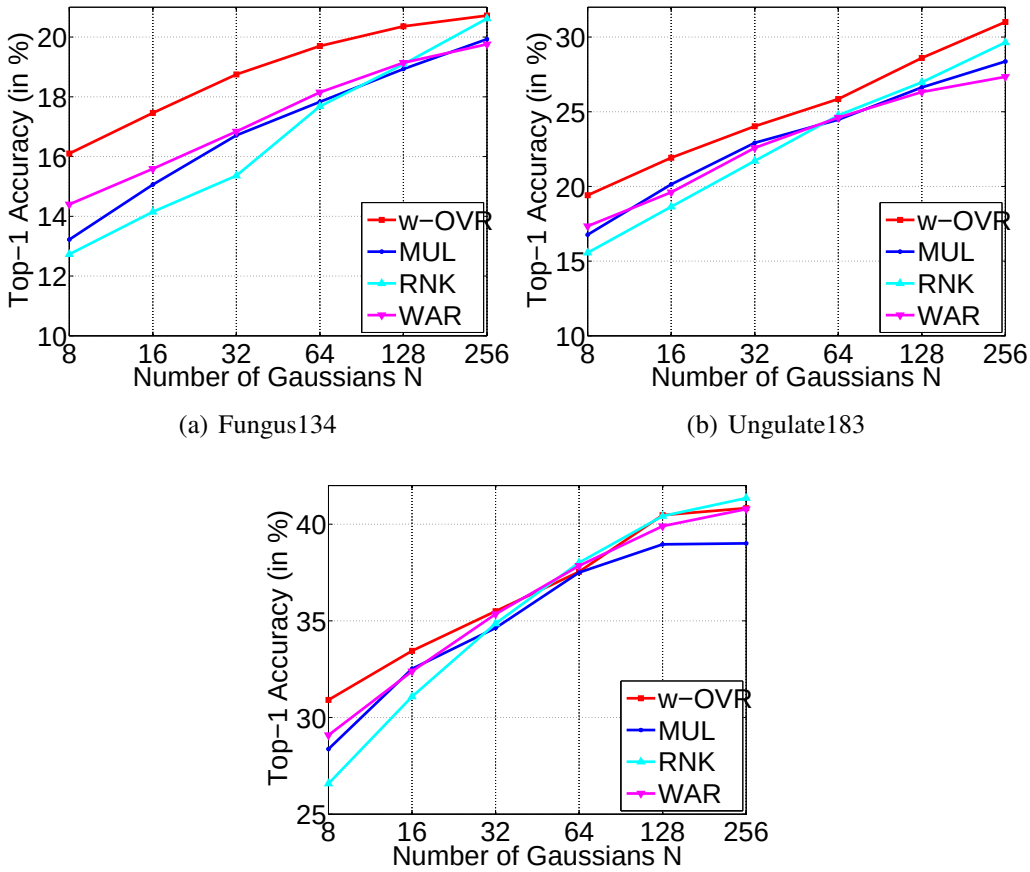


Figure 3.7 *Experiments on dataset density for Fungus134. Plot shows the top-1 accuracy as a function of the number of classes using BOV with 4,096 dimensions. To vary the density, different numbers of classes are subsampled: 10, 25, 50, 100. A small number of classes would typically result in a lower class density. The ordering of the classifiers does not seem to be dependent on the number of classes and therefore on the data density.*

In order to test this idea with a standard dataset, we performed further experiments on the Fungus134 dataset. To vary the density, we subsample different numbers of classes such as 10, 25, 50, 100 classes from the whole dataset. The training and test partition is similar to the previous configuration, *i.e.* we select the same 10, 25, 50 and 100 classes from training, validation and test sets. A small number of classes would typically result in a lower class density. The results with BOV vectors can be seen in Figure 3.7. The ordering of the classifiers does not seem to be dependent on the number of classes and therefore

on the data density. According to the experimental results obtained on these synthetic and real data experiments, it is therefore difficult to argue for the merits of one classifier or the other for denser or sparser datasets.

Influence of Feature Dimensionality In order to investigate the effect of the feature dimensionality on the accuracy, we compare the four objective functions (w-OVR, MUL, RNK and WAR). Figure 3.8 shows the results for the FVs of different dimensionality, i.e. the number of Gaussians varies from $N = 8$ to $N = 256$.



(a) Fungus134

(b) Ungulate183

(c) Vehicle262

Figure 3.8 *Comparison of Top-1 Accuracy (in %) between the w-OVR, MUL, RNK and WAR as a function of the FV dimensionality (with no spatial pyramids). As the descriptor dimensionality increases, the difference between the different objective functions gets smaller.*

Figure 3.8 shows that the w-OVR provides a competitive performance with more complicated ranking methods. The classification accuracy obtained with w-OVR using small image signatures is 4 points higher than the other methods but with the increasing dimensionality size, the difference gets smaller. Moreover, as the feature dimensionality

increases, the difference in accuracy between the different objective functions decreases. Hence, for high-dimensional features, the learning objective function seems to have little impact. Put into the perspective of the statistical learning theory Bartlett et al. (2003), this implies that the impact of the different choices of surrogate loss is mitigated as the capacity of the classifier increases. In Bartlett et al. (2003), the $\psi(\cdot)$ function defines how the performance in the surrogate loss transfers to the target loss (see Theorem 10 in Bartlett et al. (2003)). For each surrogate loss there is a corresponding $\psi(\cdot)$ function. In the Figure 3.8, one can see that the different learning objectives corresponding to different surrogate losses lead to similar, yet, good results. Therefore, in these experiments, the capacity of the classifiers at hand is "large" enough, relative to the difficulty of the dataset, so large that the difference of the corresponding $\psi(\cdot)$ functions is almost canceled.

3.5.2 Large-Scale Experiments

In this section, the experiments with the ILSVRC 2010 Berg et al. (2010a) and the ImageNet10K datasets are explained. These two datasets are large scale with respect to the number of classes and the number of images, such that The ILSVRC 2010 contains 1K classes, 1.2M images and the ImageNet10K contains 10K classes, 9M images.

1) Experiments on the ILSVRC 2010 dataset

In this section, we perform the experiments on the ILSVRC 2010 dataset measuring the influence of several parameters such as reweighting, regularization, the image descriptor dimensionality and compare the four objective functions, *i.e.* OVR (unweighted and weighted), MUL, RNK, WAR.

Influence of data rebalancing. First, we compare the effect of the dimensionality of image descriptors in the large scale setting. We use BOV with $D_{BOV} = 4,096$ dimensions and FV with $D_{FV} \in \{2K, 8K, 32K, 130K\}$ dimensions. We vary the proportion of negative and positive samples through the parameter $\beta \in \{1, 2, 4, 8, 16, 32, 64\}$.

We observe from Figure 3.9 that carefully tuning β on small dimensional image representations has a significant impact on accuracy in the truly large scale datasets. Especially for smaller dimensional representations, the natural rebalancing, *i.e.* $\beta = 1$ is insufficient. However, when the image descriptors are large enough, *e.g.* $D_{FV} = 130K$ the effect of the imbalance parameter β diminishes. For the largest FV, cross-validating β has a small effect.

On the other hand, reweighting the data for MUL, RNK and WAR has virtually no impact on accuracy. One of the reasons is that, the data is much less unbalanced for MUL, RNK and WAR than for OVR. Indeed, the ratio between the number of samples in the most

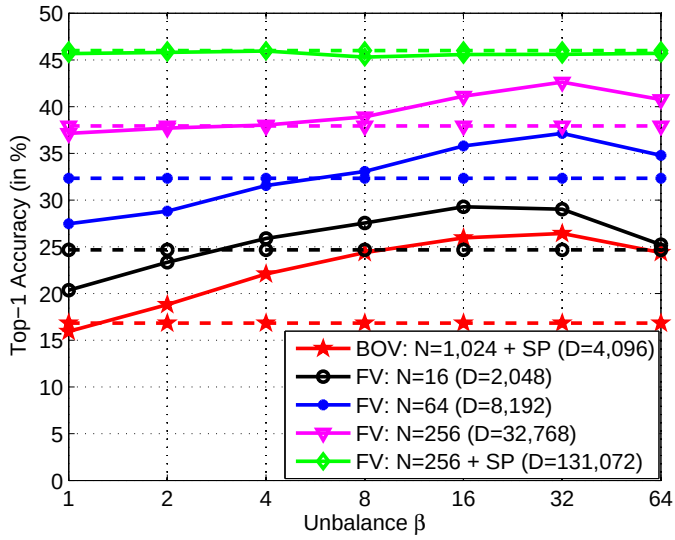


Figure 3.9 *Impact of imbalance $\beta = (1-\rho)/\rho$ on the accuracy on ILSVRC 2010. The plain lines correspond to w-OVR while the dashed lines correspond to u-OVR (which is independent of β). For each method, N is the number of Gaussians, SP indicates spatial pyramids and D is the dimensionality of the descriptors. Similar curves were obtained for top-5 accuracy.*

and least populated classes on the ILSVRC 2010 is approximately 4. However, in the OVR case the imbalance for a given class is the ratio between the number of positive and negative samples and is therefore on the order of 999.

Influence of the regularization parameter. In our second set of experiments, we evaluate the importance of the regularization parameter for the ILSVRC 2010 dataset. We compare (i) implicit regularization with decreasing step-size, (ii) implicit regularization with constant step size and (iii) explicit regularization. Similar to the fine grained experiments reported in the previous sections, the results that are shown in Figure 3.10 indicate that the standard approach by Bottou (2003), is slightly faster to converge in the first iterations. On the other hand, at convergence, all the three methods yield similar results. However, the implicit regularization has the advantage of requiring one less parameter to tune.

Influence of descriptor dimensionality. Similar to the results obtained with fine grained datasets, when high-dimensional FV features are used in the experiments with the ILSVRC 2010 dataset, all the methods perform similarly. For instance, the difference between the best and worst performing methods is 0.5% at top-1 and 2.8% at top-5 (see Table 3.9).

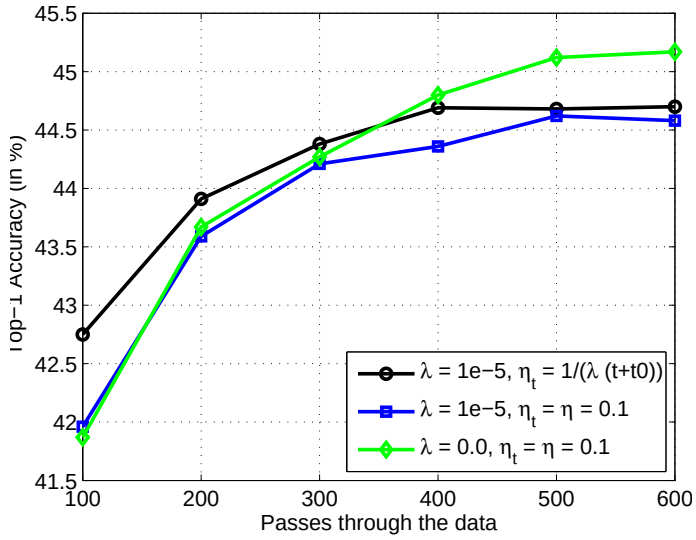


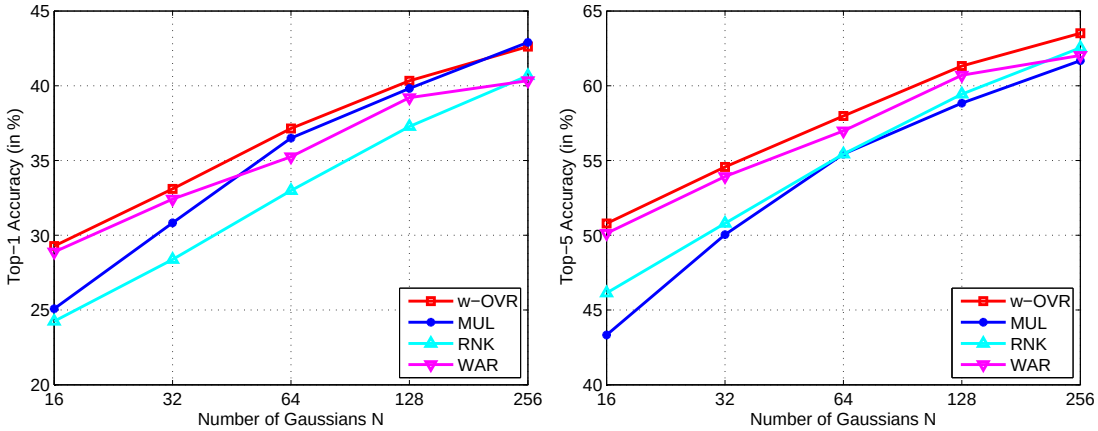
Figure 3.10 *Impact of regularization on w-OVR (with $\beta = 1$). Results on the ILSVRC 2010 with 130K-dim FVs. One pass through the data corresponds to seeing all the positive samples for a given class + (on average) as many negatives as these positive samples. Hence, 500 passes is approximately as costly as seeing each sample of the dataset once (because there are 1,000 classes). Similar curves were obtained for top-5 accuracy.*

		u-OVR	w-OVR	MUL	RNK	WAR
Top-1	BOV	15.8	26.4	22.7	20.8	24.1
	FV	45.9	45.7	46.2	46.1	46.1
Top-5	BOV	28.8	46.4	38.4	41.2	44.2
	FV	63.7	65.9	64.8	65.8	66.5

Table 3.9 *The comparison between plain OVR, weighted OVR (the rebalance parameter β is cross-validated), MUL, RNK and WAR training objectives for linear SVMs. The accuracy (in %) is measured on the standard ILSVRC 2010 dataset for 4K-dim BOV and 130K-dim FV.*

We report the results of the experiments that measure the influence of the FV dimensionality in Figure 3.11. As the number of Gaussians N increases, *i.e.* as the FVs grow larger, the difference of accuracy between different methods becomes smaller.

The second observation from Figure 3.11 is that, w-OVR always performs the best and is closely followed by WAR. As expected, MUL which focuses on the first rank performs better at top-1 than at top-5, while RNK which optimizes the rank of the correct labels is more competitive for top-5. An important conclusion is that, despite its simplicity and its theoritical suboptimality, OVR is a competitive alternative to more complex objective functions on ILSVRC 2010.



(a) Top-1 (b) Top-5

Figure 3.11 *Impact of the number of Gaussians N (and therefore of the feature dimensionality) on the ILSVRC 2010 dataset. Left: top-1. Right: top-5. Because of the cost of running these experiments, these results are computed on FV without SPs.*

Comparison with the State-of-the-Art. The two winning teams at the ILSVRC 2010 reported better results than the ones we reported in this work as 66.5% at top-5. Sánchez and Perronnin (2011) reports 74.3% by combining SIFT and color descriptors and by using 520K-dim FVs while Lin et al. (2011) reports 71.8% by combining SIFT and LBP descriptors as well as multiple encoding techniques and spatial pyramids. While better features can indeed increase accuracy, this is out of the scope of this work.

2) Experiments on the ImageNet10K dataset

In this section, we performed our experiments at the scale of $O(10^4)$ categories as in Deng et al. (2010); Sánchez and Perronnin (2011); Weston et al. (2010). Table 3.10, shows the results for the 4K-dim BOV and the 130K-dim FV using top-1 accuracy as in Deng et al. (2010); Sánchez and Perronnin (2011) on the ImageNet10K dataset which contains approximately 10K classes and 9M images. Given the cost of running the experiments on the high-dimensional FVs, the experiments have been carried-out only with the two objective functions which performed best on the ILSVRC 2010 dataset: w-OVR and WAR.

As a conclusion, w-OVR performs better than more complex objective functions. Some example classes from ImageNet10K which reflect the difficulty of the dataset along with the top-1 per-class accuracies are presented in Figure 3.12. Figure 3.13 gives more detailed per-class results.

Comparison with the State-of-the-Art. Compared to Deng et al. (2010), our BOV results are on par (even slightly better since we report 7.5% while they report 6.4%)

	u-OVR	w-OVR	MUL	RNK	WAR
BOV 4K-dim	3.8	7.5	6.0	4.4	7.0
FV 130K-dim	-	19.1	-	-	17.9

Table 3.10 *Comparison between OVR, MUL, RNK and WAR on ImageNet10K. Reported results are top-1 accuracy (in %). As a conclusion, w-OVR performs better than more complex objective functions (at least on this dataset with those features).*

(a) Star Anise (92%) (b) Nest Egg (87%) (c) Geyser (86%)

(d) Traction engine (77 %) (e) Ready to Wear (76 %) (f) Stonechat (50%)

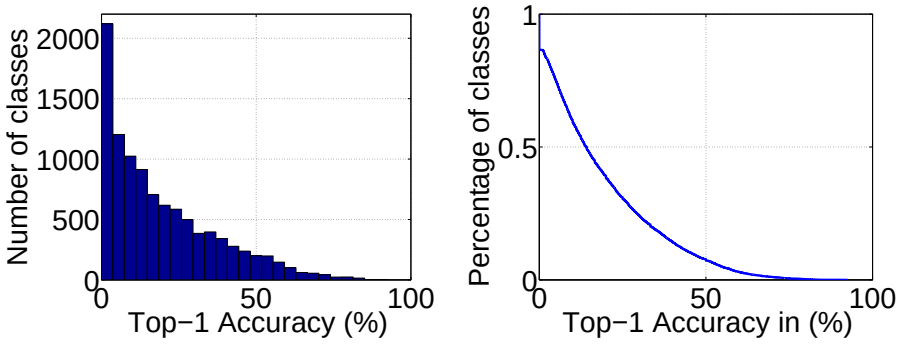
(g) Tortrix (25%) (h) Pyralid (25%) (i) Egyptian cobra (10%)

(j) Hare (5%) (k) Weasel (5%) (l) Felt fungus (0%)

Figure 3.12 *ImageNet10K results (top-1 accuracy in %) obtained with w-OVR and 130K-dim Fisher vectors. (a-c) Sample classes from the best performing ones, (d-f) sample classes with the accuracy of between 75 and 50%. (g-i) Sample classes with performance between 25 and 10%, (j-l) sample classes from the worst performing ones.*

and our FV results are significantly (almost 3 times) better due to the use of higher-dimensional features. However, our focus in this work is not on features but on (linear) classifier learning strategies.

Weston et al. (2010) show that WAR outperforms OVR on BOV descriptors in a different ImageNet subset. However, their OVR baseline do not reweight the positives/negatives, *i.e.* it is similar to our u-OVR. We also observed that WAR significantly outperforms u-OVR. Additionally, we show that w-OVR performs significantly better than u-OVR and slightly better than WAR.



(a) Histogram (b) Percentage

Figure 3.13 *Top-1 accuracies on ImageNet10K. Left: histogram of the top-1 accuracies of all the 10K classes. Right: percentage of classes whose top-1 accuracy is above a threshold.*

Previously Sánchez and Perronnin (2011) also report results with w-OVR with the natural rebalancing ($\beta = 1$). The results reported in this chapter improve their baseline by an absolute 2.4%, from 16.7% to 19.1% by cross-validating the rebalance parameter β in w-OVR setting and keeping other settings the same (*i.e.* using the same FV descriptors). It is interesting to note that while rebalancing the data has little impact on the 130K-dim FV on the ILSVRC 2010 dataset, it has a significant impact on the ImageNet10K dataset. This is not in contradiction with the fact that different objective functions perform similarly on high-dimensional image descriptors. In fact, the image descriptors are high-dimensional with respect to the complexity of the problem and especially the number of classes. While 130K-dim is high-dimensional with respect to the 1K categories of the ILSVRC 2010 dataset, it is not high-dimensional anymore with respect to the 10K categories of the ImageNet10K dataset.

Le et al. (2012) and Krizhevsky et al. (2012) also report results on the same subset of 10K classes. They report respectively a top-1 per-image accuracy of 19.2% and 32.6% [5]. The per-image accuracy corresponding to the experiments reported here is 21.0%. However, the optimization here has not been done with respect to this metric which might lead to better results. Note that this study is not about comparing FVs to features learned with deep architectures. Here, the learned features of Le et al. (2012); Krizhevsky et al. (2012) could be used if they had been available, i.e., as image descriptors, the output of any of the intermediate layers could have been used.

[5] While it is standard practice to report per-class accuracy on this dataset (see Deng et al. (2010); Sánchez and Perronnin (2011)), Le et al. (2012); Krizhevsky et al. (2012) report a per-image accuracy. This results in a more optimistic number since those classes which are over-represented in the test data also have more training samples and therefore have (on average) a higher accuracy than those classes which are under-represented. This was clarified through a personal correspondence with the first authors of Le et al. (2012); Krizhevsky et al. (2012).

Timing for ImageNet10K for 130K-dim FVs. For the computation a small cluster of machines with 16 CPUs and 32GB of RAM has been used. The feature extraction step (including SIFT description and FV computation) takes approximately 250 CPU days, the learning of the w-OVR SVM approximately 400 CPU days and the learning of the WAR SVM approximately 500 CPU days. Note that w-OVR performs slightly better than WAR and is much easier to parallelize since the classifiers can be learned independently.

3.6 Conclusion

In this chapter, we studied the visual classification problem on large-scale datasets, *i.e.* when we have to deal with a large number of classes, a large number of images and high dimensional features. Two main conclusions emerge from this chapter. The first one is that, despite its theoretical suboptimality, one-vs-rest is a very competitive training strategy to learn SVMs. Furthermore, one-vs-rest SVMs are easy to implement and to parallelize, *e.g.* by training the different classifiers on multiple machines/cores. However, to obtain state-of-the-art results, properly cross-validating the imbalance between positive and negative samples is a must. The second major conclusion is that stochastic training is very well suited to our large-scale setting. Moreover simple strategies such as implicit regularization with early stopping and fixed-step-size updates work well in practice. Following these good practices, we were able to improve the state-of-the-art on a large subset of 10K classes and 9M images from 16.7% Top-1 accuracy to 19.1%.

4

Label Embedding for Image Classification

Contents

4.1 Introduction **66**

4.2 Related Work **68**

4.3 Label Embedding with Attributes **70**

4.3.1 Framework 71

4.3.2 Parameter learning 73

4.3.3 Beyond attributes 75

4.4 Experiments **76**

4.4.1 Zero-Shot Learning 78

4.4.2 Few-Shots Learning 84

4.4.3 All Data Experiments 86

4.5 Conclusion **92**

Attributes act as intermediate representations that enable parameter sharing between classes, a must when training data is scarce. In this chapter, we propose to view attribute-based image classification as a label-embedding problem: each class is embedded in the space of attribute vectors. We introduce a function that measures the compatibility between an image and a label embedding. The parameters of this function are learned on a training set of labeled samples to ensure that, given an image, the correct classes rank higher than the incorrect ones. Results on the Animals With Attributes and Caltech-UCSD-Birds datasets show that the proposed framework outperforms the standard Direct Attribute Prediction baseline in a zero-shot learning scenario. The label embedding enjoys a built-in ability to leverage alternative sources of information in addition to attributes, *e.g.* class hierarchies or error correcting output codes. We further improve the baseline label embedding accuracy by utilizing continuous attributes that model confidence level of the presence of a

particular attribute. Moreover, label embedding encompasses the whole range of learning settings with side information, from zero-shot learning to regular learning with a large number of labeled examples.

4.1 Introduction

We consider the image classification problem where the task is to annotate a given image with one (or multiple) class label(s) describing its visual content. Image classification is a prediction task: the goal is to learn from a labeled training set a function $f : \mathcal{X} \rightarrow \mathcal{Y}$ which maps an input x in the space of images $\mathcal{X}$ to an output y in the space of class labels $\mathcal{Y}$. In this work, we are especially interested in the case where the classes are related (*e.g.* they all correspond to animals) but where we do not have any (positive) labeled sample for some of the classes. This problem is generally referred to as zero-shot learning Larochelle et al. (2008); Palatucci et al. (2009); Lampert et al. (2009); Farhadi et al. (2009). Given the impossibility to collect labeled training samples in an exhaustive manner for all possible visual concepts, zero-shot learning is a problem of high practical value.

A popular solution to zero-shot learning, called attribute-based learning, has recently gained popularity in computer vision. Attribute-based learning consists in introducing an intermediate space $\mathcal{A}$ referred to as the attribute layer Lampert et al. (2009); Farhadi et al. (2009). Attributes correspond to high-level properties of the objects which are shared across multiple classes. They can be detected by machines and understood by humans. Each class is represented as a vector of class-attribute associations according to the presence or absence of each attribute for that class. Such class-attribute associations are often binary. As an example, if the classes correspond to animals, possible attributes include "has paws", "has stripes" or "is black". For the class "zebra", the "has paws" entry of the attribute vector is zero whereas the "has stripes" would be one. The most popular attribute-based prediction algorithm requires learning one classifier per attribute. To classify a new image, its attributes are predicted using the learned classifiers and the attribute scores are combined into class-level scores. This two-step strategy is referred to as Direct Attribute Prediction (DAP) in Lampert et al. (2009).

We note that DAP suffers from several shortcomings. First, DAP proceeds in a two-step fashion, learning attribute-specific classifiers as a first step and combining attribute scores into class-level scores as a second step. Since attribute classifiers are learned independently of the end-task the overall strategy of DAP might be optimal at predicting attributes but not necessarily at predicting classes. Second, we would like an approach that can perform zero-shot prediction if no labeled samples are available for some classes, but that can also leverage new labeled samples for these classes as they become available. While DAP is straightforward to implement for zero-shot learning problems, it is not straightforward to extend to such an incremental learning scenario. Third, while attributes can be

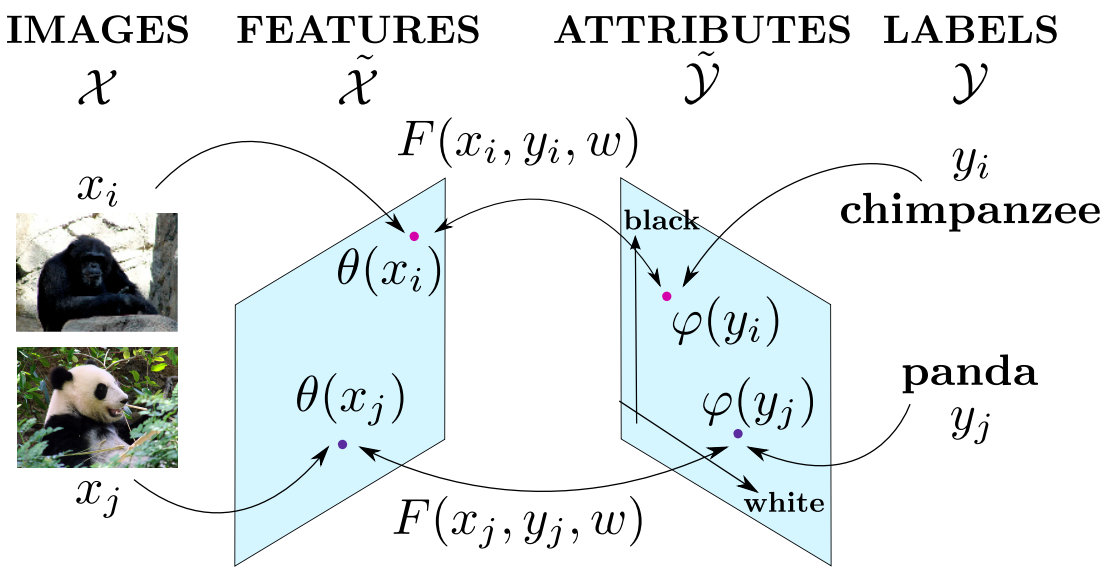


Figure 4.1 *Much work in computer vision has been devoted to image embedding (left): how to extract suitable features from an image? Here, the focus is on label embedding (right): how to embed class labels in a Euclidean space? Attributes can be used as as side information for the label embedding and measure the "compatibility"' between the embedded inputs and outputs with a function F.*

a useful source of prior information, they are expensive to obtain and the human labeling is not always reliable. Therefore, it is advantageous to seek complementary or alternative sources of side information. For instance, semantic hierarchies such as Wordnet[1] can bring valuable information and potentially boost predictive performance. Indeed, images of classes which are close in a semantic hierarchy are usually more similar than images of classes which are far Deselaers and Ferrari (2011). It is not straightforward to design an efficient way to incorporate these additional sources of information into DAP.

Various solutions have been proposed to address each of these problems separately. However, we do not know of any existing solution that addresses all of them in a principled manner. In this work, we propose such a solution by making use of the label embedding framework. We underline that, while there is an abundant literature in the computer vision community on image embedding (how to describe an image?) much less work has been devoted in comparison to label embedding in the $\mathcal{Y}$ space (how to describe a class?). We embed each class $y \in \mathcal{Y}$ in the space of attribute vectors and thus refer to our approach as *Attribute Label Embedding* (ALE). We use a structured output learning formalism and introduce a function which measures the compatibility between an image x and a label y (see Figure 4.1). The parameters of this function are learned on a training set of labeled samples to ensure that, given an image, the correct class(es) rank higher than the incorrect

[1] http://wordnet.princeton.edu/

ones. Given a test image, recognition consists in searching for the class with the highest compatibility.

ALE addresses in a principled fashion all three problems mentioned previously. First, we learn the model parameters and therefore optimize directly the class ranking, whereas DAP proceeds in two steps by solving intermediate problems. We show experimentally that ALE outperforms DAP in the zero-shot setting. Second, if available, labeled samples can be used to learn the embedding. Third, the label embedding framework is generic and not restricted to attributes: other sources of prior information can be combined with attributes or used as alternatives.

The rest of this chapter is organized as follows. In the next section, we review related work. In Section 4.3, we introduce ALE. In Section 4.4 we present experimental results on two public datasets: Animals with Attributes (AWA) Lampert et al. (2009) and Caltech-UCSD-Birds (CUB) Wah et al. (2011). Finally, we draw conclusions.

4.2 Related Work

The related work on zero-shot and few-shots learning is presented in Section 2.2.2. In this section we briefly review the related work that is most relevant to this chapter.

Attributes. Attributes have been used for image description Ferrari and Zisserman (2007); Farhadi et al. (2009); Chen et al. (2012), caption generation Kulkarni et al. (2011); Ordonez et al. (2011), face recognition Kumar et al. (2009); Scheirer et al. (2012); Chen et al. (2013), image retrieval Kumar et al. (2008); Siddiquie et al. (2011); Douze et al. (2011), action recognition Liu et al. (2011); Yao et al. (2011), novelty detection Wah and Belongie (2013) and object classification Lampert et al. (2009); Farhadi et al. (2009); Wang and Forsyth (2009); Wang and Mori (2010); Mahajan et al. (2011); Sharmanska et al. (2012); Mensink et al. (2012b). Since our task is object classification in images, we focus on the latter references.

The most popular approach to attribute-based recognition is the Direct Attribute Prediction (DAP) model which consists in predicting the presence of attributes in an image and combining the attribute prediction probabilities into class prediction probabilities Lampert et al. (2009). A significant limitation of DAP is the fact it assumes that attributes are independent from each other, an assumption which is generally incorrect (see our experiments on attribute correlation in section 4.4.1). Consequently, DAP has been improved to take into account the correlation between attributes or between attributes and classes Wang and Forsyth (2009); Wang and Mori (2010); Yu and Aloimonos (2010); Mahajan et al. (2011). However, all these models have limitations of their own. Wang and Forsyth (2009) assume that images are labeled with both classes and attributes. In our work we only assume

that classes are labeled with attributes, which requires significantly less hand-labeling of the data. Mahajan et al. (2011) use transductive learning and, therefore, assume that the test data is available as a batch, a strong assumption we do not make. The topic model of Yu and Aloimonos (2010) is only applicable to bag-of-visual-word image representations and, therefore, cannot leverage recent state-of-the-art image features such as the Fisher Vector Sánchez et al. (2013). Finally the latent SVM framework of Wang and Mori (2010) is not applicable to zero-shot learning, the focus of this chapter.

Several works have also considered the problem of discovering a vocabulary of attributes Berg et al. (2010b); Duan et al. (2012); Marchesotti and Perronnin (2013). Berg et al. (2010b) leverage text and images sampled from the Internet and use the mutual information principle to measure the information of a group of attributes. Duan et al. (2012) discovers local attributes and integrate humans in the loop for recommending the selection of attributes that are semantically meaningful. Marchesotti and Perronnin (2013) discover attributes from images, textual comments and ratings for the purpose of aesthetic image description. In our work, we assume that the class-attribute association matrix is provided. In this sense, this chapter is complementary to those.

Label embedding. In computer vision, a vast amount of work has been devoted to input embedding, *i.e.* how to represent an image. This includes works on patch encoding (see Chatfield et al. (2011) for a recent comparison), on kernel-based methods Shawe-Taylor and Cristianini (2004) with a recent focus on explicit embeddings Maji and Berg (2009); Vedaldi and Zisserman (2010), on dimensionality reduction Shawe-Taylor and Cristianini (2004) and on compression Jégou et al. (2011); Sánchez and Perronnin (2011); Vedaldi and Zisserman (2012). Comparatively, much less work has been devoted to label embedding.

Provided that the embedding function $\varphi : \mathcal{Y} \rightarrow \tilde{\mathcal{Y}}$ is chosen correctly, label embedding can be an effective way to share parameters between classes. Consequently, the main applications have been multiclass classification with many classes Amit et al. (2007); Weinberger and Chapelle (2008); Weston et al. (2010); Bengio et al. (2010) and zero-shot learning Larochelle et al. (2008); Palatucci et al. (2009). We now provide a taxonomy of embeddings. While this taxonomy is valid for both input $\theta : \mathcal{X} \rightarrow \tilde{\mathcal{X}} = \Re^D$ and output embeddings $\varphi : \mathcal{Y} \rightarrow \tilde{\mathcal{Y}}$, we focus here on output embeddings. They can be (i) fixed and data-independent, (ii) learned from data, or (iii) computed from side information.

- **Data-Independent Embeddings.** Kernel dependency estimation Weston et al. (2002) is an example of a strategy where φ is data-independent and defined implicitly through a kernel in the $\mathcal{Y}$ space. The compressed sensing approach of Hsu et al. (2009), is another example of data-independent embeddings where φ corresponds to random projections. The Error Correcting Output Codes (ECOC) framework encompasses a large family of embeddings that can be built using information-theoretic arguments Hamming (1950). ECOC approaches allow in particular to

tackle multi-class learning problems as described by Dietterich and Bakiri (1995). The reader can refer to Escalera et al. (2010) for a summary of ECOC methods and latest developments in the ternary output coding methods. Other data-independent embeddings are based on pairwise coupling and variants thereof such as generalized Bradley-Terry models Hastie et al. (2008).

- **Learned Embeddings.** A strategy consists in learning jointly θ and φ to embed the inputs and outputs in a common intermediate space $\mathcal{Z}$. The most popular example is Canonical Correlation Analysis (CCA) Hastie et al. (2008), which maximizes the correlation between inputs and outputs. Other strategies have been investigated which maximize directly classification accuracy, including the nuclear norm regularized learning of Amit et al. (2007) or the WSABIE algorithm of Weston et al. (2010).

- **Embeddings Derived From Side Information.** There are situations where side information is available. This setting is particularly relevant when little training data is available, as side information and the derived embeddings can compensate for the lack of data. Side information can be obtained at an image level Farhadi et al. (2009) or at a class level Lampert et al. (2009). We focus on the latter setting which is more practical as collecting side information at an image level is more costly. Side information may include "hand-drawn" descriptions Larochelle et al. (2008), text descriptions Farhadi et al. (2009); Lampert et al. (2009); Palatucci et al. (2009) or class taxonomies Weinberger and Chapelle (2008); Bengio et al. (2010).

While our focus is on embeddings derived from side information for zero-shot recognition, we also considered data independent embeddings and learned embeddings (using side information as a prior) for few-shots recognition.

4.3 Label Embedding with Attributes

Given a training set $\mathcal{S} = \{(x_n, y_n), n = 1 \dots N\}$ of input/output pairs with $x_n \in \mathcal{X}$ and $y_n \in \mathcal{Y}$ the goal of prediction is to learn a function $f : \mathcal{X} \to \mathcal{Y}$ by minimizing an empirical risk of the form

$$\frac{1}{N} \sum_{n=1}^{N} \Delta(y_n, f(x_n)) \tag{4.1}$$

where $\Delta : \mathcal{Y} \times \mathcal{Y} \to \Re$ measures the loss incurred from predicting $f(x)$ when the true label is y. In what follows, we focus on the 0/1 loss: $\Delta(y, z) = 0$ if $y = z$, 1 otherwise. In machine learning, a common strategy is to use embedding functions $\theta : \mathcal{X} \to \tilde{\mathcal{X}}$ and

$\varphi : \mathcal{Y} \rightarrow \tilde{\mathcal{Y}}$ for the inputs and outputs and then to learn on the transformed input/output pairs.

In this section, we first describe our model, *i.e.* our choice of f. We then explain how to leverage attributes to compute label embeddings. We also discuss how to learn the model parameters. Finally, we show that the label embedding framework is generic enough to accommodate for other sources of side information.

4.3.1 Framework

Figure 4.1 illustrates our model. Inspired from the structured prediction formulation Tsochantaridis et al. (2005), we introduce a compatibility function $F : \mathcal{X} \times \mathcal{Y} \rightarrow \Re$ and define f as follows:

$$f(x; w) = \arg\max_{y \in \mathcal{Y}} F(x, y; w) \tag{4.2}$$

where w denotes the model parameter vector of F and $F(x, y; w)$ measures how compatible is the pair (x, y) given w. It is generally assumed that F is linear in some combined feature embedding of inputs/outputs $\psi(x, y)$:

$$F(x, y; w) = w'\psi(x, y) \tag{4.3}$$

and that the joint embedding ψ can be written as the tensor product between the image embedding $\theta : \mathcal{X} \rightarrow \tilde{\mathcal{X}} = \Re^D$ and the label embedding $\varphi : \mathcal{Y} \rightarrow \tilde{\mathcal{Y}} = \Re^E$:

$$\psi(x, y) = \theta(x) \otimes \varphi(y) \tag{4.4}$$

and $\psi(x, y) : \Re^D \times \Re^E \rightarrow \Re^{DE}$. In this case w is a DE-dimensional vector which can be reshaped into a $D \times E$ matrix W. Consequently, we can rewrite $F(x, y; w)$ as a bilinear form:

$$F(x, y; W) = \theta(x)'W\varphi(y). \tag{4.5}$$

Other compatibility functions could have been considered. For example, the function:

$$F(x, y; W) = -\|\theta(x)'W - \varphi(y)\|^2 \tag{4.6}$$

is typically used in regression problems.

Also, if D and E are large, it might be advantageous to consider a low-rank decomposition $W = U'V$ to reduce the number of parameters. In such a case, we have:

$$F(x, y; U, V) = (U\theta(x))' (V\varphi(y)) . \tag{4.7}$$

The CCA Hastie et al. (2008) or the WSABIE Weston et al. (2010) methods rely, for example, on such a decomposition.

Attribute Label Embedding We now consider the problem of computing label embeddings $\varphi^{\mathcal{A}}$ from attributes which we refer to as Attribute Label Embedding (ALE). We assume that we have C classes, *i.e.* $\mathcal{Y} = \{1, \ldots, C\}$ and that we have a set of E attributes $\mathcal{A} = \{a_i, i = 1 \ldots E\}$ to describe the classes. We also assume that we are provided with an association measure $\rho_{y,i}$ between each attribute a_i and each class y. These associations may be binary or real-valued if we have information about the association strength (*e.g.* if the association value is obtained by averaging votes). We embed class y in the E-dim attribute space as follows:

$$\varphi^{\mathcal{A}}(y) = [\rho_{y,1}, \ldots, \rho_{y,E}] \tag{4.8}$$

and denote $\Phi^{\mathcal{A}}$ the $E \times C$ matrix of attribute embeddings which stacks the individual $\varphi^{\mathcal{A}}(y)$'s.

We note that in equation (4.5) the image and label embeddings play symmetric roles. In the same way it makes sense to normalize samples when they are used as input to large-margin classifiers, it can make sense to normalize the output vectors $\varphi^{\mathcal{A}}(y)$. In section 4.4.1 we compare (i) continuous embeddings, (ii) binary embeddings using $\{0, 1\}$ for the encoding and (iii) binary embeddings using $\{-1, +1\}$ for the encoding. We also explore two normalization strategies: (i) mean-centering (*i.e.* compute the mean over all learning classes and subtract it) and (ii) ℓ_2-normalization. We underline that such encoding and normalization choices are not arbitrary but relate to prior assumptions we might have on the problem. For instance, underlying the $\{0, 1\}$ embedding is the assumption that the presence of the same attribute in two classes should contribute to their similarity, but not its absence. Here we assume a dot-product similarity between attribute embeddings which is consistent with our linear compatibility function (4.5). Underlying the $\{-1, 1\}$ embedding is the assumption that the presence or the absence of the same attribute in two classes should contribute equally to their similarity. As for mean-centered attributes, they take into account the fact that some attributes are more frequent than others. For instance, if an attribute appears in almost all classes, then in the mean-centered embedding, its absence will contribute more to the similarity than its presence. This is similar to an IDF effect in TF-IDF encoding. As for the ℓ_2-normalization, it enforces that each class is closest to itself according to the dot-product similarity, a reasonable assumption.

Also, in the case where attributes are redundant, it might be advantageous to decorrelate them. In such a case, we make use of the compatibility function (4.7). The matrix V may be learned from labeled data jointly with U. As a simpler alternative, it is possible to first learn the decorrelation, such as by performing a Singular Value Decomposition (SVD) on the $\Phi^{\mathcal{A}}$ matrix, and then learning the U. We will study the effect of attribute decorrelation in our experiments.

4.3.2 Parameter learning

We now turn to the estimation of the model parameters W from a labeled training set $\mathcal{S}$. The simplest learning strategy is to maximize directly the compatibility between the input and output embeddings:

$$\frac{1}{N}\sum_{n=1}^{N} F(x_n, y_n; W) \tag{4.9}$$

with potentially some constraints and regularization on W. This is exactly the strategy adopted in regression Palatucci et al. (2009); Socher et al. (2013). However, such an objective function does not optimize directly our end-goal which is image classification. Therefore, we draw inspiration from the WSABIE algorithm that uses the WAR Weston et al. (2010) objective function introduced in Section 3.3. WSABIE learns jointly image and label embeddings from data to optimize classification accuracy. The crucial difference between WSABIE and ALE is the fact that the latter uses attributes as side information. Note that the proposed ALE is not tied to WSABIE and that we report results in Section 4.4.1 with other objective functions including regression and SSVM. We chose to focus on the WSABIE objective function with ALE because it yields good results and is scalable.

In what follows, we briefly review the WSABIE objective function Weston et al. (2010) and then explain how we adapt it to (i) zero-shot learning with side information and (ii) learning with few (or more) examples with side information. We then mention the optimization of our objective functions. In what follows, Φ is the matrix which stacks the embeddings $\varphi(y)$.

WSABIE. Let $\mathbb{1}(u) = 1$ if u is true and 0 otherwise. Let:

$$\ell(x_n, y_n, y) = \Delta(y_n, y) + \theta(x)'W[\varphi(y) - \varphi(y_n)] \tag{4.10}$$

Let $r(x_n, y_n)$ be the rank of label y_n for image x_n. Finally, let $\alpha_1 \geq \alpha_2 \geq \ldots \geq \alpha_C \geq 0$ be a sequence of C coefficients and let $\beta_k = \sum_{j=1}^{k} \alpha_j$. Usunier et al. (2009) propose to use the following ranking loss for $\mathcal{S}$:

$$\frac{1}{N}\sum_{n=1}^{N} \beta_{r(x_n, y_n)}. \tag{4.11}$$

Maximizing the equation (4.11) enforces correct labels to rank higher than incorrect ones. The penalty incurred by going from rank k to $k+1$ is α_k. Hence, a decreasing sequence $\{\alpha_j\}_{j\geq 1}$ implies that a mistake on the rank when the true rank is at the top of the list incurs a higher loss than a mistake on the rank when the true rank is lower in the list.

Algorithm 1 SGD optimization for ALE

Intitialize $W^{(0)}$ randomly.

for $t = 1$ to T **do**
  Draw (x,y) from $\mathcal{S}$.
  for $k = 1, 2, \ldots, C-1$ **do**
    Draw $\bar{y} \neq y$ from $\mathcal{Y}$
    if $\ell(x, y, \bar{y}) > 0$ **then**
      a) Update W

$$W^{(t)} = W^{(t-1)} + \eta_t \beta_{\lfloor \frac{C-1}{k} \rfloor} \theta(x)[\varphi(y) - \varphi(\bar{y})]' \tag{4.14}$$

      b) Update Φ **(not applicable to zero-shot)**

$$\varphi^{(t)}(y) = (1 - \eta_t \mu)\varphi^{(t-1)}(y) + \eta_t \mu \varphi^{\mathcal{A}}(y) + \eta_t \beta_{\lfloor \frac{C-1}{k} \rfloor} W_t' \theta(x) \tag{4.15}$$

$$\varphi^{(t)}(\bar{y}) = (1 - \eta_t \mu)\varphi^{(t-1)}(\bar{y}) + \eta_t \mu \varphi^{\mathcal{A}}(\bar{y}) - \eta_t \beta_{\lfloor \frac{C-1}{k} \rfloor} W_t' \theta(x) \tag{4.16}$$

      break
    end if
  end for
end for

Instead of optimizing an upper-bound on equation(4.11), Weston et al. (2010) propose to optimize the following approximation of objective (4.11):

$$R(\mathcal{S}; W, \Phi) = \frac{1}{N} \sum_{n=1}^{N} \frac{\beta_{r_\Delta(x_n, y_n)}}{r_{\Delta(x_n, y_n)}} \sum_{y \in \mathcal{Y}} \max\{0, \ell(x_n, y_n, y)\} \tag{4.12}$$

where an upper-bound on the rank of label y_n for image x_n is:

$$r_\Delta(x_n, y_n) = \sum_{y \in \mathcal{Y}} \mathbb{1}(\ell(x_n, y_n, y) > 0). \tag{4.13}$$

The main advantage of the WSABIE formulation denoted in the equation (4.12) is that it can be optimized efficiently through Stochastic Gradient Descent (SGD). Following Weston et al. (2010), we choose $\alpha_j = 1/j$. In WSABIE, the label embedding space dimensionality is a parameter to tune.

Zero-Shot Objective. We adapt the WSABIE objective to zero-shot learning. In such a case, we cannot learn Φ from labeled data (contrary to WSABIE) but rely on side information. Therefore, the matrix Φ is fixed and set to $\Phi^{\mathcal{A}}$. We only optimize the objective

(4.12) with respect to W. We note that, when Φ is fixed and only W is learned, the objective (4.12) is closely related to the (unregularized) structured SVM (SSVM) objective of Tsochantaridis et al. (2005):

$$\frac{1}{N}\sum_{n=1}^{N}\max_{y\in\mathcal{Y}}\ell(x_n, y_n, y) \tag{4.17}$$

The main difference is the loss function, which is the multi-class loss function for SSVM. The multi-class loss function focuses on the score with the highest rank while WSABIE takes all scores into account in a weighted fashion. A major advantage of WSABIE is its scalability to large datasets Weston et al. (2010); Perronnin et al. (2012).

Few-Shots Objective. We now adapt the WSABIE objective to the case where we have labeled data and side information. In such a case, we want to learn the class embeddings using as prior information $\Phi^{\mathcal{A}}$. We therefore add to the objective (4.12) a regularizer:

$$R(\mathcal{S}; W, \Phi) + \frac{\mu}{2}||\Phi - \Phi^{\mathcal{A}}||^2 \tag{4.18}$$

and optimize jointly with respect to W and Φ. Note that the previous equation is somewhat reminiscent of the ranking model adaptation of Geng et al. (2012).

Optimization. As for the optimization, both in the zero-shot and few-shots learning, we follow Weston et al. (2010) and use Stochastic Gradient Descent (SGD). Training with SGD consists at each step t in (i) choosing a sample (x, y) at random, (ii) sampling classes $\bar{y} \neq y$ until a violating class is found, *i.e.* until $\ell(x, y, \bar{y}) > 0$, and (iii) updating the projection matrix (and the class embeddings in case of few-shots learning) using a sample-wise estimate of the regularized risk. Following Weston et al. (2010); Perronnin et al. (2012), we use a constant step size $\eta_t = \eta$. The detailed SGD procedure for the ALE method is provided in Algorithm 1.

4.3.3 Beyond attributes

While attributes make sense in the label embedding framework, we note that label embedding is more general and can accommodate other sources of side information. The canonical example is that of structured learning with a taxonomy of classes (see "classification with taxonomies" in Section 4.2 in Tsochantaridis et al. (2005)). Assuming that classes are organized in a tree structure, meaning that we have an ordering operation $\prec$

in $\mathcal{Y}$, we can define $\xi_{y,z} = 1$ if $z \prec y$ or $z = y$. The hierarchy embedding $\varphi^{\mathcal{H}}(y)$ can be defined as the C dimensional vector:

$$\varphi^{\mathcal{H}}(y) = [\xi_{y,1}, \ldots, \xi_{y,C}]. \tag{4.19}$$

We later refer to this embedding as Hierarchy Label Embedding (HLE) and we compare $\varphi^{\mathcal{A}}$ and $\varphi^{\mathcal{H}}$ as sources of prior information in our experiments. In the case where classes are not organized in a tree structure but form a graph, then other types of embeddings could be used, for instance by performing a kernel PCA on the commute time kernel Saerens et al. (2004).

Different embeddings can be easily combined in the label embedding framework, *e.g.* through simple concatenation of the different embeddings or through more complex operations such as a CCA of the embeddings. This is to be contrasted with DAP which cannot accommodate so easily other sources of prior information.

4.4 Experiments

In this section, we first detail the experimental setting, namely the the datasets, the input embeddings (*i.e.* image features) and output embeddings that were used in our experiments. Following, we present the zero-shot recognition experiments where training and test classes are disjoint. The experimental evaluation goes beyond zero-shot learning and considers the case where we have plenty of training data for some background classes and little training data for the classes of interest. Finally, we report results in the case where we have equal amounts of training data for all classes.

Datasets. The results have been reported in this work on two public datasets. The Animals With Attributes (AWA) Lampert et al. (2009) contains roughly 30,000 images of 50 animal classes. The CUB-200-2011 (CUB) Wah et al. (2011) contains roughly 11,800 images of 200 bird classes. For zero-shot learning using AWA dataset, the training and test classes are determined by Lampert et al. (2009). From CUB, as there is not such a partitioning determined by default, we randomly select 150 classes for training and the remaining 50 classes for testing in zero-shot learning experiments. For many-shots learning experiments we divided the dataset in two parts with training and test partitions has the same number of images. For each experiments, the training is done on the 2/3 of the training data and cross validation is done on the remaining 1/3 of the training data. The specific partitioning of the datasets will be explained more detailed in its respective subsection. The results are in terms of top-1 accuracy (in %) that are averaged over the classes.

Input Embeddings (Image Features). Images are resized to 100K pixels if larger while keeping the aspect ratio. The local image features are 128-dim SIFT features Lowe (2004) and 96-dim color features Clinchant et al. (2007) extracted from regular grids at multiple scales. Both of them are reduced to 64-dim using PCA. These features are then aggregated into an image descriptors using the Fisher Vector (FV) Perronnin et al. (2010b) which was shown to be a state-of-the-art patch encoding technique in Chatfield et al. (2011). Using Gaussian Mixture Models with 16 or 256 Gaussians, one SIFT FV and one color FV has been computed per image and they are concatenated into either 4,096 (4K) or 65,536-dim (64K) FVs. As opposed to Akata et al. (2013a), the PQ-compression has not been applied which leads to better results as reported in this chapter.

Output Embeddings. In AWA, each class was annotated with 85 attributes by 10 students Osherson et al. (1991). Continuous class-attribute associations were obtained by averaging the per-student votes and subsequently thresholded to obtain binary attributes. In CUB, 312 attributes were obtained from a bird field guide. Each image was annotated according to the presence/absence of these attributes. The per-image attributes were averaged to obtain continuous-valued class-attribute associations and thresholded with respect to the overall mean to obtain binary attributes. By default, we use continuous attribute embeddings in our experiments.

As an alternative to attributes, we use the Wordnet hierarchy as a source of prior information to compute output embeddings. We refer to such an approach as Hierachical Label Embedding (HLE). We compute the binary output codes as follows: we use a $\{0, 1\}$ encoding according the absence and presence of a given node among the ancestors of a certain class.

The embedding can be learned from data when a sufficient amount of labeled training data is available for all classes, for instance using the WSABIE algorithm Weston et al. (2010). Another approach is to use "data-independent embeddings", leveraging recent ideas from compressed sensing DeVore (2007). In summary, we compared the following approaches for data-driven embeddings:

- Web Scale Annotation By Image Embedding (WSABIE): The objective function of WSABIE is provided in equation (4.12) and the corresponding optimization algorithm is similar to the one of ALE described in Algorithm 1. The difference is that WSABIE does not use any prior information and therefore that the regularization value μ is set to 0 in equation (4.15 and 4.16). Another difference with ALE is that the embedding dimensionality E is a parameter of WSABIE which is obtained through cross-validation. This is an advantage of WSABIE since this provides an additional free parameter with respect to ALE. On the other hand, the cross-validation procedure is an expensive one.

- One-Vs-Rest embedding (OVR): the embedding dimensionality is C where C is the number of classes and the matrix Φ is the $C \times C$ identity matrix. This is equivalent to training independently one classifier per class.

and the approaches below for data-independent embeddings:

- Gaussian Label Embeddings (GLE): The class embeddings are drawn from a standard normal distribution, similar fo random projections in compressed sensing DeVore (2007). Similarly to WSABIE, the label embedding dimensionality E is a parameter of GLE which needs to be cross-validated. For GLE, since the embedding is randomly drawn, we repeat the experiments 10 times and report the average (as well as the standard deviation where relevant).
- Hadamard Label Embedding: An Hadamard matrix is a square matrix whose rows/columns are mutually orthogonal and whose entries are $\{-1, 1\}$ DeVore (2007). Hadamard matrices can be computed iteratively with $H_1 = 1$ and $H_{2^k} = [H_{2^{k-1}} H_{2^{k-1}}; H_{2^{k-1}} - H_{2^{k-1}}]$. Note that in practice, we never obtained improved results with Hadamard embedding over GLE.

Note that WSABIE, GLE and OVR are not applicable to zero-shot learning.

4.4.1 Zero-Shot Learning

In this section, we evaluate the proposed ALE in the zero-shot experimental setting. For AWA, we use the standard zero-shot setup which consists in learning parameters on 40 of the classes and evaluating accuracy on the 10 remaining classes. In these experiments, we use the all the images in 40 learning classes ($\approx 24,700$ images) to learn and cross-validate the model parameters. We then use all the images in 10 evaluation classes ($\approx 6,200$ images) to measure the accuracy. For CUB, we use 150 classes for learning ($\approx 8,900$ images) and 50 for evaluation ($\approx 2,900$ images).

Direct Attribute Prediction (DAP) Baseline. We first provide the baseline for DAP on AWA and CUB. In DAP, an image x is assigned to the class y with the highest posterior probability using the following formula:

$$p(y|x) \propto \prod_{e=1}^{E} p(a_e = \rho_{y,e}|x) \tag{4.20}$$

where $\rho_{y,e}$ is the association measure between attribute a_e and class y, and $p(a_e = 1|x)$ is the probability that image x contains the attribute e. We train for each attribute one linear

classifier on the FVs. We use a (regularized) logistic loss which provides an attribute classification accuracy similar to the SVM but with the added benefit that its output is already a probability.

	Object prediction	
	DAP	ALE
AWA	41.0	**48.5**
CUB	12.3	**26.9**

Table 4.1 *Comparison of the DAP baseline Lampert et al. (2009) with the proposed Attribute Label Embedding (ALE) approach.The object classification accuracy (top-1 in %) on the 10 AWA and 50 CUB evaluation classes.*

The results are provided in Table 4.1. With the 64K-dim features, our DAP baseline is 41.0% on AWA and 12.3% on CUB. Our DAP AWA results are comparable to the 40.5% reported by Lampert, but with different features. The advantage of using FVs is that they work well with linear classifiers while the Lampert features work best with costly non-linear kernel classifiers (χ^2 SVMs). The efficient linear classifiers enable us to run more experiments more efficiently.

Comparison of Output Encodings. We now compare three different output encodings: (i) continuous encoding, *i.e.* the class-attribute associations are in the form of confidence level of each attribute, (ii) binary $\{0, 1\}$ encoding, *i.e.* the class-attribute associations are in the form of presence and absence of each attribute and (iii) binary $\{-1, +1\}$ encoding, *i.e.* the embeddings have a constant norm. We also compare two normalizations: (i) mean-centering of the output embeddings, *i.e.* the mean of the output encoding vector has been subtracted from the vector itself and (ii) ℓ_2-normalization, *i.e.* the output encoding vector has been ℓ_2-normalized.

The experimental results in Table 4.2 show that significantly better results are obtained with continuous embeddings than with thresholded binary embeddings. This is expected since continuous embeddings encode the strength of association between a class and an attribute and therefore they carry more information. We believe that this is a major strength of the proposed ALE approach as other algorithms such as DAP cannot accommodate such soft values in a straightforward manner. Mean-centering seems to have little impact and ℓ_2-normalization makes a significant difference except in the case of the $\{-1, +1\}$ encoding since the embeddings already have a constant norm in this case. In what follows, we focus our experiments on the continuous ℓ_2-normalized embeddings.

Comparison of Learning Algorithms. We now compare three objective functions to learn the mapping between inputs and outputs. The first one is Ridge Regression (RR)

		AWA					
		FV=4K			FV=64K		
μ	ℓ_2	cont	$\{0,1\}$	$\{-1,+1\}$	cont	$\{0,1\}$	$\{-1,+1\}$
no	no	41.5	34.2	32.5	44.9	42.4	41.8
yes	no	42.2	33.8	33.8	44.9	42.4	42.4
no	yes	**45.7**	34.2	34.8	**48.5**	44.6	41.8
yes	yes	44.2	34.9	34.9	47.7	44.8	44.8
		CUB					
		FV=4K			FV=64K		
μ	ℓ_2	cont	$\{0,1\}$	$\{-1,+1\}$	cont	$\{0,1\}$	$\{-1,+1\}$
no	no	17.2	10.4	12.8	22.7	20.5	19.6
yes	no	16.4	10.4	10.4	21.8	20.5	20.5
no	yes	**20.7**	15.4	15.2	**26.9**	22.3	19.6
yes	yes	20.0	15.6	15.6	26.3	22.8	22.8

Table 4.2 *Comparison of the continuous non-binarized embedding (cont), the binary $\{0,1\}$ embedding and the binary $\{+1,-1\}$ embedding. The impact of mean-centering (μ) and ℓ_2-normalization has also been studied here. The experiments are performed on AWA and CUB with FV=4K and FV=64K. The output embedding dimensionality is always 85 for AWA and 312 for CUB.*

which was used in Palatucci et al. (2009) to map input features to output attribute labels. In a nutshell, RR consists in optimizing a regularized quadratic loss for which there exists a closed form formula. The second one is the standard structured SVM (SSVM) multiclass objective function of Tsochantaridis et al. (2005). The third one is the ranking objective of WSABIE Weston et al. (2010) which is described in detail Section 4.3.The results are provided in Table 4.3.

	RR	multi	rank
AWA	44.5	47.9	**48.5**
CUB	21.6	**26.3**	**26.3**

Table 4.3 *Comparison of different learning algorithms for ALE: ridge-regression (denoted as RR) with quadratic loss in Palatucci et al. (2009), multi-class SSVM (denoted as multi) which is also known as structured SVM of Tsochantaridis et al. (2005) and ranking based on WSABIE of Weston et al. (2010) (denoted as rank)*

The conclusion is that the multiclass and ranking frameworks are on-par and outperform the simple ridge regression. This is not surprising since the two former objective functions are more closely related to our end goal which is classification. In what follows, we

always use the ranking framework to learn the parameters of our model since it was shown to be more scalable Weston et al. (2010); Perronnin et al. (2012).

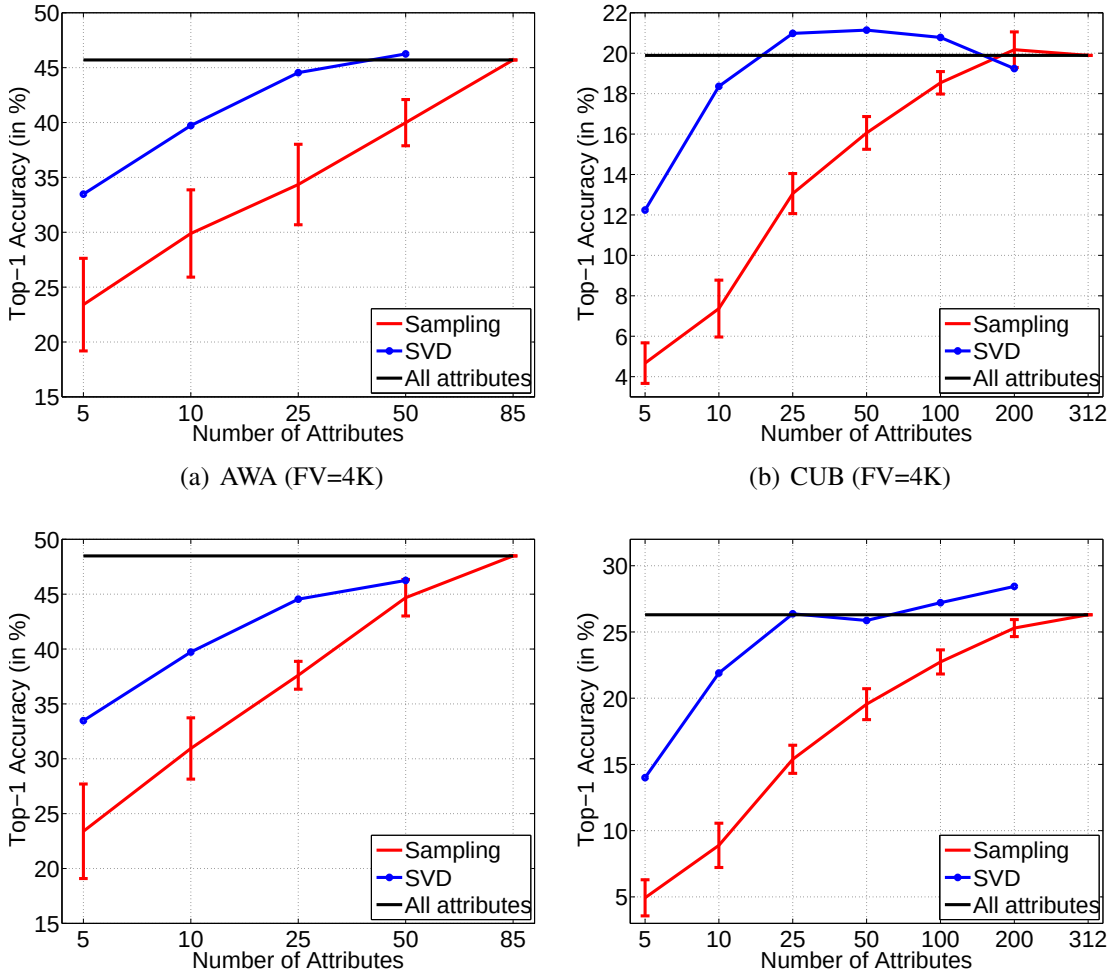
Attribute Correlation. While correlation in the input space is a well-studied topic, comparatively little work has been done to measure the correlation in the output space. Here, we reduce the output space dimensionality and study the impact on the classification accuracy. It is worth noting that reducing the output dimensionality leads to significant speed-ups at training and test times. We explore two different techniques: Singular Value Decomposition (SVD) and attribute sampling. We learn the SVD on AWA (resp. CUB) on the 50×85 (resp. 200×312) $\Phi^{\mathcal{A}}$ matrix. For the sampling, we sub-sample a fixed number of attributes and repeat the experiments 10 times for 10 different random sub-samplings. The results of these experiments can be seen on Figure 4.2.

The experimental results on Figure 4.2 show that there is a significant amount of correlation between attributes. For instance, on AWA (Figure 4.2(c)) the accuracy drops from 48.5% to approximately 45% when reducing from an 85-dim space to a 25-dim space. More impressively, on CUB (Figure 4.2(d)), with 25 dimensions the accuracy is on par with the 312-dim embedding. SVD outperforms random sampling of the attribute dimensions although there is no guarantee that SVD will select the most informative dimensions (see for instance the small pit in performance on CUB at 50 dimensions). In random sampling of output embeddings, the choice of the attributes seems to be an important factor that affects the descriptive power of output embeddings. Consequently, the variance is higher when a small number of attributes is selected.

Comparison of ALE and HLE. As mentioned earlier, while attributes can be a useful source of prior information to embed classes, other sources of side information exist. We consider as an alternative the Wordnet hierarchy. We collect the set of ancestors of the 50 AWA (resp. 200 CUB) classes from Wordnet and build a hierarchy with 150 (resp. 299) nodes[2]. We used the $\{0, 1\}$ embedding with ℓ_2-norm. We also consider the combination of attributes and hierarchies. We explore two simple alternatives: the concatenation of the embeddings (AHLE early) and the late fusion of classification scores (AHLE late) calculated by averaging the scores obtained using ALE and HLE separately.

Results are provided in Table 4.4. On both AWA and CUB, ALE outperforms HLE showing that continuous attributes have the potential to bring significantly more prior information than class hierarchies. Note that in Akata et al. (2013a), we reported better results on AWA with HLE compared to ALE. The main difference with the current experiment is that we use continuous attribute encodings while Akata et al. (2013a) was using a binary encoding. The late fusion of ALE and HLE is always superior to the early fusion and improves slightly over ALE alone.

[2]In some cases, some of the nodes have a single child. We did not clean the automatically obtained hierarchy.



(a) AWA (FV=4K) (b) CUB (FV=4K)

(c) AWA (FV=64K) (d) CUB (FV=64K)

Figure 4.2 *Classification accuracy on AWA and CUB as a function of the label embedding dimensionality. We compare the baseline which uses all attributes, with an SVD dimensionality reduction and a sampling of attributes (we report the mean and standard deviation over 10 samplings).*

	ALE	HLE	AHLE early	AHLE late
AWA	48.5	40.4	46.8	**49.4**
CUB	26.9	18.5	27.1	**27.3**

Table 4.4 *Comparison of attributes (ALE) and hierarchies (HLE) for label embedding. We consider their combination by simple concatenation (AHLE early) or by the averaging of the scores (AHLE late).*

Attribute Interpretability In ALE, each column of W can be interpreted as an attribute classifier and $\theta(x)'W$ as a vector of attribute scores of x. However, one major difference with DAP is that we do not optimize for attribute classification accuracy. This might be viewed as a disadvantage of our approach as we might loose interpretability, an important property of attribute-based systems when, for instance, one wants to include a human in the loop Branson et al. (2010); Wah et al. (2011). We therefore measured the attribute prediction accuracy of DAP and ALE. For each attribute, following Lampert et al. (2009), we measure the AUC on the set of the evaluation classes and report the mean.

	Attribute prediction	
	DAP	ALE
AWA	**72.7**	**72.7**
CUB	**64.8**	59.4

Table 4.5 *Comparison of the DAP baseline Lampert et al. (2009) with the proposed Attribute Label Embedding (ALE) approach. The attribute prediction accuracy (AUC in %) on the 85 AWA and 312 CUB attributes.*

Attribute prediction scores are shown in Table 4.5 (right columns). The attribute prediction accuracy of DAP is at least as high as that of ALE. This is expected since DAP optimizes directly attribute-classification accuracy. However, the AUC for ALE is still reasonable, especially on AWA (the performance is on par). Thus, our learned attribute classifiers should still be interpretable. We provide qualitative results on AWA in Figure 4.3: we show the four highest ranked images for some of the attributes with the highest AUC scores (namely >90%) and lowest AUC scores (namely <50%).

(a) is quadrapedal (b) lives in ocean (c) lives on the ground

(d) is weak (e) lives in plains (f) hibernates

Figure 4.3 *Sample attributes recognized with high (> 90%) accuracy (top) and low (*i.e. *<50%) accuracy (bottom) by ALE on AWA. For each attribute we show the images ranked highest. Note that a AUC < 50% means that the prediction is worse than random on average.*

Some qualitative results on AWA can be seen on Figure 4.3. The four highest ranked images for certain classes with the highest AUC scores (namely >90%) are shown on

the top row of Figure 4.3 and the classes with the lowest AUC scores (namely <50%) are shown on the bottom row of Figure 4.3. Although for some attributes ALE performs worse than random, the highest ranked images still are descriptive about the attributes they represent.

Comparison with The State-of-the-Art We can compare our results to those published in the literature on AWA since we are using the standard training/testing protocol (again, there is no such zero-shot protocol on CUB). To the best of our knowledge, the best zero-shot recognition results on AWA are those of Yu et al. (2013) who report 48.3% accuracy. We report 48.5% with ALE and 49.4% with AHLE (late fusion of ALE and HLE). Note that we use different features and that care should be taken when comparing these numbers.

4.4.2 Few-Shots Learning

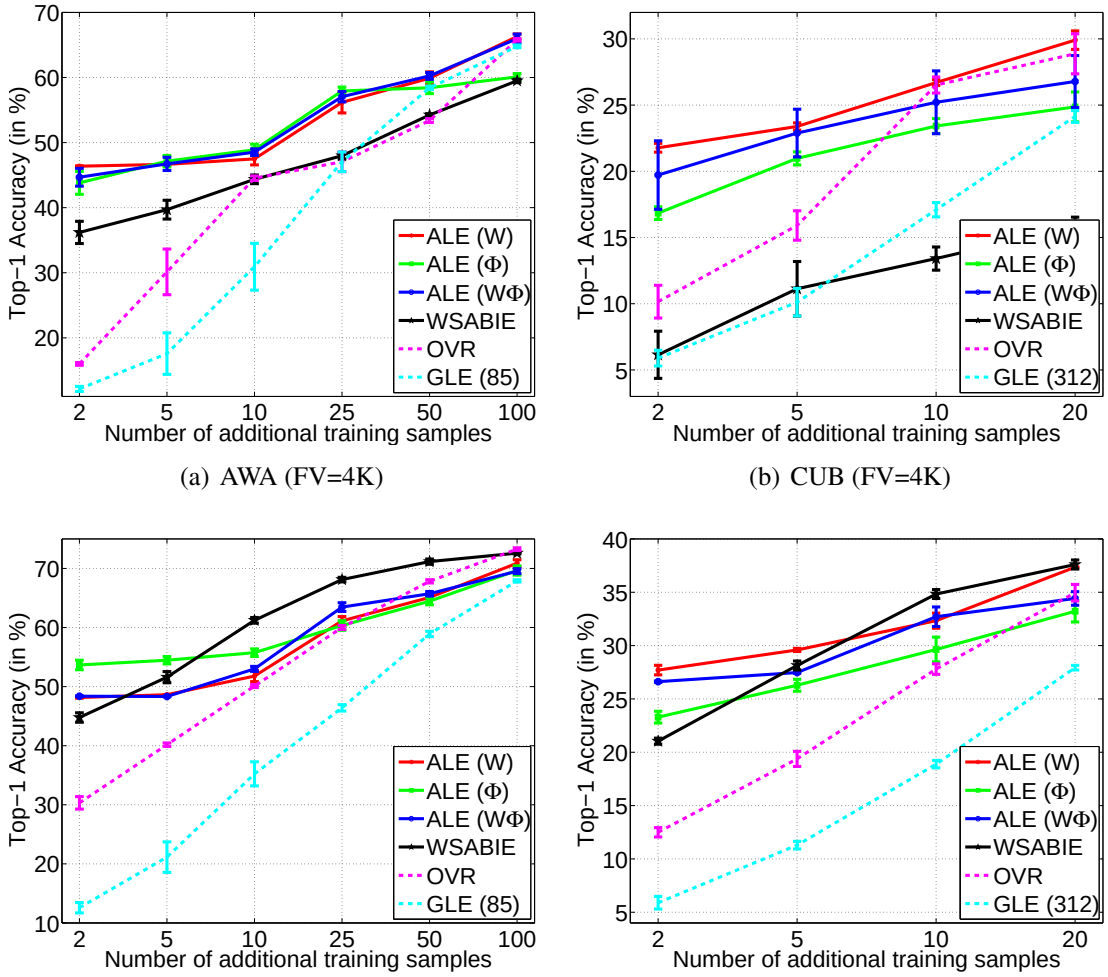
In these experiments, we assume that we have few (*e.g.* 2, 5, 10, etc.) training samples for a set of classes of interest (the 10 AWA and 50 CUB evaluation classes) in addition to all the samples from a set of "background classes" (the remaining 40 AWA and 150 CUB classes). For each evaluation class, we reserve approximately half of the data for training and cross-validation purposes (the 2, 5, 10, etc. training samples are drawn from this pool) and half of the data for test purposes. For AWA, the minimum number of training samples in the evaluation classes is 302 (151 training and 151 test images). Hence, we restricted the number of training samples for the evaluation classes to 100. For CUB, the minimum number of training samples in the evaluation classes is 42 (21 training and 21 test images). Hence, we restricted the number of training samples for the evaluation classes to 20.

Algorithms. We compare the proposed ALE with three baselines: OVR, GLE and WSABIE. We are especially interested in analyzing the following factors: (i) the influence of parameter sharing (GLE, WSABIE and ALE) *vs*. no parameter sharing (OVR), (ii) the influence of learning the embedding (WSABIE) *vs*. having a fixed embedding (GLE and ALE) and (iii) the influence of prior information (ALE) *vs*. no prior information (OVR, GLE and WSABIE).

For ALE and WSABIE, W is intialized to the matrix learned in the zero-shot experiments. For ALE, we experimented with three different learning variations:

- ALE(W) consists in learning the parameters W and keeping the embedding fixed ($\Phi = \Phi^{\mathcal{A}}$).

- ALE(Φ) consists in learning the embedding parameters Φ and keeping W fixed.
- ALE($W\Phi$) consists in learning both W and Φ.



(a) AWA (FV=4K) (b) CUB (FV=4K)

(c) AWA (FV=64K) (d) CUB (FV=64K)

Figure 4.4 *Classification accuracy on AWA and CUB as a function of the number of training samples per class. Additional to all the images from training classes in zero-shot learning, few additional images from the test classes have been used. Reported results are 10 way in AWA and 50 way in CUB. In ALE(Φ), only the matrix Φ is updated, in ALE(W), only the matrix W is updated and in ALE($W\Phi$), both W and Φ are updated. ALE has also been compared with WSABIE, GLE (with 85 dim in AWA and 312 dim in CUB) and simple One-vs-Rest (OVR) techniques.*

Results. We show results in Figure 4.4 for AWA and CUB using 4K-dim and 64K-dim features. We can draw the following conclusions. First, GLE underperforms all other

approaches for limited training data which shows that random embeddings are not appropriate in this setting. Second, in general, WSABIE and ALE outperform OVR and GLE for small training sets (*e.g.* for less than 10 training samples) which shows that learned embeddings (WSABIE) or embeddings based on prior information (ALE) can be effective when training data is scarce. The only exception is WSABIE on CUB for the small 4K-dim FVs. Third, for tiny amounts of training data (*e.g.* 2-5 training samples per class), ALE outperforms WSABIE which shows the importance of prior information in this setting. Fourth, all variations of ALE – ALE(W), ALE(Φ) and ALE($W\Phi$) – perform somewhat similarly. We note however that the ALE(W) variant seems to have a slight edge over the the other ones showing that embeddings need not necessarily be learned when the prior is informative. Fifth, as the number of training samples increases, all algorithms seem to converge to a similar accuracy which shows that, as expected, parameter sharing and prior information are less crucial when training data is plentiful.

4.4.3 All Data Experiments

In these experiments, we learn and test the classifiers on the 50 AWA (resp. 200 CUB) classes. For each class, we reserve approximately half of the data for training and cross-validation purposes (the 2, 5, 10, etc. training samples are drawn from this pool) and half of the data for test purposes. On CUB, we use the standard training/test partition provided with the dataset.

Comparison of Discrete and Continuous Attributes. As was the case for the zero-shot experiments, we first compare different encoding techniques (continuous embedding *vs.* binary embedding) and normalization strategies (with/without mean centering and with/without ℓ_2-normalization). The results are provided in Table 4.6.

As was the case for zero-shot learning, we observe that mean-centering does not have a positive effect in general and ℓ_2-normalization consistently improves performance. However, a major difference with the zero-shot case is that continuous embeddings seem to perform comparable to binary embeddings. This seems to indicate that the quality of the prior information used in label embedding has little impact when training data is plentiful.

Comparison of Learning Algorithms We now compare on the whole training sets several learning algorithms: OVR, GLE, WSABIE, ALE (we use the ALE(W) variant where the embedding parameters are kept fixed), HLE, AHLE (with early and late fusion).

As it can be seen from Table 4.7, the OVR baseline performs on par with ALE on AWA and better than ALE on CUB with 64K FV. Since the same behavior is not observed on FV with 4K dimensions, it is arguable that when the dimensionality of the data is high

		AWA				CUB			
		FV=4K		FV=64K		FV=4K		FV=64K	
ℓ_2	mc	$\{0,1\}$	cont	$\{0,1\}$	cont	$\{0,1\}$	cont	$\{0,1\}$	cont
no	no	42.3	41.6	45.3	46.2	13.0	13.9	16.5	16.7
yes	no	44.3	44.6	52.5	**53.3**	16.2	**17.5**	21.4	**21.6**
no	yes	42.2	41.6	45.8	46.2	13.2	13.9	16.5	16.7
yes	no	**44.8**	44.5	51.3	52.0	16.1	17.3	17.3	**21.6**

Table 4.6 *Comparison of different attribute embeddings: binary $\{0,1\}$ embedding and continuous embedding with/without mean-centering (*i.e. *starting from the $\{0,1\}$ embedding, we compute the mean over all learning classes and subtract it) and with/without ℓ_2-norm. The ℓ_2-normalization enforces that each class is closest to itself according to the dot-product similarity.*

	FV=4K						
	OVR	WSABIE	ALE	HLE	AHLE early	AHLE late	GLE
AWA	46.9	44.0	46.9	45.0	47.1	**47.5**	**47.5**
CUB	19.4	15.9	17.4	17.4	19.1	**19.9**	18.7
	FV=64K						
	OVR	WSABIE	ALE	HLE	AHLE early	AHLE late	GLE
AWA	52.3	51.6	52.5	55.9	55.3	55.8	**56.1**
CUB	**26.6**	19.5	21.6	22.5	24.6	25.5	22.5

Table 4.7 *Comparison of different learning algorithms on the full datasets (50 resp. 200 classes). OVR and WSABIE do not use any prior information while ALE, HLE and AHLE do. Late fusion is done by calculating the classification accuracy after combining the scores of each sample classified by using either of the embeddings.*

and there is plentiful labeled data, side information does not bring extra information. Another hypothesis is that regardless of the dimensionality of the image descriptors, this is because a priori information plays a limited role when training data is plentiful. To test this hypothesis, experiments with different partitions (1/4, 2/4, 3/4 and all) of the training data on AWA and CUB have been performed. These experiments are explained in the following section.

Another conclusion from the experiments is that, AHLE with late fusion performs 3.5% higher than OVR and 4.2% higher than WSABIE on AWA with 64K FV. On CUB, WSABIE baseline performs 7% lower than OVR but when side information of attributes and hierarchies are combined in AHLE late, it performs similarly with OVR with 4K FV and

only 1% lower than OVR 64K FV. The reason for this would be when the number of classes is higher, in multiclass learning, the dimensionality of the image features makes the classes separable enough so the side information does not contribute significantly.

Reducing the training set size. We also studied the effect of reducing the amount of training data by using only 1/4, 1/2 and 3/4 of the whole training set. For these experiments, we report GLE results with two settings: using the same number of output dimensions as ALE (*i.e.* 85 for AWA and 312 for CUB) and using a large number of output dimensions (2,500 was the largest dimensionality we could afford to run the experiments in a reasonable amount of time and which led to the best performance). We show results in Table 4.8 and Figure 4.5 provides a visual summary of these results.

The conclusions are different for small- and high-dimensional features: while for the 4K-dim features AHLE always performs the best, for the 64K-dim features the simplest techniques (OVR and GLE) have a slight edge. This is consistent with the findings of Akata et al. (2013a) that showed experimentally that, for large enough features, different learning algorithms are expected to perform similarly (see also Bartlett et al. (2003)). When using poor-man's features such as the $4K$-dim features, more involved learning algorithms such as AHLE outperforms simple baselines such as OVR or GLE. However, when using large features such as the 64K-dim FVs, even simple algorithms such as the OVR which are not well-justified for multi-class classification problems can lead to state-of-the-art results. Hence, a major conclusion of our work is that a better output label embedding compensates for poor input image embeddings and vice-versa.

Comparison of Output Embeddings Apart from the attribute based side information which require manual annotation of the attributes and the hierarchy based side information which requires the presence of a sophisticated hierarchy like WordNet, different error correcting output codes (ECOC) can be used as output embeddings Allwein et al. (2000).

In this part of the experimental evaluation three different output embedding types have been used as side information, (1) OVR: the embedding matrix has N_c dichotomizers with a single bit difference, (2) OVO: the embedding matrix has $N_c(N_c-1)/2$ dichotomizers, and (3) Hadamard and Gaussian output coding.

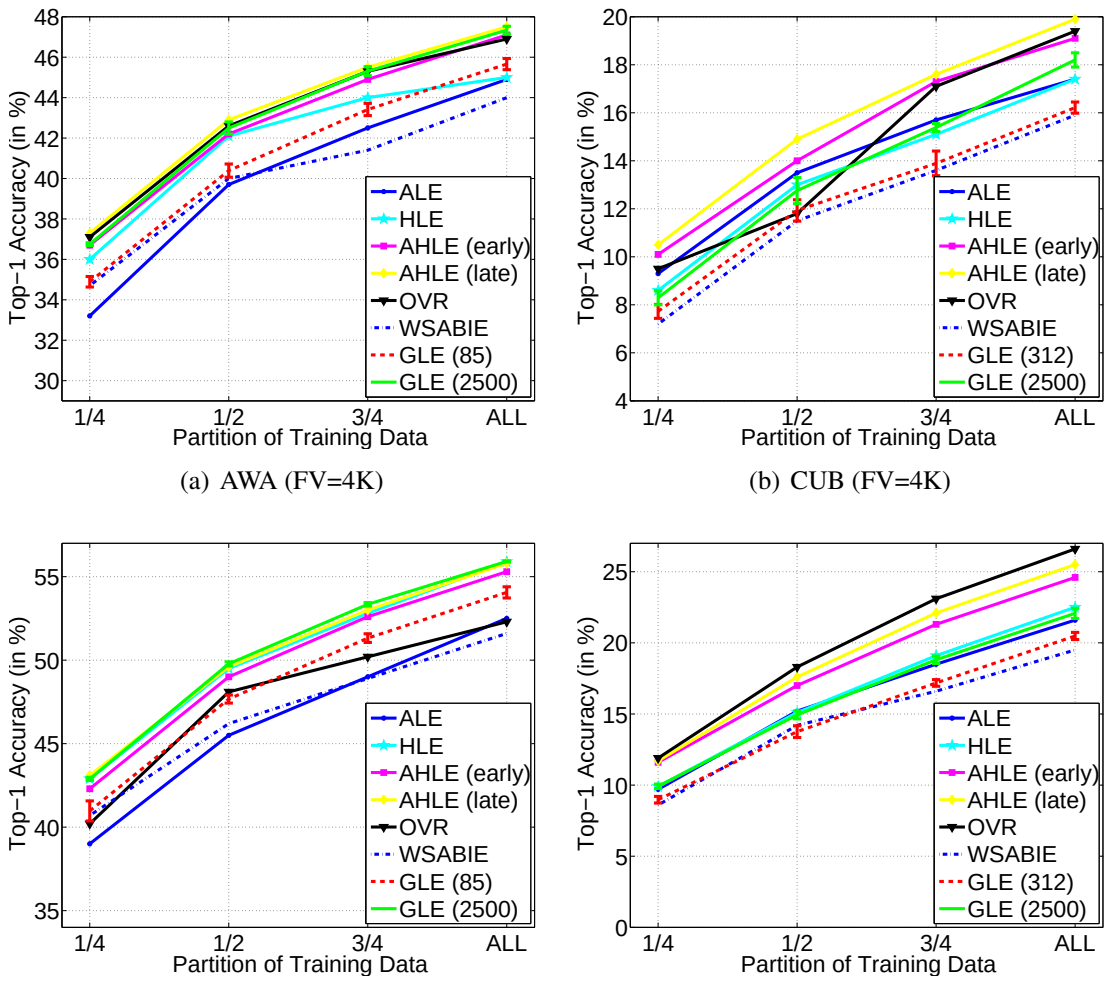
Since Hadamard Matrix is a square matrix with the number of entries as powers of two, there is a restriction in choosing the output space dimensionality. For this reason, dimensionality of the output code vectors can be at least 64 for AWA (since there are 50 classes) and 256 on CUB (since there are 200 classes). The dimensionality of the output space can be then increased to 128, 256, 512, 1024 and 2048. The Gaussian Output Matrix however can be as large or as small as necessary. Here, the starting dimensionality in Gaussian embeddings is selected as 25 and this number has been increased to 50, 100, 250, 500, 1000 and 2500. The increase stops at 2500 since from there the accuracy does not improve and also the cost of computation gets too high.

AWA	4K FV				64K FV			
	1/4	2/4	3/4	ALL	1/4	2/4	3/4	ALL
ALE	33.2	39.7	42.5	44.9	39.0	45.5	49.0	52.5
HLE	36.0	42.1	44.0	45.0	42.9	49.5	52.8	55.9
AHLE (early)	36.7	42.2	44.9	47.1	42.3	49.0	52.6	55.3
AHLE (late)	**37.3**	**42.9**	**45.5**	**47.5**	**43.1**	49.6	53.0	55.8
OVR	37.1	42.6	45.3	46.9	40.2	48.1	50.2	52.3
WSABIE	34.7	40.0	41.4	44.0	40.7	46.2	48.9	51.6
GLE (85)	34.9 (0.3)	40.4 (0.3)	43.4 (0.3)	45.7 (0.3)	41.0 (0.6)	47.7 (0.2)	51.3 (0.2)	54.1 (0.3)
GLE (2500)	36.8 (0.1)	42.5 (0.3)	45.3 (0.2)	47.3 (0.2)	42.9 (0.1)	**50.0** (0.1)	**53.4** (0.1)	**56.0** (0.1)
CUB	4K FV				64K FV			
	1/4	2/4	3/4	ALL	1/4	2/4	3/4	ALL
ALE	9.3	13.5	15.7	17.4	9.7	15.2	18.5	21.6
HLE	8.6	13.0	15.1	17.4	9.9	15.1	19.1	22.5
AHLE (early)	10.1	14.0	17.3	19.1	11.6	17.0	21.3	24.6
AHLE (late)	**10.5**	**14.9**	**17.6**	**19.9**	11.7	17.6	22.1	25.5
OVR	9.5	11.8	17.1	19.4	**11.9**	**18.3**	**23.1**	**26.6**
WSABIE	7.2	11.5	13.6	15.9	8.6	14.2	16.6	19.5
GLE (312)	7.8 (0.3)	12.0 (0.4)	14.0 (0.5)	16.2 (0.3)	9.0 (0.2)	13.8 (0.4)	17.2 (0.2)	20.5 (0.2)
GLE (2500)	8.3 (0.3)	12.8 (0.5)	15.4 (0.2)	18.2 (0.3)	9.9 (0.2)	14.9 (0.3)	18.8 (0.3)	22.1 (0.3)

Table 4.8 *Incremental learning on AWA and CUB using 1/4, 2/4, 3/4 and all the training data. Compared output embeddings: ALE, HLE, AHLE(early), AHLE(late), GLE with 85 (resp 312) dimensions, GLE with 2500 dimensions and OVR and WSABIE which do not use side information. Experiments for GLE have been repeated 10 times for different sampling of Gaussians.*

In the Gaussian case, we sampled the entries of the output embedding matrix from the standard normal distribution. To provide statistical generalization, the experiments are repeated with 10 different random samplings. The Hadamard Matrix (H) is built using the standard formulation as $\mathbf{H} * \mathbf{H^T} = \mathbf{nI_n}$ where $\mathbf{I_n}$ is the $n \times n$ identity matrix.

One conclusion from Figure 4.6 is that when the dimensionality of the output coding vectors are high enough, by using Gaussian output embeddings, we obtain a higher top-1 accuracy than ALE. Since manual attribute annotation is a costly and time consuming process, even if there is not any manually annotated attributes available for a dataset, we



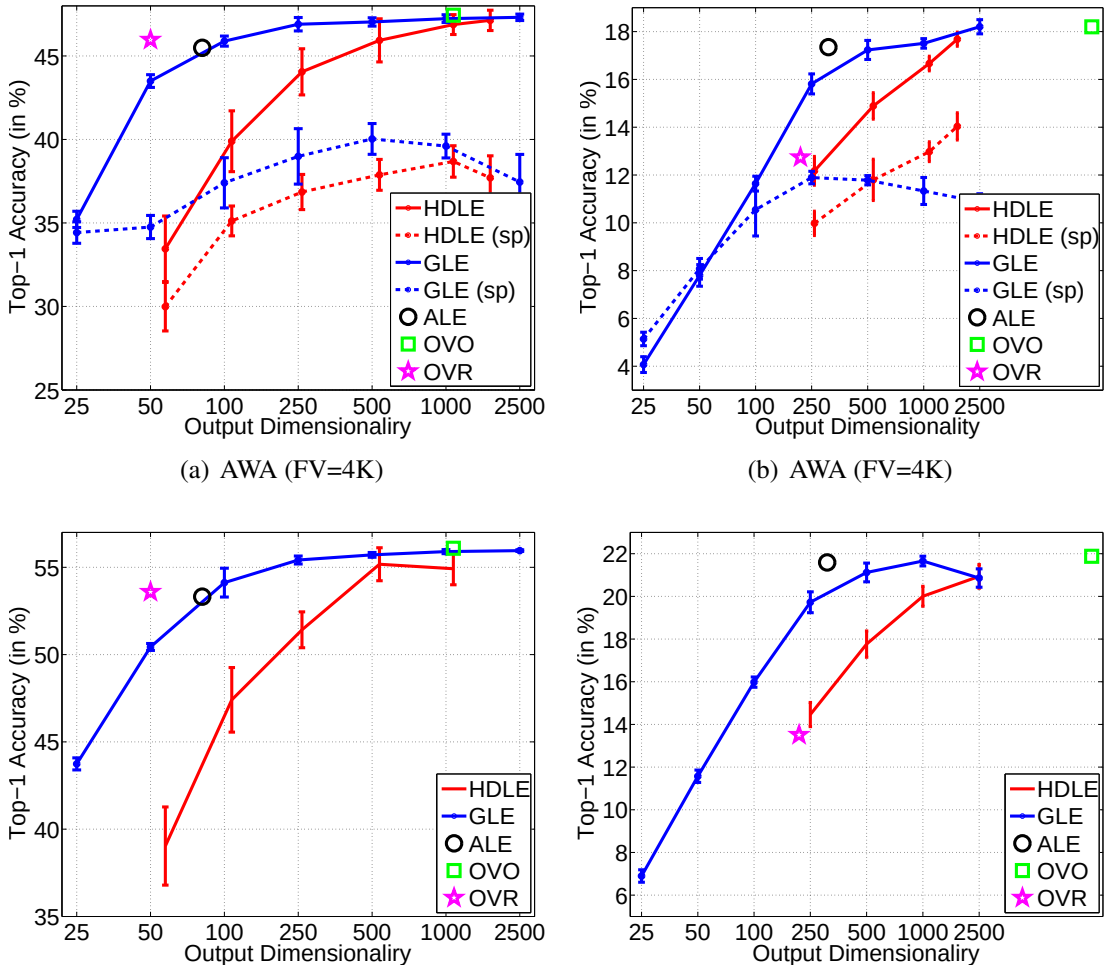
(a) AWA (FV=4K) (b) CUB (FV=4K)

(c) AWA (FV=64K) (d) CUB (FV=64K)

Figure 4.5 *Incremental learning on AWA and CUB using 1/4, 2/4, 3/4 and all the training data. Summary of the results reported on Table 4.8. Compared output embeddings: ALE, HLE, AHLE(early), AHLE(late), GLE with 85 (resp 312) dimensions, GLE with 2500 dimensions, OVR and WSABIE. Experiments for GLE have been repeated 10 times for different sampling of Gaussians.*

can construct the output embeddings using simple statistics which leads to a higher accuracy than costly user annotation. One disadvantage of this approach is that the training time increases with the increasing attribute dimensionality.

Following Weston et al. (2011), we employ a projection step where the output embedding matrix is sparsified using the Orthogonal Maching Pursuit (OMP) algorithm Mallat and Zhang (1993). We use 4K-dim FV on AWA and CUB datasets where the output code matrix Ψ is set to embeddings derived from a Gaussian distribution. Same as non-sparse



(a) AWA (FV=4K)

(b) AWA (FV=4K)

(c) AWA (FV=4K)

(d) AWA (FV=4K)

Figure 4.6 *Comparison of One vs Rest (OVR), One vs All (OVA), Attribute (ALE), Random (GLE) and Hadamard (HDLE) on FV with 4K and 64K dimensions on AWA and CUB datasets. GLE and HDLE are further compared with sparse projections Sparse Random (GLE (sp)), Sparse Hadamard (HDLE (sp)) respectively on FV with 4K dimensions.*

version, the experiments are repeated 10 times to provide statistical consistency. The results can be seen in Figure 4.6(a) for AWA and in Figure 4.6(b) for CUB.

Sparsity improved the results only when the embedding space dimensionality is low in CUB dataset. In all of the other cases, sparse projection results are lower than the dense projection results. This might be due to the fact that the label embedding method already gives a good estimate of the underlying embedding space, therefore sparse coding does not bring any advantage.

To test this hypothesis, we use ridge regression to predict the embedding space which is less optimal than label embedding method. For the speed of computation, we use the

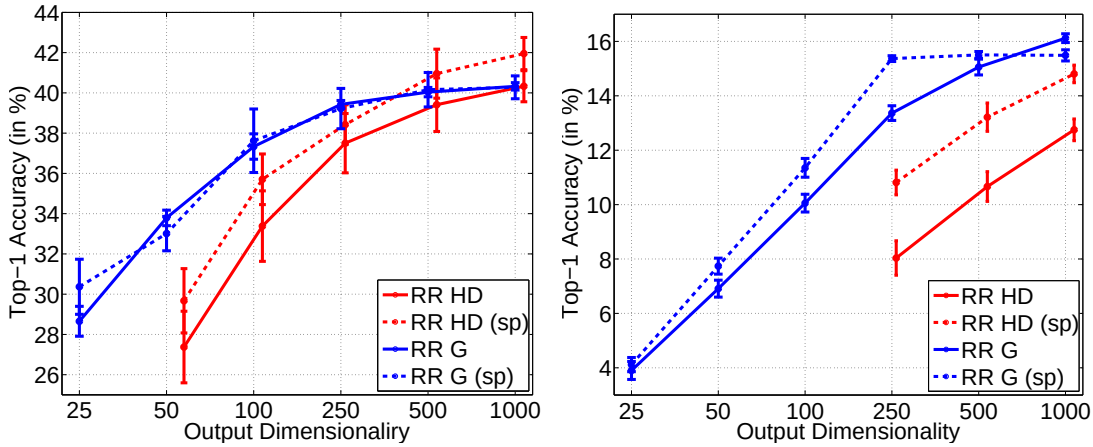


Figure 4.7 *Comparison of Random (RR G), Sparse Random (RR G (sp)), Hadamard (RR HD)and Sparse Hadamard (RR HDLE (sp)) embeddings used as side information when the embedding matrix is predicted using the ridge regression framework on AWA (top) and CUB (bottom). (FV=4K)*

lower dimensional FV with 4K. The results empirically prove the previous hypothesis that sparsity improves results when the underlying embedding space is predicted sub-optimally. Figure 4.7 shows that in all cases in comparison to the dense projection with ridge regression, there is an improvement in accuracy. The accuracy increase becomes more prominent when the embedding space dimensionality is lower. This is expected because projecting data in lower dimensional space leads to a worse prediction of the embedding space and therefore sparse coding is useful. Therefore, the conclusion is, when the optimization method predicts the embedding space poorly, as in ridge regression, sparse coding helps.

4.5 Conclusion

In this chapter we cast the problem of attribute-based classification as one of label-embedding. The proposed Attribute Label Embedding (ALE) approach addresses in a principled fashion the limitations of the original DAP model. First, we solve directly the problem at hand (image classification) without introducing an intermediate problem (attribute classification). Second, our model can leverage labeled training data (if available) to update the label embedding, using the attribute embedding as a prior. Third, the label embedding framework is not restricted to attributes and can accommodate other sources of prior information such as class taxonomies or automatically constructed output embeddings.

In the zero-shot setting, we improved image classification results with respect to DAP without losing attribute interpretability. Continuous attributes can be effortlessly used in

ALE, leading to a large boost in zero-shot classification accuracy. As an addition, we have shown that the dimensionality of the output space can be significantly reduced with a small loss of accuracy. In the few-shots setting, we showed improvements with respect to the WSABIE algorithm, which learns the label embedding from labeled data but does not leverage prior information. Finally, in experiments where training data is plentiful, we showed that label embedding can still be useful, when dealing with simpler (*i.e.* lower-dimensional) features. Hence a good output embedding has the potential to compensate for a poor input embedding.

In this chapter, we considered very simple techniques for combining different outputs embeddings, for instance through the combination of attributes and class hierarchies. While there is an abundant literature in the machine learning and computer vision communities on how to combine several inputs, for instance using a Multiple Kernel Learning framework (see for instance Gehler and Nowozin (2009)), there is relatively little work on the combination of outputs, a noticeable exception being Hwang et al. (2012). We believe this is a worthwhile research path to pursue.

5

Conclusion

Contents

5.1 Contributions 95

5.2 Future Directions 97

Classifying images based on their content is essential for managing image collections. However, image classification is a difficult task especially with a large number of classes and images. To make it feasible, the state-of-the-art practice is to use high dimensional image representations in combination with efficient linear classifiers. This thesis, building on such image representations, focused on the learning problem for image classification in a large-scale setting. In the following, we summarize our contributions in Section 5.1 and discuss about the future work in Section 5.2.

5.1 Contributions

In this section, we summarize our contributions on two challenges of large-scale problems: (1) Large-scale learning with linear classifiers using high dimensional features on real-world datasets where the number of classes is in the order of thousands and the number of images is in the order of millions. (2) Learning with a lack of labeled training data, also known as the zero-shot learning problem.

Large Scale Image Classification. We first focused on the problem of learning with many samples, *i.e.* up to 4.5 million images in our experiments. We proposed a benchmark of supervised learning algorithms for linear classifiers: One-vs-Rest, Multiclass, Ranking and Weighted Average Ranking. We focused on online learning with stochastic gradient descent and provided a detailed analysis of several parameters such as learning rate, regularization, *etc*. The outcome of this work was a set of good practices for large-scale learning:

1. Stochastic Training: Online learning algorithms such as Stochastic Gradient Descent (SGD) lead to similar results as batch learning algorithms. As SGD requires to keep only the image descriptor of a single example in the memory, the classifiers are computed efficiently.

2. Data Rebalancing: In large scale datasets where the number of classes is in the order of thousands, the imbalance in the number of positive versus negative samples degrades the robustness of one-vs-rest classifiers. Therefore, we show that natural rebalancing is not sufficient, data rebalancing is essential for state-of-the-art performance using one-vs-rest classifiers.

3. Early Stopping: Another good practice in large-scale learning is to eliminate the number of parameters to be tuned as much as possible. Therefore, from our experiments with implicit and explicit regularization, we conclude that regularization through early stopping results in fast training and good generalization performance for large scale learning.

4. Step Size: Using a small-enough fixed step size is a competitive alternative to decreasing the step size. A small-enough fixed step size with respect to the learning rate is often sufficient for state-of-the-art performance.

5. One-vs-Rest: One might think the more complicated the problem gets, the more complicated the learning algorithm should be. When the number of training images per class is small, ranking(RNK) based learning consistently and significantly outperforms one-vs-rest(OVR) based learning. However, we show that in a large scale setting, a straightforward and efficient learning algorithm such as OVR performs better than more complicated ranking based methods.

6. Capacity Saturation: With high dimensional image descriptors, all the methods perform similarly. However, the dimensionality of the image descriptors is relative to the size of the dataset. For the ILSVRC10 dataset, the 130K dimensional FVs result in the state-of the-art accuracy, whereas for ImageNet10K, 130K dimensional FVs are not descriptive enough for the samples to be linearly separable. Therefore, a larger dataset requires higher dimensional image representations.

Zero-Shot Learning. With an increasing number of classes, the density of the dataset increases, meaning that the average distance between two classes decreases. However, labeling the images that belong to similar object classes, *i.e.* fine-grained sets of classes, requires domain knowledge. Obtaining such knowledge through expert opinion is a costly process. Therefore, for some classes, there is a lack of labeled training data. In the extreme case, we might not have a single training sample to train some of the classes. One solution to this zero-shot scenario is to use side information such as attributes that provide a connection between classes. We proposed a novel algorithm which consists

of embedding the classes in a Euclidean space using side information such as attributes, hierarchies *etc*. We introduced a compatibility function between input and outputs where we learn the model parameters using a ranking objective, focusing on stochastic gradient descent learning. We can conclude from our experiments reported in Chapter 4 that:

1. Our approach, Attribute Label Embedding (ALE), obtains a higher accuracy than the state-of-the-art DAP method of Lampert et al. (2009) in the zero-shot learning scenario.

2. For a method such as DAP, where the class membership is decided through attribute classifiers, the attributes are encoded with binary vectors. However, for ALE it is straightforward to encode the attributes with continuous numbers which provide a confidence score for each attribute. This modification leads to a significant improvement over binary attributes.

3. Another advantage of ALE is the ability to accommodate other sources of side information such as class hierarchies. We showed that the combination of attributes and hierarchies can lead to further improvements.

4. We also showed experimentally that with sufficient training data, random label embeddings can outperform embeddings that are based on side information.

5. Furthermore, with the ALE method, it is straightforward to integrate more training data in the learning process when the availability of the annotated training images increases. In such cases, the label embedding framework uses the previous embeddings to build a new set of embeddings using the extra training images.

5.2 Future Directions

The focus of this thesis was on large-scale image classification, in particular the computational challenges due to the large scale nature of the dataset and the lack of labeled training data. Even though in our work we have pushed the state-of-the-art accuracy, we are still far away from a 100% success rate in large scale image collections. Moreover, as the datasets are still growing, the difficulty of classification is constantly increasing. Therefore, there exists many unanswered questions. In this section, we discuss several extensions to the work conducted within this thesis that might lead to better solutions to the problems at hand.

Deep Learning. We wish to explore deep learning from two different points of view: (1) as an alternative to the standard supervised image classification pipeline. (2) as an architecture to be used in zero-shot learning to learn better embeddings.

1. The standard feature extraction pipeline which aggregates the patch descriptors can be criticized as being too shallow to learn complex invariances and high-level concepts. Recently, several large-scale works have considered learning features directly from pixel values using deeper architectures Bengio et al. (2013); Le et al. (2012); Krizhevsky et al. (2012). One crucial factor to obtain good results when learning deep architectures with millions if not billions of parameters is the availability of vast amounts of training data (see section 10.1 in Bengio et al. (2013)). In Le et al. (2012); Dean et al. (2012), the features are learned using a deep autoencoder which is constructed by replicating three times the same three layers – made of local filtering, local pooling and contrast normalization – thus resulting in an architecture with 9 layers. The learned features were shown to give excellent results in combination with a simple linear classifier. In Krizhevsky et al. (2012), a deep network with 8 layers was proposed where the first 5 layers are convolutional LeCun et al. (1989, 2004); Jarrett et al. (2009), the remaining three are fully connected and the output of the last fully connected layer is fed to a softmax which produces a distribution over the class labels. For more details on deep architectures, interested readers can refer to a recent survey paper by Bengio et al. (2013).

2. Side information comes in many different forms such as attributes, hierarchies, *etc*. which have different behavior depending on the dataset and the amount of training data. In order to benefit from all sources of side information, concatenating the output code vectors or averaging the scores are the most straightforward approaches. We showed in Chapter 4 that such a simple combination of different output codes consistently leads to significant improvements.

   However, it is still not clear which is the best strategy to combine the output codes, and the difference between various approaches should be further investigated. There exists very few works on combining multiple output embeddings. One exception is the work of Hwang et al. (2012) who propose a kernel forest approach based on Multiple Kernel Learning (MKL) to learn discriminative visual features using semantic information from different semantic taxonomies. This leads to significant accuracy improvements on a subset of ImageNet and the AWA dataset which gives good evidence that committing to only one source of side information is insufficient. We propose as an alternative, to learn the embedding spaces one after the other from several sources of side information. In such a case, the images are projected through a nonlinearity function on the embedding space that was learned using one channel, such as attributes. Stacking such embeddings would lead to a deeper and deeper architecture.

Sampling Methods. In Chapter 3, we showed that random sampling of the training images with or without replacement in the context of online learning gave similar results. In large scale datasets, the number of training images is very high. However, a vast majority of these examples bring little information. Sampling methods can help decreasing

the size of the training set by selecting informative samples that would lead to faster convergence. Therefore, in the literature, many sampling methods exist such as gradient selection Loosli et al. (2005), active and auto-active selection Bordes et al. (2005), *etc*. Gradient selection picks the most poorly classified sample among a batch of randomly selected samples and updates the classifier using that sample. Active selection picks the sample that is closest to the decision boundary. Auto-active selection has the ability to select a large amount of samples randomly. It stops when a predefined amount of samples fall inside the margin and selects the one that is closest to the decision boundary to do the gradient update. Loosli et al. (2007) applied random sampling, active selection and auto-active selection to the MNIST dataset and shows that auto-active selection leads to accuracy and efficiency improvements over random sampling. Vijayanarasimhan et al. (2010) proposed a method for optimally selecting a set of examples for a support vector machine classifier and optimized the learning with an efficient iterative minimization technique. However, the active sampling methods become impractical if they are exhaustively applied to all unlabeled points at each round of learning for large scale datasets. Vijayanarasimhan et al. (2013) proposed to use randomized hash functions to be more efficient in approximate hyperplane-to-point search. Gorisse et al. (2011) focuses on both the sample selection and the ranking process to make active learning strategies scalable to a database of 180K images. However, sampling methods have not been investigated in datasets as large as the ImageNet. Therefore, how to integrate them efficiently and to improve the accuracy in large scale image classification is a future work that we believe is worth investigating.

Loss Function. The objective functions that are explained in Chapter 3 use a zero-one loss. During training, if our algorithm predicts a wrong label for a sample, the misclassification loss assigned to that sample is 1, otherwise it is 0. Depending on the objective function, we update the classifiers accordingly. However, for large scale datasets such as the ImageNet, there exists a well defined hierarchy among the classes. In such a case, using a hierarchical loss Tsochantaridis et al. (2005) is the most intuitive alternative to condition the update procedure.

On the other hand, the SVM objective upper bounds the misclassification loss with the hinge loss. However, in the literature other loss formulations exist. For instance, if we replace the hinge loss with quadratic loss, *i.e.*, $l(\mathbf{x}_i, y_i, \mathbf{w}) = (y_i - \mathbf{w}^T\mathbf{x}_i)^2$, the regularized risk minimization objective becomes ridge regression Hoerl and Kennard (1970) for which there exists a closed form solution. Ridge regression can yield similar results to SVMs especially for high dimensional data Rahimi and Recht (2007). Therefore, we propose to explore the impact of different loss functions in the linear SVM formulations as a line of future work.

Side Information. In Chapter 4, we investigated attributes and hierarchies as side information. However, obtaining attributes through crowdsourcing is a costly process. There-

fore, as a future work we propose to extract attribute-like side information from public resources such as internet repositories. The necessary knowledge about the unseen classes can be created using unsupervised text corpora as proposed in Socher et al. (2013). As an example to such corpora, the expert created language ontology of WordNet, the articles in Wikipedia and short summary texts returned by search engines can be used Rohrbach et al. (2010b). As an alternative, the semantic representation of the visual content can be generated using videos Rohrbach et al. (2013). Mining side information from text corpora is worth investigating as a future work.

Publications

Articles in peer-reviewed journals

- Label Embedding for Image Classification,
  Z.Akata, F. Perronnin, Z.Harchaoui, C.Schmid,
  Submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI).

- Good Practice in Large Scale Learning for Image Classification,
  Z.Akata, F. Perronnin, Z.Harchaoui, C.Schmid,
  IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 2013.

International peer-reviewed conferences

- Label-Embedding for Attribute-Based Classification,
  Z.Akata, F. Perronnin, Z.Harchaoui, C.Schmid,
  IEEE Computer Vision and Pattern Recognition (CVPR), 2013.

- Towards Good Practice in Large Scale Learning for Image Classification,
  F. Perronnin, Z.Akata, Z.Harchaoui, C.Schmid,
  IEEE Computer Vision and Pattern Recognition (CVPR), 2012.

- Fisher Vectors for Fine-Grained Visual Categorization,
  J. Sanchez, F. Perronnin, Z.Akata, Z.Harchaoui,
  1st FGVC Workshop in IEEE Computer Vision and Pattern Recognition (CVPR), 2011.

Patents

- Label-Embedding View of Attribute-Based Recognition,
  Z.Akata, F. Perronnin, Z.Harchaoui, C.Schmid,
  Submitted to United States Patent and Trademark Office (USPTO), 2013

Bibliography

Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid. Label-Embedding for Attribute-Based Classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013a.

Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid. Good Practice in Large-Scale Learning for Image Classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2013b.

Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid. Label-Embedding for Image Classification. 2013c.

E. L. Allwein, R. E. Schapire, and Y. Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. In *International Conference on Machine Learning (ICML)*, 2000.

Y. Amit, M. Fink, N. Srebro, and S. Ullman. Uncovering shared structures in multiclass classification. In *International Conference on Machine Learning (ICML)*, 2007.

C. Arms. Historical collections for the national digital library: Lessons and challenges at the library of congress [part i]. *D-Lib Magazine*, 1996.

B. Bai, J. Weston, D. Grangier, R. Collobert, O. Chapelle, and K. Weinberger. Supervised semantic indexing. In *ACM International Conference on Information and Knowledge Management (CIKM)*, 2009.

P. L. Bartlett, M. I. Jordan, and J. D. McAuliffe. Convexity, classification, and risk bounds. In *Advances in Neural Information Processing Systems (NIPS)*, 2003.

H. Bay, T. Tuytelaars, and L. V. Gool. Surf: Speeded up robust features. In *European Conference of Computer Vision (ECCV)*, 2006.

S. Bengio, J. Weston, and D. Grangier. Label embedding trees for large multi-class tasks. In *Advances in Neural Information Processing Systems (NIPS)*, 2010.

Y. Bengio, A. C. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis Machine Intelligence (TPAMI)*, 2013.

A. Berg, J. Deng, and L. Fei-Fei. Imagenet large scale visual recognition challenge (ilsvrc). http://www.image-net.org/challenges/LSVRC/2010/index, 2010a.

T. Berg, A. Berg, and J. Shih. Automatic attribute discovery and characterization from noisy web data. In *European Conference of Computer Vision (ECCV)*, 2010b.

A. Bergamo, L. Torresani, and A. Fitzgibbon. PICODES: Learning a compact code for novel-category recognition. In *Advances in Neural Information Processing Systems (NIPS)*, 2011.

A. Beygelzimer, V. Dani, T. P. Hayes, J. Langford, and B. Zadrozny. Error limiting reductions between classification tasks. In *International Conference on Machine Learning (ICML)*, 2005.

I. Biederman. Recognition-by-components: A theory of human image understanding. *Psychological Review*, 1987.

A. Bordes, L. Bottou, P. Gallinari, and J. Weston. Solving multiclass support vector machines with LaRank. In *International Conference on Machine Learning (ICML)*, 2007.

A. Bordes, S. Ertekin, J. Weston, and L. Bottou. Fast kernel classifiers with online and active learning. *ACM Journal of Machine Learning Research (JMLR)*, 2005.

L. Bottou. Stochastic learning. In *Machine Learning Summer School (MLSS)*, 2003.

L. Bottou and O. Bousquet. The tradeoffs of large scale learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.

L. Bottou. SGD. leon.bottou.org/_media/projects/sgd-1.1.tar.gz, 1997.

Y.-L. Bourreau, F. Bach, Y. LeCun, and J. Ponce. Learning mid-level features for recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

S. Branson, C. Wah, B. Babenko, F. Schroff, P. Welinder, P. Perona, and S. Belongie. Visual recognition with humans in the loop. In *European Conference of Computer Vision (ECCV)*, 2010.

M. Brown and D. G. Lowe. Recognising panoramas. In *IEEE International Conference of Computer Vision (ICCV)*, 2003.

P. K. Chan and S. J. Stolfo. On the accuracy of meta-learning for scalable data mining. *Springer Journal of Intelligent Information Systems (JIIS)*, 1996.

C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM TIST*, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

K. Chatfield, V. Lempitsky, A. Vedaldi, and A. Zisserman. The devil is in the details: an evaluation of recent feature encoding methods. In *British Machine Vision Conference (BMVC)*, 2011.

H. Chen, A. Gallagher, and B. Girod. Describing clothing by semantic attributes. In *European Conference of Computer Vision (ECCV)*, 2012.

H. Chen, A. Gallagher, and B. Girod. What's in a name? first names as facial attributes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

R. G. Cinbis, J. Verbeek, and C. Schmid. Image categorization using Fisher kernels of non-iid image models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

S. Clinchant, G. Csurka, F. Perronnin, and J.-M. Renders. XRCE's participation to ImageEval. In *ImageEval Workshop at CVIR*, 2007.

K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research (JMLR)*, 2, 2002.

N. Craswell and M. Szummer. Random walks on the click graph. In *ACM Special Interest Group on Information Retrieval (SIGIR)*, 2007.

G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *In Workshop on Statistical Learning in Computer Vision, European Conference of Computer Vision (ECCV)*, 2004.

R. Datta, D. Joshi, J. Li, and J. Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys*, 2008.

J. Dean, G. Corrado, R. Monga, M. Devin, Q. Le, M. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, and A. Ng. Large scale distributed deep networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B*, 1977.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

J. Deng, A. Berg, K. Li, and L. Fei-Fei. What does classifying more than 10,000 image categories tell us? In *European Conference of Computer Vision (ECCV)*, 2010.

J. Deng, S. Satheesh, A. Berg, and L. Fei-Fei. Fast and balanced: efficient label tree learning for large scale object recognition. In *Advances in Neural Information Processing Systems (NIPS)*, 2011.

T. Deselaers and V. Ferrari. Visual and semantic similarity in ImageNet. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

R. A. DeVore. Deterministic constructions of compressed sensing matrices. *ACM Journal of Complexity*, 2007.

T. G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of the Artificial Intelligence Research (JAIR)*, 1995.

M. Douze, H. Jégou, H. Sandhawalia, L. Amsaleg, and C. Schmid. Evaluation of gist descriptors for web-scale image search. In *ACM International Conference on Image and Video Retrieval (CIVR)*, 2009.

M. Douze, A. Ramisa, and C. Schmid. Combining attributes and Fisher vectors for efficient image retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

K. Duan, D. Parikh, D. J. Crandall, and K. Grauman. Discovering localized attributes for fine-grained recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

S. Escalera, O. Pujol, and P. Radeva. Error-correcting ouput codes library. *Journal of Machine Learning Research (JMLR)*, 2010.

M. Everingham, L. V. Gool, C. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (VOC) challenge. *International Journal of Computer Vision (IJCV)*, 2010.

R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research (JMLR)*, 2008.

A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

J. Farquhar, S. Szedmak, H. Meng, and J. Shawe-Taylor. Improving bag-of-keypoints image categorisation. Technical report, University of Southampton, 2005.

L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2006.

R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2003.

V. Ferrari and A. Zisserman. Learning visual attributes. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.

V. Ferrari, T. Tuytelaars, and L. van Gool. Simultaneous object recognition and segmentation from single or multiple model views. *International Journal of Computer Vision (IJCV)*, 2006.

M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker. Query by image and video content: The qbic system. *Computer*, 1995.

D. Forsyth, A. Efros, L. Fei-Fei, A. Torralba, and A. Zisserman. The promise and perils of benchmark datasets and challenges. In *Workshop on Frontiers of Computer Vision (FCV)*, 2011.

V. Franc and S. Sonnenburg. Optimized cutting plane algorithm for support vector machines. In *International Conference on Machine Learning (ICML)*, 2008.

T. Gao and D. Koller. Discriminative learning of relaxed hierarchy for large-scale visual recognition. In *IEEE International Conference of Computer Vision (ICCV)*, 2011.

P. Gehler and S. Nowozin. On feature combination for multiclass object classification. In *ICCV*, 2009.

J. Gehrke, R. Ramakrishnan, and V. Ganti. Rainforest - a framework for fast decision tree construction of large datasets. *Data Mining and Knowledge Discovery (DMKD)*, 2000.

J. C. Gemert, J.-M. Geusebroek, C. J. Veenman, and A. W. Smeulders. Kernel codebooks for scene categorization. In *European Conference of Computer Vision (ECCV)*, 2008.

B. Geng, L. Yang, C. Xu, and X.-S. Hua. Ranking model adaptation for domain-specific search. *IEEE Transactions on Knowledge and Data Engineering (IEEE TKDE)*, 2012.

Y. Gong, T. Huang, F. Lv, J. Wang, C. Wu, W. Xu, J. Yang, K. Yu, T. Zhang, and X. Zhou. Image classification using Gaussian mixture and local coordinate coding. PASCAL VOC 2009 workshop, http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2009/ workshop/yu.pdf, 2009.

Y. Gong and S. Lazebnik. Comparing data-dependent and data-independent embeddings for classification and ranking of internet images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

D. Gorisse, M. Cord, and F. Precioso. Salsas: Sub-linear active learning strategy with approximate k-nn search. *Pattern Recognition*, 2011.

D. Grangier and S. Bengio. A discriminative kernel-based model to rank images from text queries. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2008.

D. Grangier, F. Monay, and S. Bengio. A discriminative approach for the retrieval of images from text queries. In *European Conference of Machine Learning (ECML)*, 2006.

G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical report, California Institute of Technology, 2007.

R. Hamming. Error Detecting and Error Correcting Codes. *Bell System Technical Journal*, 1950.

Z. Harris. Distributional structure. *Word*, 1954.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning (2nd Ed.)*. Springer Series in Statistics. Springer, 2008.

A. E. Hoerl and R. W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 1970.

D. Hsu, S. Kakade, J. Langford, and T. Zhang. Multi-label prediction via compressed sensing. In *Advances in Neural Information Processing Systems (NIPS)*, 2009.

S. J. Hwang, K. Grauman, and F. Sha. Semantic kernel forests from multiple taxonomies. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

T. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In *Advances in Neural Information Processing Systems (NIPS)*, 1998.

K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun. What is the best multi-stage architecture for object recognition? In *IEEE International Conference of Computer Vision (ICCV)*, 2009.

H. Jégou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 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 (TPAMI)*, 2012.

T. Joachims. Making large-scale support vector machine learning practical. In *Advances in Kernel Methods*. MIT Press, 1999.

T. Joachims. Optimizing search engines using clickthrough data. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2002.

A. Krizhevsky, I. Sutskever, and G. Hinton. Image classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

G. Kulkarni, V. Premraj, S. Dhar, S. Li, Y. choi, A. Berg, and T. Berg. Baby talk: understanding and generating simple image descriptions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

N. Kumar, P. Belhummeur, and S. Nayar. FaceTracer: A search engine for large collections of images with faces. In *European Conference of Computer Vision (ECCV)*, 2008.

N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Attribute and simile classifiers for face verification. In *IEEE International Conference of Computer Vision (ICCV)*, 2009.

C. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

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 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.

Q. Le, M. Ranzato, R. Monga, M. Devin, K. Chen, G. Corrado, J. Dean, and A. Ng. Building high-level features using large scale unsupervised learning. In *International Conference on Machine Learning (ICML)*, 2012.

Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel. Handwritten digit recognition with a back-propagation network. In *Advances in Neural Information Processing Systems (NIPS)*, 1989.

Y. LeCun, L. Bottou, G. Orr, and K. Muller. Efficient backprop. In *Neural Networks: Tricks of the trade*. Springer, 1998.

Y. LeCun, F. Huang, and L. Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.

T. Lee, Y. Lin, and G. Wahba. Multicategory support vector machines: Theory and application to the classification of microarray data and satellite radiance data. *Journal of the Acoustical Society of America (JAAS)*, 2004.

L. Li, H. Su, E. Xing, and L. Fei-Fei. Object bank: A high-level image representation for scene classification and semantic feature sparsification. In *Advances in Neural Information Processing Systems (NIPS)*, 2010.

Z. Li, H. Ning, L. Cao, T. Zhang, Y. Gong, and T. S. Huang. Learning to search efficiently in high dimensions. In *Advances in Neural Information Processing Systems (NIPS)*, 2011.

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 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

J. Liu, B. Kuipers, and S. Savarese. Recognizing human actions by attributes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

S. P. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory (TIT)*, 1982.

G. Loosli, S. Canu, and L. Bottou. Training invariant support vector machines using selective sampling. In L. Bottou, O. Chapelle, D. DeCoste, and J. Weston, editors, *Large Scale Kernel Machines*. MIT Press, 2007.

G. Loosli, S. Canu, S. V. N. Vishwanathan, and A. J. Smola. Invariances in classification: an efficient svm implementation. In *Proceedings of the 11th International Symposium on Applied Stochastic Models and Data Analysis*, 2005.

D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 2004.

D. Mahajan, S. Sellamanickam, and V. Nair. A joint learning framework for attribute models and object descriptions. In *IEEE International Conference of Computer Vision (ICCV)*, 2011.

S. Maji and A. Berg. Max-margin additive classifiers for detection. In *IEEE International Conference of Computer Vision (ICCV)*, 2009.

S. Mallat and Z. Zhang. Matching pursuit with time-frequency dictionaries. *IEEE Transactions on Signal Processing (TSP)*, 1993.

L. Marchesotti and F. Perronnin. Learning beautiful (and ugly) attributes. In *British Machine Vision Conference (BMVC)*, 2013.

R. Marée, P. Geurts, J. H. Piater, and L. Wehenkel. Random subwindows for robust image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005.

M. Marszalek and C. Schmid. Constructing category hierarchies for visual recognition. In *European Conference of Computer Vision (ECCV)*, 2008.

M. Mehta, R. Agrawal, and J. Rissanen. Sliq: A fast scalable classifier for data mining. In *International Conference on Extending Database Technology (EDBT)*, 1996.

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 *European Conference of Computer Vision (ECCV)*, 2012a.

T. Mensink, J. Verbeek, and G. Csurka. Tree-structured CRF models for interactive image labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2012b.

K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. In *IEEE European Conference on Computer Vision (ICCV)*, 2002.

K. Mikolajczyk and C. Schmid. Indexing based on scale invariant interest points. In *IEEE International Conference of Computer Vision (ICCV)*, 2001.

K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2005.

G. A. Miller. Wordnet: a lexical database for English. *Communications of the ACM (CACM)*, 1995.

S. Nowozin and C. Lampert. Structured learning and prediction in computer vision. *Foundations and Trends in Computer Graphics and Vision*, 2011.

A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision (IJCV)*, 2001.

V. Ordonez, G. Kulkarni, and T. Berg. Im2Text: Describing images using 1 million captioned photographs. In *Advances in Neural Information Processing Systems (NIPS)*, 2011.

D. Osherson, J. Stern, O. Wilkie, M. Stob, and E. Smith. Default probability. *Cognitive Science Journal (CSJ)*, 1991.

M. Palatucci, D. Pomerleau, G. Hinton, and T. Mitchell. Zero-shot learning with semantic output codes. In *Advances in Neural Information Processing Systems (NIPS)*, 2009.

D. Parikh and C. L. Zitnick. The role of features, algorithms and data in visual recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

F. Perronnin and C. Dance. Fisher kernels on visual vocabularies for image categorization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

F. Perronnin, J. Sánchez, and Y. Liu. Large-scale image categorization with explicit data embedding. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010a.

F. Perronnin, J. Sánchez, and T. Mensink. Improving the Fisher kernel for large-scale image classification. In *European Conference of Computer Vision (ECCV)*, 2010b.

F. Perronnin, Z. Akata, Z. Harchaoui, and C. Schmid. Towards good practice in large-scale learning for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

J. C. Platt. Fast training of support vector machines using sequential minimal optimization. In *Advances in Kernel Methods*. MIT Press, 1999.

A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *In Neural Infomration Processing Systems (NIPS)*, 2007.

M. Rastegari, C. Fang, and L. Torresani. Scalable object-class retrieval with approximate and top-k ranking. In *IEEE International Conference of Computer Vision (ICCV)*, 2011.

R. Rifkin and A. Klautau. In defense of one-vs-all classification. *Journal of Machine Learning Research (JMLR)*, 2004.

M. Rohrbach, M. Stark, G. Szarvas, I. Gurevych, and B. Schiele. What hepls here – and why? Semantic relatedness for knowledge transfer. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010a.

M. Rohrbach, M. Stark, and B.Schiele. Evaluating knowledge transfer and zero-shot learning in a large-scale setting. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

M. Rohrbach, M. Stark, G. Szarvas, and B. Schiele. Combining language sources and robust semantic relatedness for attribute-based knowledge transfer. In *Parts and Attributes Workshop at European Conference of Computer Vision (ECCV)*, 2010b.

M. Rohrbach, W. Qiu, I. Titov, S. Thater, M. Pinkal, and B. Schiele. Translating video content to natural language descriptions. In *IEEE International Conference on Computer Vision (ICCV)*, 2013.

L. Rokach and O. Maimon. Top-down induction of decision trees classifiers - a survey. *IEEE Transactions on Systems, Man, and Cybernetics (TSMC)*, 2005.

B. Russell. *History of Western Philosophy*. Routledge classics. Routledge, 1946.

B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. Labelme: A database and web-based tool for image annotation. *International Journal of Computer Vision (IJCV)*, 2008.

M. Saerens, F. Fouss, L. Yen, and P. Dupont. The principal components analysis of a graph, and its relationships to spectral clustering. In *European Conference of Machine Learning (ECML)*, 2004.

S. L. Salzberg. On comparing classifiers: Pitfalls to avoid and a recommended approach. *Data Mining and Knowledge Discovery (DMKD)*, 1997.

J. Sánchez and F. Perronnin. High-dimensional signature compression for large-scale image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

J. Sánchez, F. Perronnin, and Z. Akata. Fisher Vectors for Fine-Grained Visual Categorization. In *FGVC Workshop in IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2011.

J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek. Image Classification with the Fisher Vector: Theory and Practice. *International Journal of Computer Vision (IJCV)*, 2013.

W. J. Scheirer, N. Kumar, P. N. Belhumeur, and T. E. Boult. Multi-attribute spaces: Calibration for attribute fusion and similarity search. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal estimate sub-gradient solver for SVM. In *International Conference on Machine Learning (ICML)*, 2007.

V. Sharmanska, N. Quadrianto, and C. H. Lampert. Augmented attribute representations. In *European Conference of Computer Vision (ECCV)*, 2012.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge Univ. Press, 2004.

E. Shechtman and M. Irani. Matching local self-similarities across images and videos. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

D. J. Sheskin. *Handbook of Parametric and Nonparametric Statistical Procedures*. Chapman & Hall/CRC, 4 edition, 2007.

B. Siddiquie, R. Feris, and L. Davis. Image ranking and retrieval based on multi-attribute queries. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *IEEE International Conference of Computer Vision (ICCV)*, 2003.

J. Sivic and A. Zisserman. Efficient visual search of videos cast as text retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2009.

A. W. M. 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 (TPAMI)*, 2000.

R. Socher, M. Ganjoo, H. Sridhar, O. Bastani, C. Manning, and A. Ng. Zero-shot learning through cross-modal transfer. In *International Conference on Learning Representations (ICLR)*, 2013.

R. Szeliski. *Computer Vision: Algorithms and Applications*. Springer-Verlag New York, Inc., New York, NY, USA, 1st edition, 2010.

K. Tang, M. Tappen, R. Sukthankar, and C. Lampert. Optimizing one-shot recognition with micro-set learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

D. Tao, X. Tang, X. Li, and X. Wu. Asymmetric bagging and random subspace for support vector machines-based relevance feedback in image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2006.

A. Tewari and P. L. Bartlett. On the Consistency of Multiclass Classification Methods. *Journal of Machine Learning Research (JMLR)*, 2007.

A. Torralba, R. Fergus, and W. Freeman. 80 million tiny images: a large dataset for non-parametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2008.

L. Torresani, M. Szummer, and A. Fitzgibbon. Efficient object category recognition using classemes. In *European Conference of Computer Vision (ECCV)*, 2010.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research (JMLR)*, 2005.

T. Tuytelaars. Dense interest points. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

T. Tuytelaars and L. Van Gool. Matching widely separated views based on affine invariant regions. *International Journal of Computer Vision (IJCV)*, 2004.

N. Usunier, D. Buffoni, and P. Gallinari. Ranking with ordered weighted pairwise classification. In *International Conference on Machine Learning (ICML)*, 2009.

K. van de Sande, T. Gevers, and C. Snoek. Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2010.

J. C. van Gemert, C. J. Veenman, A. W. M. Smeulders, and J. M. Geusebroek. Visual word ambiguity. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2010.

A. Vedaldi and A. Zisserman. Efficient additive kernels via explicit feature maps. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

A. Vedaldi and A. Zisserman. Sparse kernel approximations for efficient classification and detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

S. Vijayanarasimhan, P. Jain, and K. Grauman. Far-sighted active learning on a budget for image and video recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

S. Vijayanarasimhan, P. Jain, and K. Grauman. Hashing hyperplane queries to near points with applications to large-scale active learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2013.

V. Vural and J. G. Dy. A hierarchical method for multi-class support vector machines. In *International Conference on Machine Learning (ICML)*, 2004.

C. Wah and S. Belongie. Attribute-based detection of unfamiliar classes with humans in the loop. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

C. Wah, S. Branson, P. Perona, and S. Belongie. Multiclass recognition and part localization with humans in the loop. In *IEEE International Conference of Computer Vision (ICCV)*, 2011.

G. Wang, D. Hoiem, and D. Forsyth. Learning image similarity from Flickr groups using stochastic intersection kernel machines. In *IEEE International Conference of Computer Vision (ICCV)*, 2009.

G. Wang and D. Forsyth. Joint learning of visual attributes, object classes and visual saliency. In *IEEE International Conference of Computer Vision (ICCV)*, 2009.

J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

Y. Wang and G. Mori. A discriminative latent model of object classes and attributes. In *European Conference of Computer Vision (ECCV)*, 2010.

K. Weinberger, A. Dasgupta, J. Langford, A. Smola, and J. Attenberg. Feature hashing for large scale multitask learning. In *International Conference on Machine Learning (ICML)*, 2009.

K. Weinberger and O. Chapelle. Large margin taxonomy embedding for document categorization. In *Advances in Neural Information Processing Systems (NIPS)*, 2008.

J. Weston and C. Watkins. Support vector machines for multi-class pattern recognition. In *European Symposium On Artificial Neural Networks (ESANN)*, 1999.

J. Weston, O. Chapelle, A. Elisseeff, B. Schölkopf, and V. Vapnik. Kernel dependency estimation. In *Advances in Neural Information Processing Systems (NIPS)*, 2002.

J. Weston, S. Bengio, and N. Usunier. Large scale image annotation: Learning to rank with joint word-image embeddings. *European Conference of Machine Learning (ECML)*, 2010.

J. Weston, S. Bengio, and N. Usunier. Wsabie: Scaling up to large vocabulary image annotation. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2011.

J. Winn, A. Criminisi, and T. Minka. Object categorization by learned universal visual dictionary. In *IEEE International Conference on Computer Vision (ICCV)*, 2005.

J. Xu, T. Liu, M. Lu, H. Li, and W. Ma. Directly optimizing evaluation measures in learning to rank. In *ACM Special Interest Group on Information Retrieval (SIGIR)*, 2008.

J. Yang, K. Yu, and T. Huang. Efficient highly over-complete sparse coding using a mixture model. In *European Conference of Computer Vision (ECCV)*, 2010.

J. Yang, K. Yu, Y. Gong, and T. S. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

B. Yao, X. Jiang, A. Khosla, A. L. Lin, L. J. Guibas, and F.-F. Li. Human action recognition by learning bases of action attributes and parts. In *IEEE International Conference of Computer Vision (ICCV)*, 2011.

F. Yu, L. Cao, R. Feris, J. Smith, and S.-F. Chang. Designing category-level attributes for discriminative visual recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Portland, OR, June 2013.

X. Yu and Y. Aloimonos. Attribute-based transfer learning for object categorization with zero or one training example. In *European Conference of Computer Vision (ECCV)*, 2010.

Y. Yue, T. Finley, F. Radlinski, and T. Joachims. A support vector method for optimizing average precision. In *ACM Special Interest Group on Information Retrieval (SIGIR)*, 2007.

B. Zhao, L. Fei-Fei, and E. Xing. Large-scale category structure aware image categorization. In *Advances in Neural Information Processing Systems (NIPS)*, 2011.

Z. Zhou, K. Yu, T. Zhang, and T. Huang. Image classification using super-vector coding of local image descriptors. In *European Conference of Computer Vision (ECCV)*, 2010.