LARGE SCALE VISUAL RECOGNITION

JIA DENG

A DISSERTATION
PRESENTED TO THE FACULTY
OF PRINCETON UNIVERSITY
IN CANDIDACY FOR THE DEGREE
OF DOCTOR OF PHILOSOPHY

RECOMMENDED FOR ACCEPTANCE
BY THE DEPARTMENT OF
COMPUTER SCIENCE
ADVISER: FEI-FEI LI

JUNE 2012

Abstract

Visual recognition remains one of the grand goals of artificial intelligence research. One major challenge is endowing machines with human ability to recognize tens of thousands of categories. Moving beyond previous work that is mostly focused on hundreds of categories, we make progress toward human scale visual recognition. Specifically, our contributions are as follows:

First, we have constructed "ImageNet," a large scale image ontology. The Fall 2011 version consists of 22 thousand categories and 14 million images; it depicts each category by an average of 650 images collected from the Internet and verified by multiple humans. To the best of our knowledge this is currently the largest human-verified dataset in terms of both the number of categories and the number of images. Given the large amount of human effort required, the traditional approach to dataset collection, involving in-house annotation by a small number of human subjects, becomes infeasible. In this dissertation we describe how ImageNet has been built through quality controlled, cost effective, large scale online crowdsourcing.

Next, we use ImageNet to conduct the first benchmarking study of state of the art recognition algorithms at the human scale. By experimenting on 10 thousand categories, we discover that the previous state of the art performance is still low (6.4%). We further observe that the confusion among categories is hierarchically structured at large scale, a key insight that leads to our subsequent contributions.

Third, we study how to efficiently classify tens of thousands of categories by exploiting the structure of visual confusion among categories. We propose a novel learning technique that scales logarithmically with the number of classes in both training and testing, improving both accuracy and efficiency of the previous state of the art while reducing training time by 31 fold on 10 thousand classes.

Fourth, we consider the problem of retrieving semantically similar images from a large database, a problem closely related to classification. We propose an indexing approach that exploits the hierarchical structure between categories. Experiments demonstrate that our approach is more efficient, scalable, and accurate than previous work. In particular, our indexing technique achieves close to 90% of the accuracy of brute force with a 1,000 times speedup.

Finally, further exploiting the hierarchy, we show how to select the appropriate level of specificity to guarantee an arbitrary classification accuracy. We propose an algorithm that is provably optimal under mild conditions and demonstrate its effectiveness on classifying 10 thousand classes. Experiments show that our algorithm guarantees a 90% accuracy while giving informative answers 83% of the time. This holds promise toward a practical large scale recognition system.

Acknowledgements

First and foremost, I wish to thank my advisers Professor Fei-Fei Li and Professor Kai Li for their unfailing support and generous mentorship. It has been a tremendous inspiration to work with them. I am also honored to have Professor Szymon Rusinkiewicz, Professor David Blei, and Professor Adam Finkelstein on my dissertation committee.

Many thanks to my wonderful collaborators (in alphabetical order): Professor Alex Berg, Wei Dong, Jonathan Krause, Sanjeev Satheesh, and Hao Su.

I thank my colleagues, Barry Chai, Jia Li, Juan Carlos Niebles, Olga Russakovsky, Min Sun, Zhe Wang, and Bangpeng Yao (in alphabetical order) for their help and support.

This work is partially supported by NSF CAREER grant (IIS-0845230), the DARPA CSSG grant, a Google research award, NSF grant 0849512, Intel Research Council, and Gigascale Systems Research Center.

The materials of some chapters have been published and presented in the following conferences:

- Chapter 2: Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, Li Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2009.*

- Chapter 3: Jia Deng, Alex Berg, Kai Li, Li Fei-Fei. What does classifying more than 10,000 image categories tell us? *European Conference on Computer Vision (ECCV), 2010.*

- Chapter 4: Jia Deng, Sanjeev Satheesh, Alex Berg, Li Fei-Fei. Fast and Balanced: Efficient Label Tree Learning for Large Scale Object Recognition. *Neural Information Processing Systems (NIPS), 2011.*

- Chapter 5: Jia Deng, Alex Berg, Li Fei-Fei. Hierarchical Semantic Indexing for Large Scale Image Retrieval. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2011.*

- Chapter 6: Jia Deng, Jonathan Krause, Alex Berg, Li Fei-Fei. Hedging Your Bets: Optimizing Accuracy-Specificity Trade-offs in Large Scale Visual Recognition. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012.*

To my parents

Contents

Abstract iii

Acknowledgements v

1 Introduction **1**

1.1 Large Scale Visual Recognition 1

1.2 Challenges 5

1.2.1 Data 5

1.2.2 Learning algorithms 6

1.2.3 Feature representations 7

1.3 Background and Related Work 8

1.3.1 Visual recognition 8

1.3.2 Large Scale Labeled Datasets 10

1.3.3 Large Scale Learning 11

1.4 Contributions 12

1.4.1 Constructing ImageNet 12

1.4.2 Benchmarking with ImageNet 13

1.4.3 Efficient Large Scale Classification 14

1.4.4 Hierarchy-Aware Large Scale Retrieval 14

1.4.5 Multi-level Classification With Accuracy Guarantees 15

2 Constructing ImageNet **16**

2.1 Introduction 16
2.2 Properties of ImageNet 19
2.2.1 Scale 19
2.2.2 Hierarchy 20
2.2.3 Accuracy 20
2.2.4 Diversity 25
2.3 Related Work 25
2.3.1 Small Datasets 25
2.3.2 TinyImage Dataset 26
2.3.3 ESP Dataset 26
2.3.4 LabelMe and LotusHill Datasets 27
2.4 Construction Approach 29
2.4.1 Collecting Candidate Images 29
2.4.2 Verifying Candidate Images through Crowdsourcing 30
2.5 Discussion 35
2.5.1 Expanding ImageNet 35
2.5.2 Exploiting ImageNet 36
2.6 Summary 37

3 Benchmarking with ImageNet **38**

3.1 Introduction 38
3.2 Related Work 40
3.3 Approach 41
3.3.1 Datasets 41
3.3.2 Methodology 42
3.4 Results 44
3.4.1 Computation Matters 44
3.4.2 Size Matters 45

3.4.3 Density Matters 50
3.4.4 Hierarchy Matters 53
3.4.5 Experimental Details 55
3.5 Summary 56

4 Efficient Large Scale Classification 57

4.1 Introduction 57
4.2 Related Work 58
4.3 Label Tree and Label Tree Learning by Bengio et al. 59
4.4 New Label Tree Learning 61
4.4.1 Main Optimization 62
4.4.2 Learning Classifier Weights w Given Partitions P 64
4.4.3 Determining Partitions P Given Classifier Weights w 65
4.4.4 Summary of Algorithm 67
4.4.5 Efficiency Constrained Formulations 68
4.5 Experiments 69
4.6 Summary 74

5 Hierarchy-Aware Large Scale Retrieval 75

5.1 Introduction 75
5.2 Related Work 78
5.3 Approach 79
5.3.1 Exploiting Hierarchy 79
5.3.2 Efficient indexing 83
5.4 Experiments 93
5.4.1 Datasets and Evaluation Criteria 93
5.4.2 Hierarchical Retrieval 95
5.4.3 Flat Retrieval 97

5.4.4 Cross-Category Generalization 99

5.4.5 Indexing Efficiency 99

5.5 Summary 101

6 Multi-level Classification with Accuracy Guarantees 102

6.1 Introduction 102

6.2 Related Work 104

6.3 Formulation 105

6.4 The DARTS Algorithm 107

6.4.1 Maximizing the Lagrange Function 108

6.4.2 Optimality of DARTS 111

6.5 Experiments 119

6.5.1 Datasets. 119

6.5.2 Implementation. 120

6.5.3 Results on ILSVRC65 122

6.5.4 Results on ILSVRC1K and ImageNet10K 125

6.5.5 Zero-shot Recognition. 125

6.6 Summary 129

7 Conclusion and Future Work 130

7.1 Contributions 130

7.1.1 Constructing ImageNet 130

7.1.2 Benchmarking with ImageNet 131

7.1.3 Efficient Large Scale Classification 131

7.1.4 Hierarchy-Aware Large Scale Retrieval 131

7.1.5 Multi-level Classification with Accuracy Guarantees 132

7.2 Future Work 132

7.2.1 Image Representation 132

7.2.2 Crowdsourcing 133
7.2.3 Systems 133
7.2.4 Large Scale Image Parsing 134

Chapter 1

Introduction

1.1 Large Scale Visual Recognition

Visual recognition is one of the cornerstone functionalities for humans — it is so important, in fact, that nature has devoted more than 50% of neurons in the human brain to this task [1]. Recognition takes a variety of forms, such as categorization (“name the object”), localization (“find the object”), segmentation (“outline the contour of the object”), and retrieval of similar objects (“find similar objects”). The essence of all these tasks is translating 2D pixels into high level semantics. One of the grand goals of artificial intelligence is endowing machines with the same ability. This will lead to numerous applications ranging from personal robotics, image search, security, and visual assistance to digital field guides.

Despite decades of research, visual recognition remains difficult for computers. There are two main reasons. First, visual signals are notoriously complex. Each pixel in an image is the result of highly complicated physical processes. Somehow humans can effectively deal with this complexity—humans can recognize the same object or the same category of objects robust to changes in scale, location, lighting, occlusion, background clutter, and view points. Much of the traditional vision research has been concerned with discovering the physical, biological, computational principles underlying this recognition process. The second difficulty of visual recognition arises from a basic fact of high level human intelligence—the semantic space humans use to describe the visual world is extremely large. Psychologists have postulated that there are around 30 thousand visual categories [2]. Photos on flickrs have more than 3.5 million unique tags [3]. Indeed, even the most conservative estimates would point to at least tens of thousands of categories.

Large scale visual recognition, the focus of this dissertation, aims to address the very challenge of the large semantic space, an unchartered territory in previous vision research. It poses the question of how to emulate humans' ability of recognizing tens of thousands of categories. This stands in contrast with previous vision research that has been focused on at most a couple hundreds of categories [4, 5]. Figure. 1.1 and Figure. 1.2 illustrate the differences of the scale of recognition between previous research and the work this dissertation presents by comparing the categories used in Caltech101 [4], a widely used dataset in previous work, and ImageNet, a new large scale labeled dataset introduced in this dissertation that has 22K categories.

Another significant difference from small scale recognition is that the structures between the visual categories become important. The categories are related to each other in various ways. One of the most significant relations is the "is-a" relation. A "German Shepherd" is type of "dog," and also a type of "animal." This gives rise to a semantic hierarchy consisting of tens of thousands of categories, where some classes are more related to each other than others. In other words, large scale recognition deals with the semantic space in a much higher "resolution," that is, containing a large number of inter-related, "subordinate" categories. As a result, large scale recognition distinguishes itself from traditional recognition that studies a much "coarser" sampling of the semantic space and focuses on a small set of "flat," basic categories. Consider, for example, the PASCAL VOC [6, 7] challenge. As one of the most widely used benchmark for smaller scale recognition, it tests performance on 20 classes such as "bird," "dog," "cat," "car," etc., whereas large scale visual recognition deals with hundreds of sub-categories for "bird" alone.

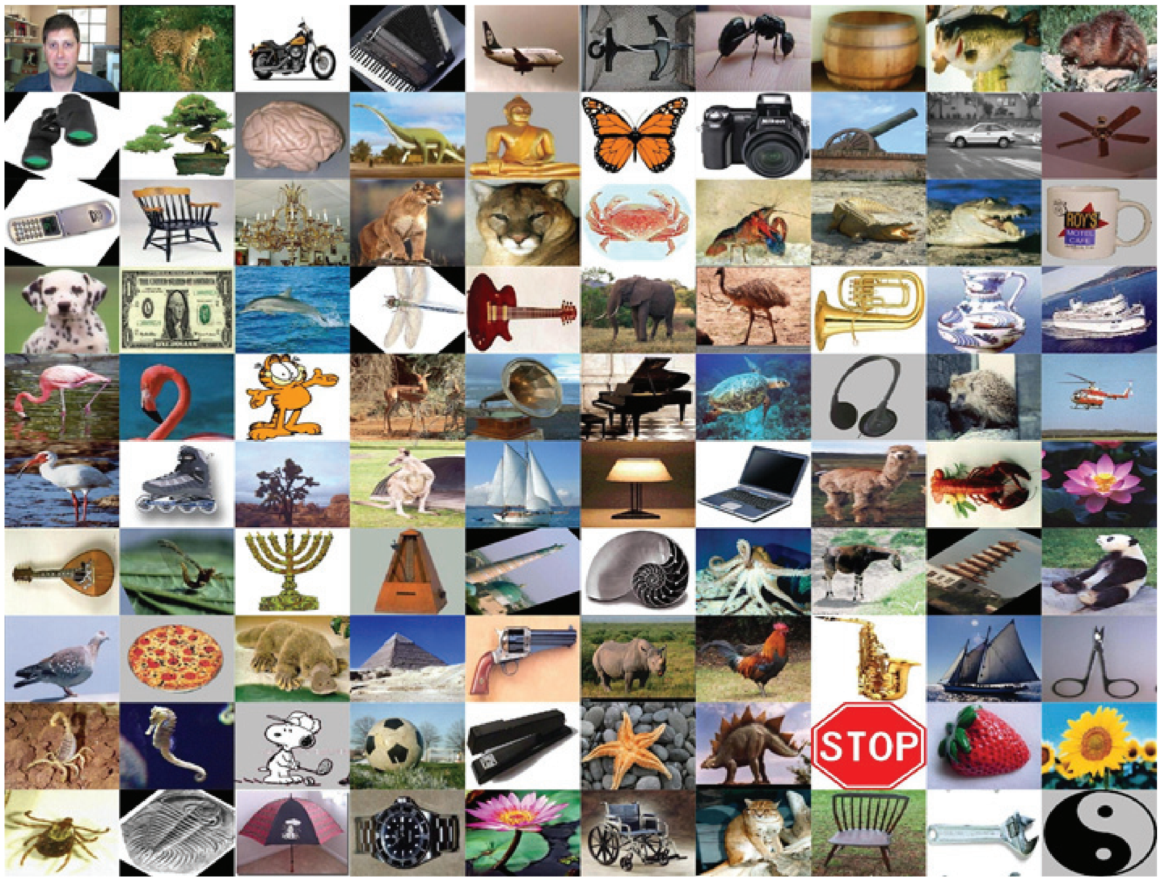


Figure 1.1: Visualization of the categories in Caltech101 [4], a widely used dataset by previous work in recognition.

Figure 1.2: Visualization of the 22K categories in ImageNet, a large scale labeled dataset introduced in this dissertation.

The enormous potential for useful applications and its unique characteristics in *scale* and *structure* makes large scale visual recognition an important and interesting problem.

1.2 Challenges

The key to solving the visual recognition problem is seeking a computational model that is capable of mapping pixels into semantics, in particular, large scale, fine-grained semantics.

The prevailing paradigm in obtaining such a model has been through *machine learning*. Visual recognition turns out to be an extremely complex process, one that has evaded most attempts to describe it as a simple set of rules. The promise of machine learning lies in its ability to handle such complexity by *learning from labeled data*, by generalizing using specific examples of mappings from pixels to semantics. Computer vision researchers' effort is still critical in designing models and feature representations by studying the underlying principles of vision, but a substantial amount of heavy-lifting is done by the data and the learning algorithms.

In this section, we discuss the challenges of large scale recognition in the machine learning paradigm. In particular, we focus on the key ingredients: data, learning algorithms, and feature representations.

1.2.1 Data

Data, in particular annotated data, is a critical component for machine learning. At the extreme, if we store every possible image in the universe as well as its annotations, then we can simply "look up" the answer to perform recognition. Of course this is unrealistic, but the hope is that by having sufficiently large labeled datasets, highly complex problems such as visual recognition would become much easier since, intuitively, we would be able to find very similar examples to the image we wish to recognize. The effectiveness of large, annotated datasets has been well demonstrated, especially in speech recognition and machine translation [8]. In the vision domain, the first breakthrough in recognition on the task of face detection came about only when a large amount of well labeled data became available [9, 10].

The information revolution has brought about an exponential growth of raw image data. There are more than 6 billion photos uploaded on Flickr [11] and 2 billion *every month* on Faceoook [12]. In addition, these photos often come with tags provided by users. This opens up great opportunities for us to attack large scale visual recognition using a massive amount of web data. The challenge is then how to harvest the data from the web and annotate it with high quality labels.

1.2.2 Learning algorithms

The goal of a learning algorithm is to produce a model from existing data in a way that generalizes to unseen data. For large scale recognition, the challenges in learning arise from the scale as well as the structure.

With regard to scale, the question is how to effectively process a large number of examples and classes. With millions of examples, traditional machine learning algorithms break down in two ways: first, the dataset cannot fit into memory any more, especially with high dimensional image feature representations; second, there can be so many examples that it can be prohibitively expensive to even go through the data once. In this setting, the learning algorithm is no longer bound by the number of examples, but by the computational resources. A fundamental question is how to make the most out of the almost unlimited supply of data under limited, distributed computational resources. With a large number of classes, traditional classification models can also be prohibitively expensive as most of them scale at least linearly with the number of classes. This calls for more efficient algorithms that scale sublinearly with respect to the number of classes.

With regard to structure in large scale recognition, the challenge is how to explore the hierarchical relations between categories. Large scale recognition involves tens of thousands of fine-grained classes that have much richer relationships than hundreds of weakly related classes traditionally dealt with. Traditional classification models treat the classes as a flat, mutually exclusive set. In contrast, for a large number of classes that form a hierarchy, this flat view of classes is no longer appropriate. First of all, the hierarchy consists of many levels of abstractions. A German Shepherd is also a dog, an animal, and a living thing. These categories overlap with each other and are no longer mutually exclusive. Moreover, some misclassification errors can be less desirable than others–it would be much worse to classify a German Shepherd as a microwave than as a husky. In order to take into account the semantic hierarchy, we need new learning algorithms.

1.2.3 Feature representations

Recognizing tens of thousands of categories demands powerful, efficient image representations. The standard image representation has converged to the bag-of-words model with spatial pooling [13, 14, 15, 16]. However, it is unlikely that this generic representation will ultimately work for tens of thousands of categories. Discriminative signals for fine-grained categories could well be lost in this generic representation developed on small-scale, "sparse" sets of classes. For example, the most effective features for classifying flowers may be very different from those for classifying cars. On the other hand, categories are related and some features can be shared among many categories. This poses the question of how to go about creating the representations. Is there a unified representation and learning mechanism that can handle all fine-grained classes? Or should we handcraft highly specialized, domain-specific features for many different categories? These questions continue to be explored in large scale recognition.

1.3 Background and Related Work

1.3.1 Visual recognition

Recognition has been an active research area of computer vision. The ultimate goal is to be able to fully understand every pixel of an image in terms of high level semantics, e.g., the scene, the objects, and the spatial, functional relations between them. This task, however, can be too difficult to tackle as a whole. Instead, it is decomposed into more manageable problems, such as classification (identify the class of the object), detection (localize the object), segmentation (assign every pixel an object label), etc. In this section, we will focus on the classification task, because it is a basic task that can serve as a building block for other tasks — for example, the detection task is typically solved as repeated classification of sliding windows in an image.

Given an input image with a dominant foreground object, the image classification task seeks to predict the class of the object. This is the standard multiclass classification problem in machine learning. Often the task is also called "image categorization" or "object categorization" and we will use these terms interchangeably.

The pipeline for image classification has converged to bag-of-words models (BoW) [17] with quantized local descriptors [13, 14, 15] and support vector machines [14, 18] as basic techniques. Figure 1.3 illustrates a typical pipeline. First, densely sampled local patches are extracted on the images. Each patch is described by a feature vector (descriptor). Typically the descriptor encodes histograms of gradient orientations, e.g., HOG [19] or SIFT [13]. Second, each descriptor is assigned to a visual word based on a codebook, a predetermined set of descriptors typically obtained by clustering many descriptors through KMeans. Third, given the visual words, any region of the image, including the full image itself, can be described by a histogram of the visual words. The image is typically represented by a concatenation of the histograms from a predetermined, fixed set of regions. A common choice of the regions is the spatial pyramid [15], which recursively partitions the

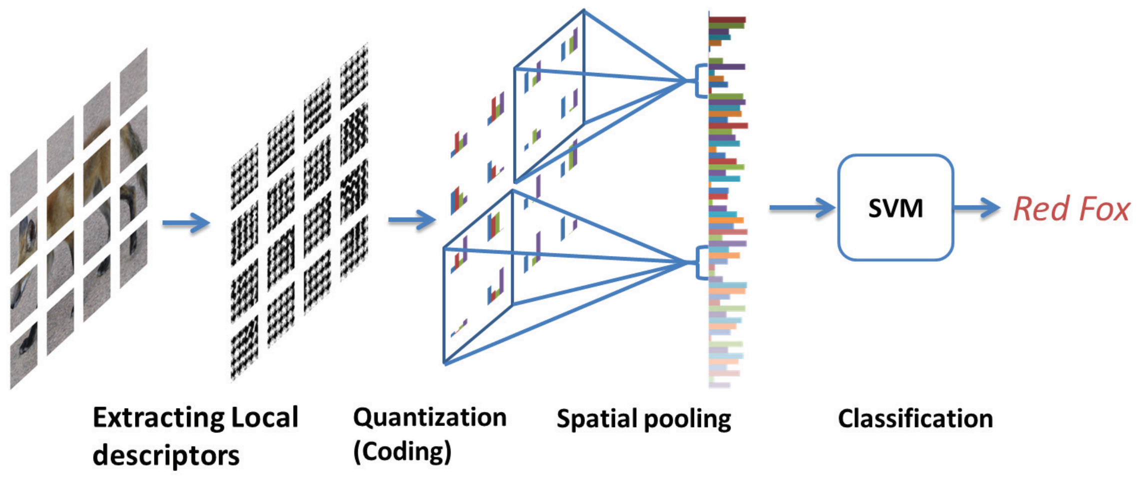


Figure 1.3: Illustration of the standard classification pipeline.

image into 2×2 grids. Finally, a multiclass classifier is learned using this representation. In the example of Figure 1.3, the classifier predicts the correct class label “red fox.” Note that the quantization and histogram forming steps are also referred to as “coding” or “pooling” in the literature. Recent developments have mostly centered on introducing variants of these two steps, but the basic elements remain the same [16, 20, 21].

A popular choice of the multiclass classifier is a combination of binary Support Vector Machine (SVM) classifiers. For each class, a “one-versus-all” classifier is learned to discriminate the class from all other classes. To classify a new image, we evaluate all one-versus-all classifiers and then predict the class with the highest classification score. To evaluate the performance, we measure the classification error by the percentage of misclassification. This standard evaluation considers the “flat” case: the classes are assumed to be mutually exclusive and the multiclass performance is evaluated by the uniform 0-1 loss, i.e., confusion between any two classes incurs a penalty of 1 and zero otherwise.

Although the majority of work in image classification only considers the “flat” case, a considerable amount of work utilizes hierarchies as well [22, 23, 24, 25]. Such works can be divided into two groups. The first group uses hierarchy to improve the performance of flat multiclass classifiers at the leaf nodes [26, 27, 28, 25, 29, 30]. Typically constructed in an automatic fashion, the hierarchy does not necessarily align with the semantic hierarchy. The idea is that a hierarchy can be employed to efficiently narrow down the possible classes as well as to transfer knowledge (e.g., by sharing examples between closely related classes) so as to improve accuracy. The second group of work considers classification for both internal nodes and leaf nodes [24, 31, 23]. Hierarchy is used either to combine models [24], or to organize the classifiers [23, 31, 25]. Typically multiclass or binary classifiers are learned for each node. A test example can then be classified at all semantic levels, e.g., by sending it down the hierarchy to a leaf node. The research question is thus how to do that in the most accurate and efficient way.

1.3.2 Large Scale Labeled Datasets

Dataset collection has been an critical part of computer vision research. A number of well labeled small datasets (Caltech101/256 [4, 5], PASCAL [32, 6, 7, 33], etc.) have served as training and evaluation benchmarks for most of today's computer vision algorithms. Caltech101 was essentially the first widely adopted dataset for object categorization. It has 101 categories and 9146 images in total. The PASCAL VOC datasets are a series of datasets used for the PASCAL Visual Object Class Challenges. There are 20 classes. The number of total images gradually increased from 10K in 2007 to 30K images in 2011. Other notable datasets include the LabelMe dataset [34] and the LotusHill dataset [35], which provide 30K and 637K labeled and segmented images, respectively [34, 35]. Both datasets have around 200 categories. [1]

The TinyImage [36] dataset was the first effort to build a dataset with many images covering significantly more than 200 categories. It consists of 32x32 pixel versions of images collected by performing web queries for the nouns in the WordNet [37] hierarchy, without further verification of content. Each class in the dataset contains an average of 1,000 im-

[1]LabelMe has 4K unique object descriptions but only 183 categories with more than 30 images each.

ages, among which likely only 10-25% are accurately labeled images [36]. Although the TinyImage dataset has had success with certain applications such as non-parametric recognition based on nearest neighbor search [36], the high level of noise and the low resolution of the images nevertheless make it less suitable for general purpose algorithm development, training, and evaluation.

1.3.3 Large Scale Learning

As digital data continues to exponentially grow and as massive parallelism gradually becomes the norm in computing, dealing with large datasets in machine learning emerges as an increasingly relevant and active research front.

One challenge of large scale machine learning is handling a large number of examples that may not fit into memory and, even if they do, may be too expensive to process with traditional algorithms. Two somewhat orthogonal approaches have been proposed. One is to use online learning algorithms, which sequentially examine the data, one small chunk a time. An effective method is the stochastic gradient descent (SGD) algorithm that randomly picks one example a time, computes the gradient based on this one example, and updates the parameters to be learned. It has been shown to be especially suitable in the large scale setting [38]. The other approach is to partition the data into smaller chunks such that each chunk can fit into the memory of one computing node and then apply a distributed learning algorithm. One programming model for this scenario is the well known MapReduce framework [39]. It turns out that many existing learning algorithms can be expressed in a certain summation form and can thus be adopted in this framework [40]. In addition, the alternating direction method of multipliers (ADMM) is found to be well suited as a generic meta algorithm to parallelize a large family of convex optimization problems [41].

Another challenge of large scale learning is dealing with a large output space, in particular, a large number of classes for the classification problem. The core issue is reducing the computational cost of learning algorithms with a large number of classes. There has been relatively little work on this front. The majority of existing models are tree based, including Filter Tree [42], Conditional Probability Tree(CPT) [43], and Label Trees [26]. These models use tree structures to reduce the multiclass problems to a small number of binary problems so that the evaluation cost can be sublinear in the total number of classes.

1.4 Contributions

This dissertation contributes to large scale visual recognition by tackling some of the challenges elaborated in Section 1.2. On the data front, through crowdsourcing, we have constructed ImageNet, a large scale, richly structured image dataset with over 20,000 categories and over 14 million images, orders of magnitude larger than any previous human verified image dataset, both in terms of the number of images and the number of categories. It has over 800 registered users worldwide. Next, on the learning front, we investigate machine learning techniques on this large scale visual data. After conducting the first benchmark of state of the art approaches, we have proposed new methods that are highly scalable, computationally efficient, and category-structure-aware.

1.4.1 Constructing ImageNet

Previous datasets for visual recognition have at most a couple of hundreds of categories, far from the tens of thousands of categories one would encounter in the real world. We have constructed ImageNet, a large scale, labeled image database with 22 thousand visual categories and 14 million images, covering visual categories that range from living things, artifacts, people, and scenes to activities and events (Chapter 2). [2] We use WordNet [37] to organize the categories and link the concepts into a semantic hierarchy based on the "is a" relation provided by WordNet [37]. We have developed an automated system that constructs ImageNet in two steps: (1) collecting candidate images for each category from

[2]An earlier version of ImageNet has been presented in [44].

the Internet by querying search engines in multiple languages and (2) verifying each image through workers on Amazon Mechanical Turk [45]. To make the system cost effective and to ensure the annotation quality, we have proposed an algorithm that dynamically estimates the confidence of the current annotation and determines the number of workers necessary to guarantee quality.

ImageNet has since its initial release three years ago enjoyed over 800 registered users worldwide and served as the benchmark dataset for two large scale recognition competitions held in conjunction with PASCAL [46, 47]. In addition to large scale recognition, it has been used for transfer learning [48], image retrieval [49], computational linguistics [50], and human computer interaction [51].

1.4.2 Benchmarking with ImageNet

Benchmarking is critical for both developing and evaluating recognition algorithms. Traditionally object recognition algorithms were evaluated on datasets of at most hundreds of categories. We benchmarked state of the art algorithms on 10,184 categories in ImageNet, the first time recognition was studied at such a scale (Chapter 3). [3] The major findings include: (1) the previous state of the art performance is still very low on 10,184 classes (6.4%); (2) evaluation at a small scale cannot predict large scale performance; (3) visual similarity roughly aligns with semantic similarity; (4) a visualization of the confusion matrix exhibits hierarchical block patterns, revealing that the difficulty comes from distinguishing semantically similar, fine-grained subordinate classes; (5) online, low memory footprint, parallel machine learning algorithms are needed. These findings are significant in that they point to new research directions and are in fact the basis for my subsequent research on efficient, category-structure-aware large scale learning algorithms.

[3]This study was first presented in [52].

1.4.3 Efficient Large Scale Classification

One major challenge of classifying a large number of categories is computational efficiency. As the number of classes increases, conventional frameworks for training and testing become inefficient as the cost scales linearly with the number of categories. In Chapter 4, we propose a novel learning technique for label trees [26], a state of the art large scale classification model that has sublinear test time with respect to the number of classes. [4] Our new learning technique significantly improves the quality of the label trees learned in terms of both test efficiency and accuracy, as shown by experiments on more than 10,000 categories. Moreover, training costs are also substantially reduced, from $O(mK)$ to $O(m \log K)$ where m is the number of examples and K is the number of categories. A systems challenge is efficiently optimizing a tightly coupled learning objective on a large amount of data that cannot fit in memory. We overcome this difficulty by applying a recently developed distributed learning framework called parallelized stochastic gradient descent using Hadoop MapReduce.

1.4.4 Hierarchy-Aware Large Scale Retrieval

A closely related problem to classification is retrieval, or finding similar images given a query image. It is highly challenging on its own due to the difficulty of measuring high level semantic similarities and indexing a large number of images. In Chapter 5 we propose to represent images as probabilities over the semantic hierarchy through multi-class learning and measure image similarity using this representation. [5] We also develop a novel locality sensitive hashing technique [55] to enable efficient retrieval. The approach is highly scalable, in fact “embarrassingly parallel” by design such that it is perfectly suited for a cloud environment. Moreover, it is much faster than previous state of art methods in both learning and retrieval. Another highlight is that the system is able to retrieve images not only from

[4]This work was first presented in [53].

[5]An early version appeared in [54].

the same category as the query image, but also from semantically similar categories.

1.4.5 Multi-level Classification With Accuracy Guarantees

Equally important is exploiting the semantic structure. The image categories form a semantic hierarchy where internal nodes are unions of their leaf nodes. For example, a kangaroo is also a mammal, an animal, and a living thing. Motivated by this observation, we propose in Chapter 6 an "infallible" classifier that tries to be as specific as possible yet almost never makes mistakes. [6] The classifier can maintain an arbitrarily high level of accuracy by outputting internal nodes in the case of uncertainty at leaf nodes. We formulated this problem as maximizing information gain of classification while maintaining a fixed, arbitrarily small error rate. This is the first time that the problem of optimizing the accuracy-specificity trade-off on a semantic hierarchy has been introduced in large scale recognition. We proposed an algorithm that provably converges to an optimal solution under practical conditions. Experiments on more than 10,000 categories demonstrated that recognition can be highly accurate yet very informative, significantly outperforming baseline approaches. This paves the way to a practical large scale recognition system.

[6]This work is also presented in [56].

Chapter 2

Constructing ImageNet

2.1 Introduction

The digital era has brought with it an enormous explosion of data. According to the IDC [57] digital universe study, the world was projected to add 1.8 zettabytes (1.8 trillion gigabytes) in 2011 alone. For visual data, the latest estimations put a number of more than 6 billion photos on Flickr [11], a similar number of video clips on YouTube and an even larger number of images in the Google Image Search database. More sophisticated and robust models and algorithms can be proposed by exploiting these images, resulting in better applications for users to index, retrieve, organize and interact with this data. But exactly how such data can be utilized and organized is a problem yet to be solved. In this chapter, we introduce a new image database called "ImageNet," a large scale ontology of images. [1] We believe that it is a critical resource for developing algorithms for large scale recognition and large scale image retrieval, as well as for providing critical training and benchmarking data for such algorithms.

ImageNet uses the hierarchical structure of WordNet [37], a lexical database of English. Each meaningful concept in WordNet, possibly described by multiple words or word

[1]An earlier version was first presented in [44].

phrases, is called a “synonym set” or “synset.” There are around $80,000$ noun synsets in WordNet. In ImageNet, the goal is to provide on average 500–1,000 images to illustrate each synset that can be visualized. Images of each concept are quality-controlled and human-annotated as described in Section 2.4.2. ImageNet, therefore, aims to offer tens of millions of accurately labeled images. The current version of ImageNet consists of 21,841 synsets and 14,197,122 images. It covers a wide range of visual concepts including animal, plant, artifact, geological formations, activities, and materials. Figure 2.1 shows a snapshot of two branches of the mammal and vehicle hierarchies. The database is publicly available at `http://www.image-net.org`.

The rest of this chapter is organized as follows. We first describe the properties of ImageNet in Section 2.2 and then compare ImageNet with related datasets in Section 2.3. Section 2.4 describes how ImageNet is constructed by leveraging Amazon Mechanical Turk. This is followed by a discussion of future work in Section 2.5.

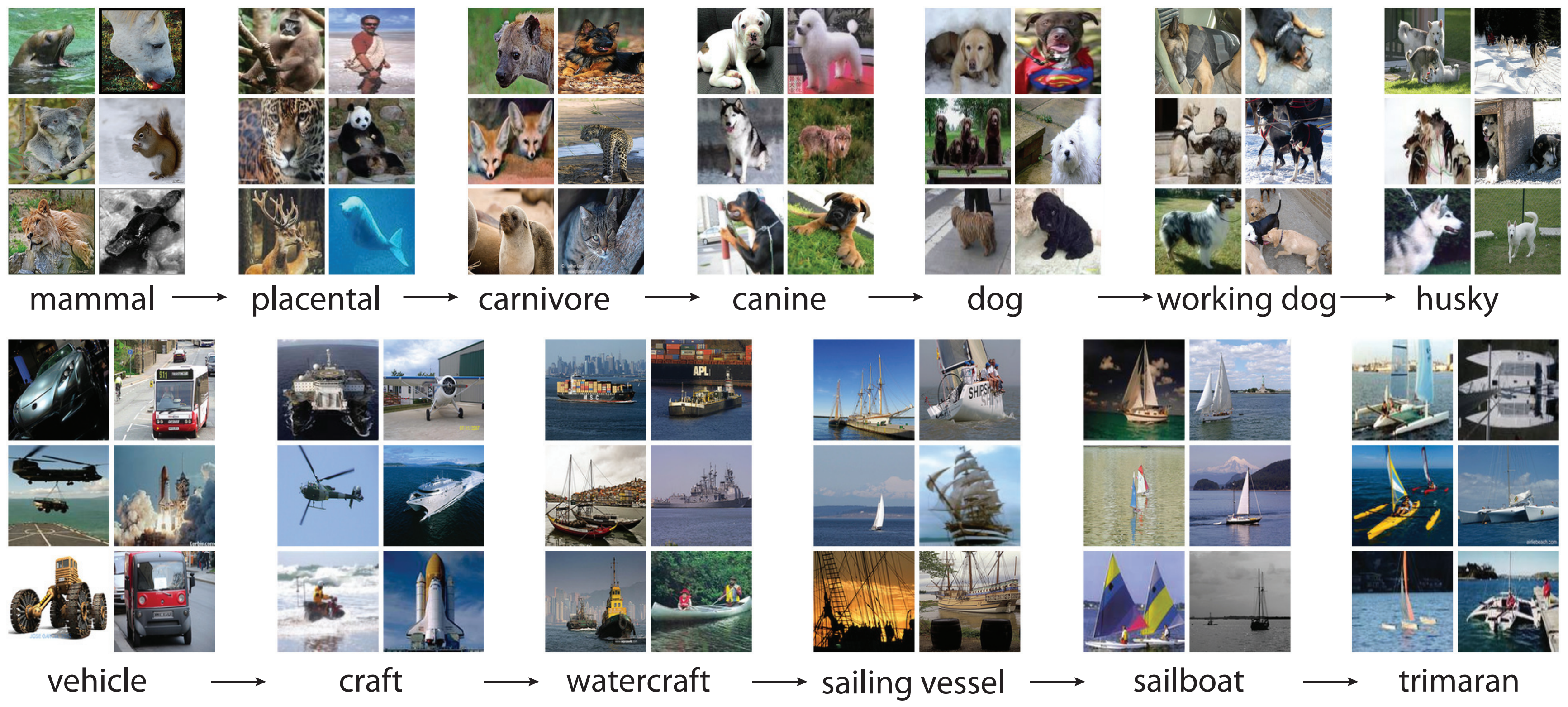


Figure 2.1: A snapshot of two root-to-leaf branches of ImageNet: the top row is from the mammal subtree; the bottom row is from the vehicle subtree. For each synset, 6 randomly sampled images are presented.

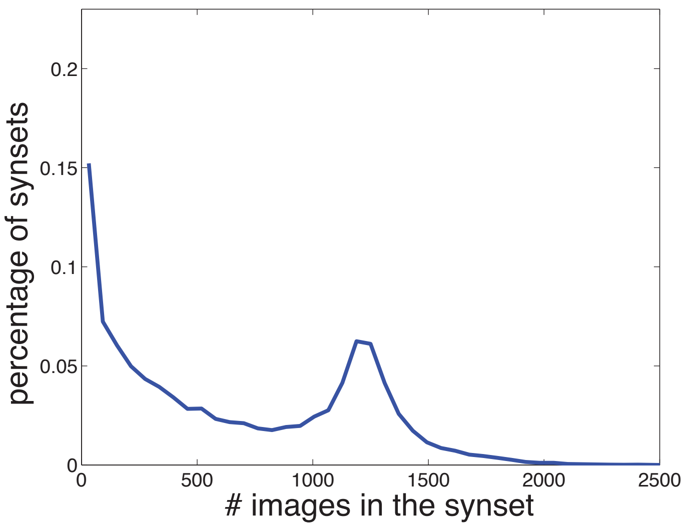


Figure 2.2: Histogram of number of images per synset in the Fall 2011 release of ImageNet. About 15% of the synsets have very few images. Over 50% synsets have more than 500 images.

2.2 Properties of ImageNet

In this section we discuss some of the most important properties of ImageNet, which is a large scale, hierarchically organized, accurate, and diverse image dataset.

2.2.1 Scale

ImageNet aims to provide a comprehensive and diverse coverage of the image world. The Fall 2011 version consists of a total of 14 million human verified images spread over $21,841$ categories. The scale of ImageNet is evidenced by its extensive coverage of the visual world that ranges from living things, artifacts, people, and scenes to activities and events. Table 2.2.1 lists the statistics of major subtrees. On average 650 images are collected for each synset. Figure 2.2 shows the distributions of the number of images per synset for the current ImageNet. Admittedly, about 20% of the synsets have very few images because either there are very few web images available, e.g., “vespertilian bat,” or the synset by definition is difficult to illustrate with images, e.g., “two-year-old horse.” To our knowledge this is currently the largest human verified image dataset available to the vision research community, in terms of the total number of images, number of images per category as well as the number of categories. It is claimed that the ESP game [58] has labeled a very large number of images, but only a subset of 60K images are publicly available.

2.2.2 Hierarchy

ImageNet organizes the different classes of images in a *densely populated* semantic hierarchy. The main asset of WordNet [37] lies in its semantic structure, i.e., its ontology of concepts. Similarly to WordNet, synsets of images in ImageNet are interlinked by several types of relations, the "is-a" relation being the most comprehensive and useful. The "is-a" relation forms a hierarchy of synsets, or more specifically a directed acyclic graph (DAG). Note that the DAG of WordNet is mostly a tree and we will use the term "tree," "DAG," and "hierarchy" interchangeably unless the difference between a tree and a DAG is significant in the context. Although one can map any dataset with category labels into a semantic hierarchy by using WordNet, the density of ImageNet is unmatched by others. For example, to our knowledge no existing vision dataset offers images of 147 dog categories. Figure 2.3 compares the "cat" and "cattle" subtrees of ImageNet and the ESP dataset [58]. We observe that ImageNet offers much denser and larger trees.

2.2.3 Accuracy

We would like to offer a accurately labeled dataset at all levels of the WordNet hierarchy. Figure 2.4 demonstrates the labeling precision on a total of 80 synsets with 500 or more images each, randomly sampled at different tree depths from the mammal and vehicle subtrees. A 99.7% precision is achieved on average. Achieving a high precision for all depths of the ImageNet tree is challenging because the lower in the hierarchy a synset is, the harder it is to classify, e.g., Siamese cat versus Burmese cat.

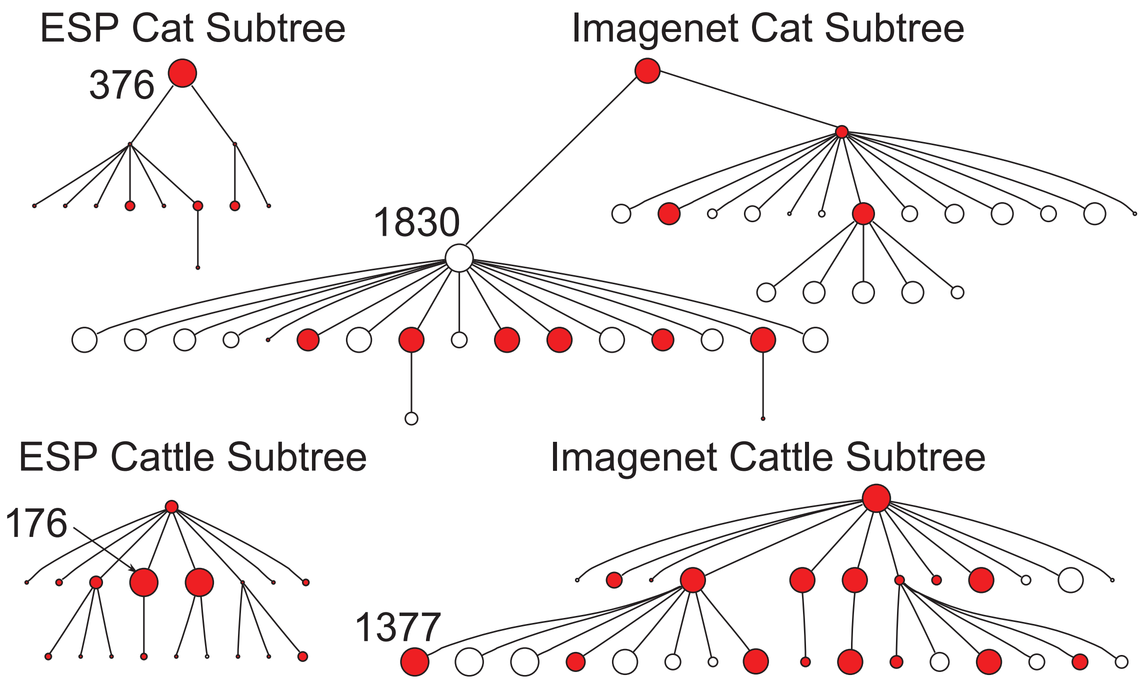


Figure 2.3: Comparison of the “cat” and “cattle” subtrees between ESP [58] and ImageNet. Within each tree, the size of a node is proportional to the number of images it contains. The number of images for the largest node is shown for each tree. Shared nodes between an ESP tree and an ImageNet tree are colored in red.

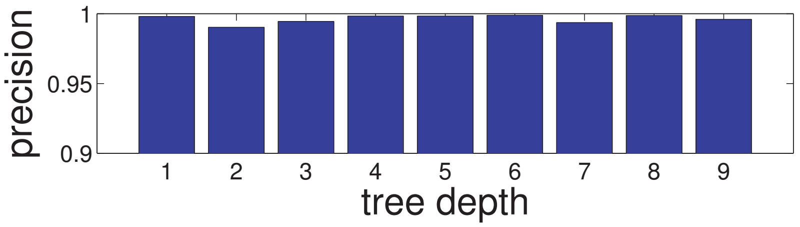


Figure 2.4: Percent of accurately labeled images at different tree depth levels in ImageNet. A total of 80 synsets are randomly sampled at every tree depth of the mammal and vehicle subtrees. An independent group of subjects verified the correctness of each of the images. An average of 99.7% precision is achieved for each synset.

subtree	# synsets	avg # of images per synset	total # of images
amphibian	94	590	55,510
animal	3,822	732	2,798,930
appliance	51	1,163	59,343
artifact	7,450	749	5,582,339
bird	856	948	812,069
covering	946	818	774,362
device	2,385	674	1,609,552
fabric	262	689	180,777
fish	566	494	279,775
flower	462	734	339,383
food	1,495	669	1,001,193
fruit	309	607	187,583
fungus	303	452	137,187
furniture	187	1,042	194,948
geological formation	151	838	126,567
invertebrate	728	572	416,832
mammal	1,138	821	934,450
musical instrument	157	891	139,899
person	21	1,152	24,208
plant	1,666	599	999,163
reptile	268	707	189,521
sport	166	1,207	200,402
structure	1,239	763	945,590
tool	316	551	174,271
tree	993	568	564,040
utensil	86	912	78,442
vegetable	176	764	134,518
vehicle	481	777	374,135

Table 2.1: Statistics of common subtrees in the Fall 2011 release of ImageNet. The subtrees listed are not mutually exclusive to each other.

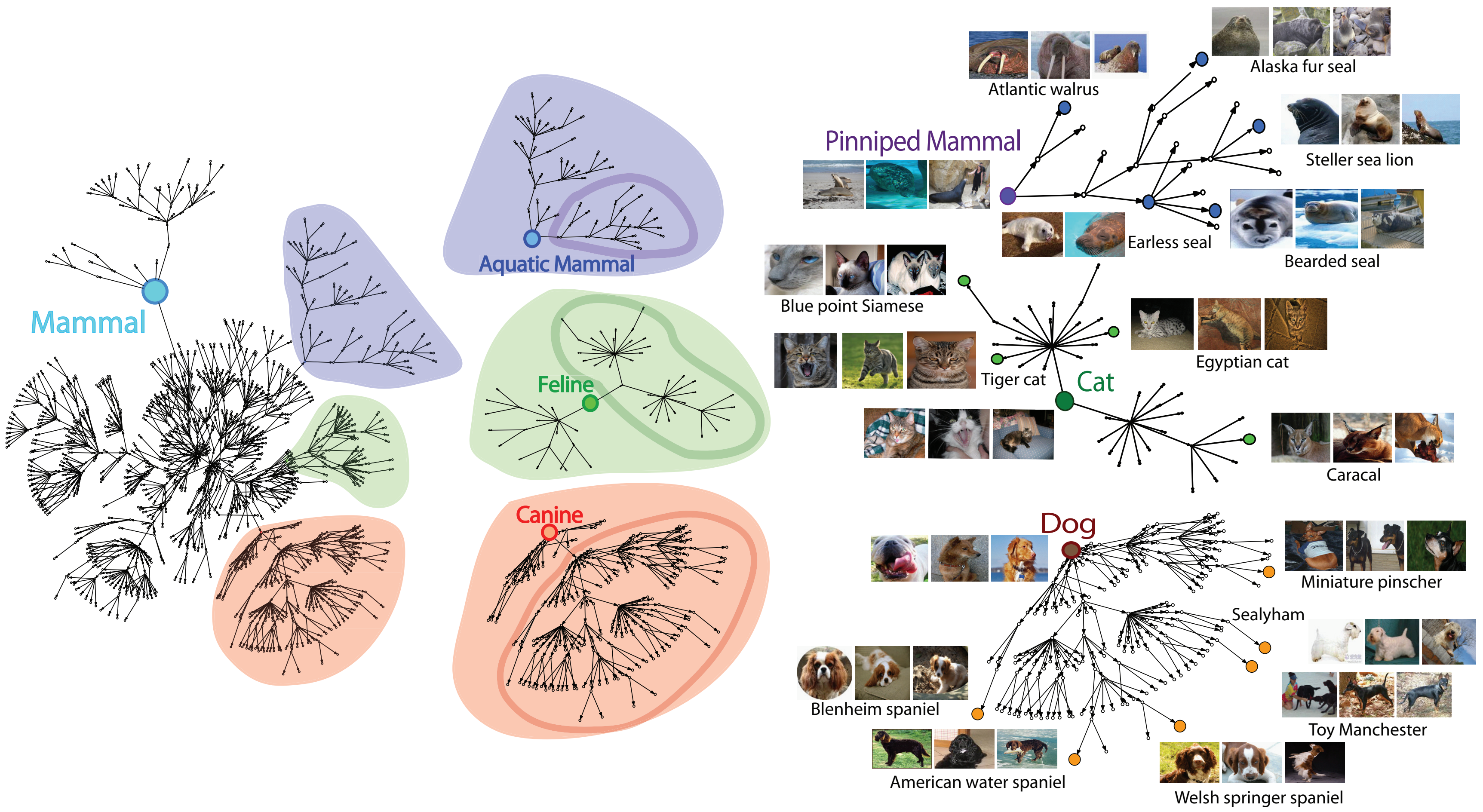


Figure 2.5: Visualization of the mammal hierarchy.

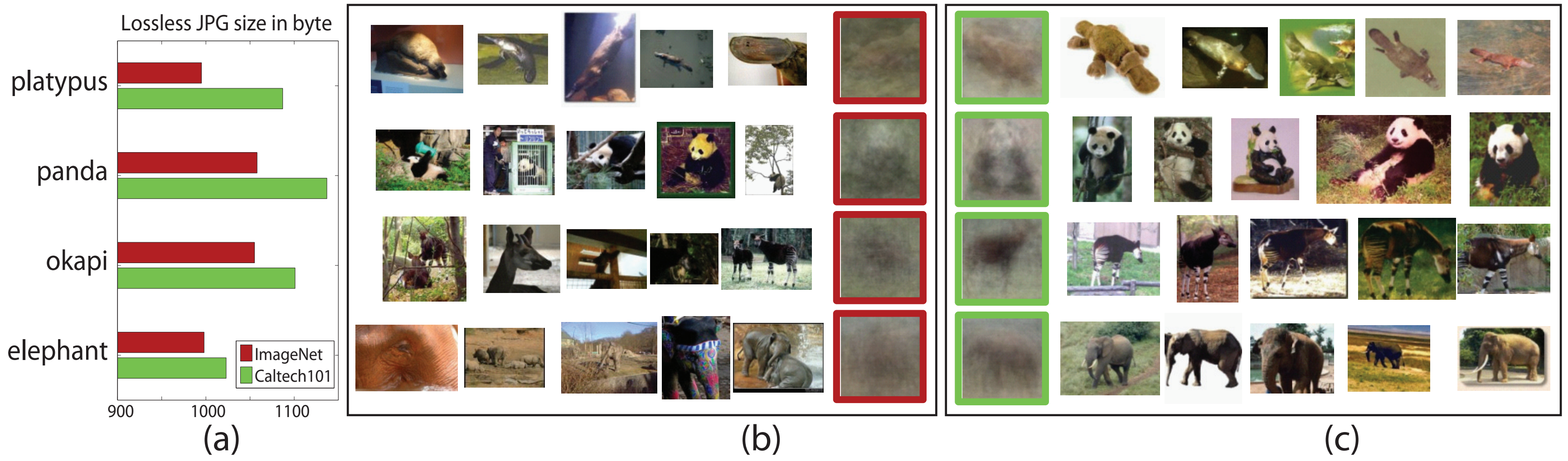


Figure 2.6: ImageNet provides diversified images. (a): Comparison of the lossless JPG file sizes of average images for four different synsets in ImageNet (the mammal subtree) and Caltech101. Average images are downsampled to 32×32 and sizes are measured in byte. A more diverse set of images results in a smaller lossless JPG file size. (b): Example images from ImageNet and average images for each synset indicated by (a). (c): Examples images from Caltech101 and average images. For each category shown, the average image is computed using all images from Caltech101 and an equal number of randomly sampled images from ImageNet.

2.2.4 Diversity

ImageNet is constructed with the goal that objects in images should have variable appearances, positions, view points, and poses as well as background clutter and occlusions. In an attempt to tackle the difficult problem of quantifying image diversity, we compute the average image of each synset and measure lossless JPG file size which reflects the amount of information in an image. Our idea is that a synset containing diverse images will result in a blurrier average image, the extreme being a gray image, whereas a synset with little diversity will result in a more structured, sharper average image. We therefore expect to see a smaller JPG file size of the average image of a more diverse synset. Figure 2.6 compares the image diversity in four randomly sampled synsets in Caltech101 [4] and the mammal subtree of ImageNet.

2.3 Related Work

In this section we compare ImageNet with other datasets. We focus our comparisons on datasets of generic objects. Special purpose datasets, such as FERET faces [59], Labeled faces in the Wild [60] and the Mammal Benchmark by Fink and Ullman [61] are not included. We summarize the main differences between ImageNet and related datasets in Table 2.2.

2.3.1 Small Datasets

A number of well labeled small datasets (Caltech101/256 [4, 5], MSRC [62], PASCAL [7] etc.) have served as training and evaluation benchmarks for most of today's computer vision algorithms. As computer vision research advances, larger and more challenging datasets are needed for the next generation of algorithms. The current ImageNet offers $20\times$ the number of categories, and $100\times$ the number of total images than these datasets.

	ImageNet	TinyImage	LabelMe	ESP	LotusHill
# Images (million)	14	80	0.03 [1]	0.07 [2]	0.6
# Classes (thousand)	22	80	4 [3]	23 [4]	0.2
Label Disambiguated	Y	Y	N	N	Y
Human Verified	Y	N	Y	Y	Y
Full Resolution	Y	N	Y	Y	Y
Segmented	N	N	Y	N	Y

[1] as of December 21, 2006.
[2] publicly available ones.
[3] unique descriptions as of December 21, 2006. There are 183 categories with at least 30 images.
[4] publicly available unique labels (without merging synonyms).

Table 2.2: Comparison of some of the properties of ImageNet versus TinyImage [36], LabelMe [34], ESP [58], and LotusHill [35]. ImageNet offers disambiguated labels, human verified annotations, full resolution images; it is also publicly available. ImageNet currently does not provide segmentation annotations.

2.3.2 TinyImage Dataset

TinyImage [36] is a dataset of 80 million 32×32 low resolution images, collected from the Internet by sending all words in WordNet as queries to image search engines. Each synset in the TinyImage dataset contains an average of 1056 images, among which 10-25% are estimated to be accurately labeled images [36]. Although the TinyImage dataset has had success with certain applications, the high level of noise and low resolution images make it less suitable for general purpose algorithm development, training, and evaluation. Compared to the TinyImage dataset, ImageNet contains high quality synsets ($\sim 99\%$ precision) and full resolution images with an average size of around 400×350.

2.3.3 ESP Dataset

The ESP dataset is acquired through an online game [58]. Two players independently propose labels to one image with the goal of matching as many words as possible in a certain time limit. Millions of images are labeled through this game, but its speeded nature also poses a major drawback. Rosch and Lloyd [63] have demonstrated that humans tend to label visual objects at an easily accessible semantic level termed as “basic level” (e.g., bird), as opposed to more specific level (“subordinate level,” e.g., sparrow), or more general level (“super-ordinate level,” e.g., vertebrate). Labels collected from the ESP game largely concentrate at the “basic level” of the semantic hierarchy as illustrated by the color bars in Figure 2.7. ImageNet, however, demonstrates a much more balanced distribution of images across the semantic hierarchy. Another critical difference between ESP and ImageNet is sense disambiguation. When human players input the word “bank,” it is unclear whether it means “a river bank” or a “financial institution.” At this large scale, disambiguation becomes a non-trivial task. Without it, the accuracy and usefulness of the ESP data could be affected. ImageNet, on the other hand, does not have this problem by construction, as detailed in Section 2.4.2. Lastly, most of the ESP dataset is not publicly available. Only 68K images and their labels can be accessed [64].

2.3.4 LabelMe and LotusHill Datasets

LabelMe [34] and the LotusHill dataset [35] provide 30K [2] and 637K labeled and segmented images, respectively. These two datasets provide complementary resources for the vision community compared to ImageNet. LabelMe has 4,210 unique object descriptions (among which 183 categories have at least 30 annotated examples each) and LotusHill has 200 object categories. In both datasets, the outlines and locations of objects are provided. ImageNet in its current form does not provide detailed object outlines (see potential extensions in Section 2.5.1), but the number of categories and the number of images per category already far exceeds these two datasets. In addition, images in these two datasets are largely uploaded or provided by users or researchers of the dataset, whereas ImageNet contains images crawled from the entire Internet. The LotusHill dataset is available for purchase whereas ImageNet is freely available.

[2]as of December 21, 2006.

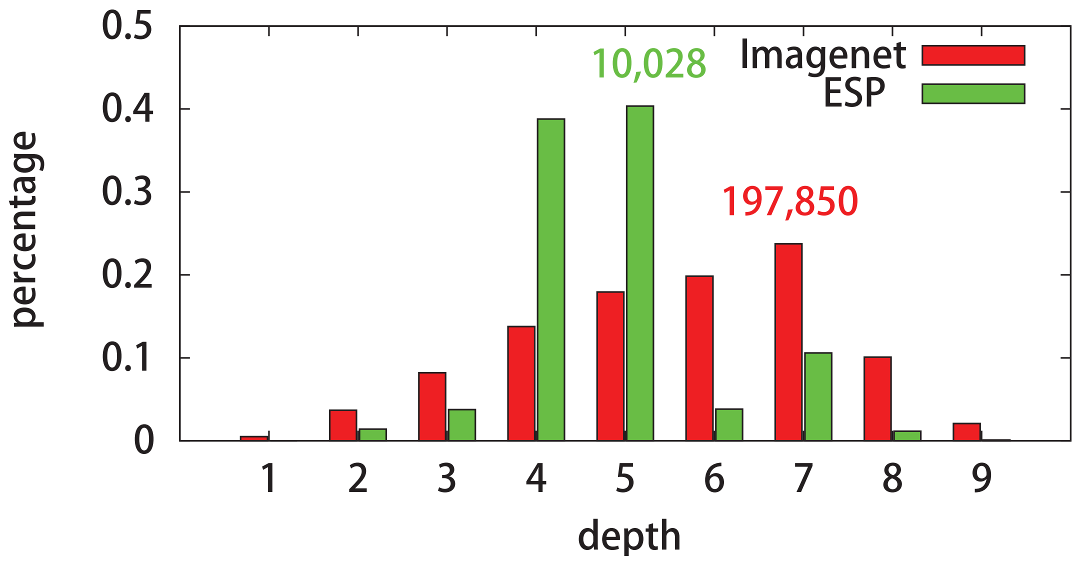


Figure 2.7: Comparison of the distribution of "mammal" labels over tree depth levels between ImageNet and ESP game. The x-axis is the depth in the WordNet hierarchy. The y-axis indicates the percentage of the labels at a certain depth. ImageNet demonstrates a much more balanced distribution, offering substantially more labels at deeper tree depth levels. The actual number of images corresponding to the highest bar is also given for each dataset.

2.4 Construction Approach

The intended scope of ImageNet made it infeasible for us to construct it using the traditional data collection method: annotating images by ourselves or by recruiting a small number of human subjects. Suppose we need 1,000 images per category for 30,000 categories. Further assume that to collect 1 verified image we need to screen 10 candidate images and that a human subject can work at a speed of 0.5 second per image without making mistakes. This would give a total of $1,000 \times 10 \times 30,000 \times 0.5$ seconds, i.e., 19 years of human time. This suggests that a more scalable solution is necessary.

In this section we describe how we construct ImageNet through large scale online crowdsourcing, shedding light on how the properties described in Section 2.2 can be ensured in this process.

2.4.1 Collecting Candidate Images

The first stage of the construction of ImageNet involves collecting candidate images for each synset. The average accuracy of image search results from the Internet is around 10% [36]. ImageNet aims to offer 500–1,000 accurately labeled images per synset. We therefore collect a large set of candidate images. After intra-synset duplicate removal, each synset has 16K images on average. This gives a total of 353 million candidate images for the Fall 211 release of ImageNet.

We collect candidate images from the Internet by querying several image search engines. For each synset, the queries are the set of WordNet synonyms. Search engines typically limit the number of images retrievable (in the order of a few hundred to a thousand). To obtain as many images as possible, we expand the query set by appending the queries with the word from parent synsets, if the same word appears in the gloss of the target synset. For example, when querying "whippet," according to WordNet's gloss a "small slender dog of greyhound type developed in England," we also use "whippet dog" and “whippet greyhound.”

To further enlarge and diversify the candidate pool, we translate the queries into other languages [65], including Chinese, Spanish, Dutch and Italian. We obtain accurate translations by WordNets in those languages [66, 67, 68, 69].

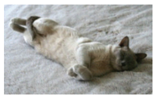
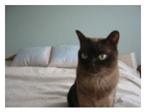
2.4.2 Verifying Candidate Images through Crowdsourcing

To collect a highly accurate dataset, we rely on humans to verify each candidate image collected in the previous step for a given synset. Traditionally researchers either verify the images themselves or recruit a small number of human subjects to help. At the scale of ImageNet this approach is no longer feasible, as our rough calculation earlier shows that it would take 19 human years.

Crowdsourcing is the process of completing tasks or solving problems through a distributed group of people. There are many types of crowdsourcing. One type is volunteer-based, such as the construction of Wikipedia where a large number of people volunteer to contribute content. Another type is paid microtasking, where people finish small tasks online for a small amount of money. A popular platform is Amazon Mechanical Turk (AMT) [45], an online market through which one can put up tasks for users to complete and to get paid. A third type of crowdsourcing is through ‘games with a purpose” [58], where players implicitly finish certain tasks through enjoyable game play.

For constructing ImageNet, we crowdsource through AMT because it has a large pool of global users on demand, whereas it can be difficult to accumulate unpaid volunteers or game players at a similar scale in a short time. Moreover, AMT has been used for labeling vision data [70] and has been shown to give good results with low cost.

In each of our labeling tasks, we present the users with a set of candidate images and the definition of the target synset (including a link to Wikipedia). We then ask the users to verify whether each image contains objects of the synset. We encourage users to select images regardless of occlusions, number of objects and clutter in the scene to ensure diver-

			
User 1	Y	Y	Y
User 2	N	Y	Y
User 3	N	Y	Y
User 4	Y	N	Y
User 5	Y	Y	Y
User 6	N	N	Y

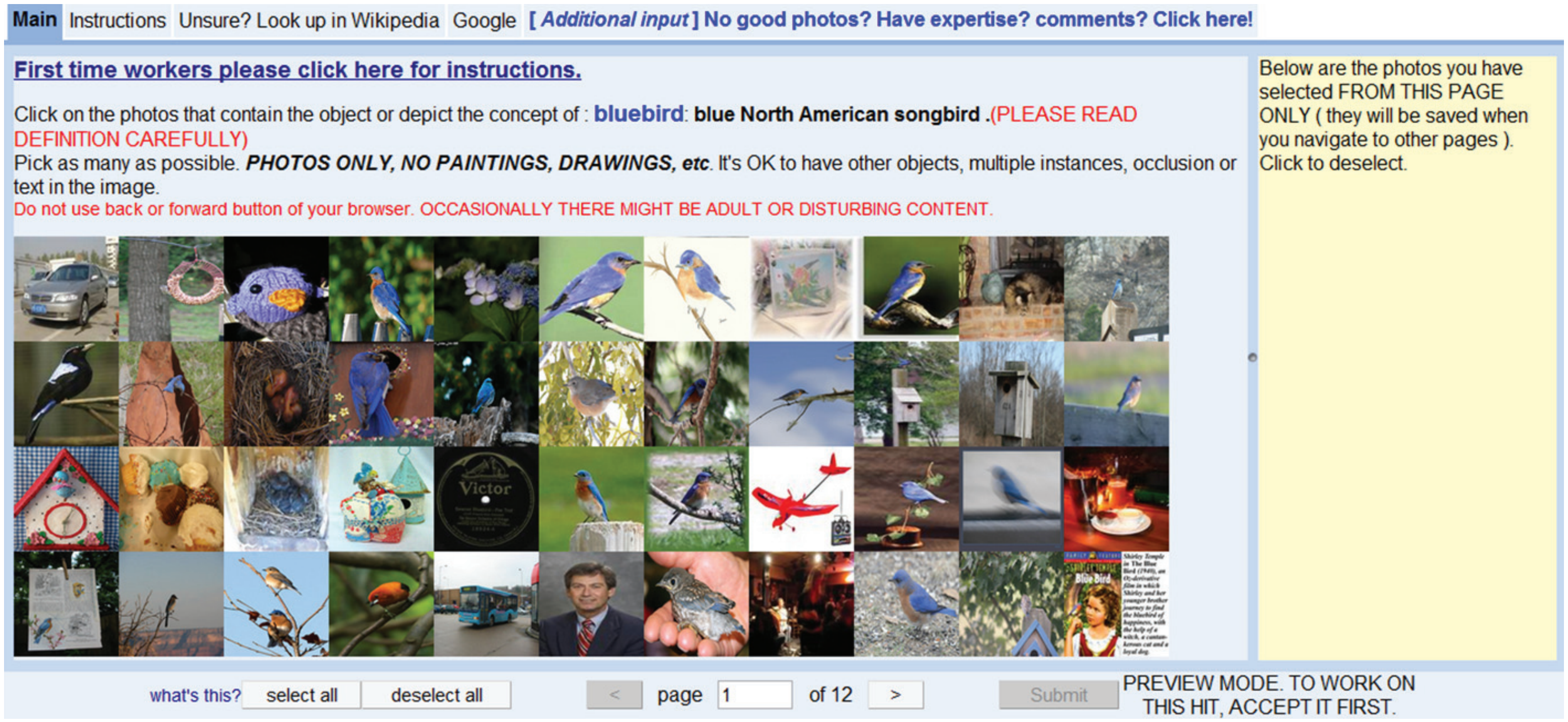
#Y	# N	Conf Cat	Conf BCat
0	1	0.07	0.23
1	0	0.85	0.69
1	1	0.46	0.49
2	0	0.97	0.83
0	2	0.02	0.12
3	0	0.99	0.90
2	1	0.85	0.68

Figure 2.8: Left: Is there a Burmese cat in the images? Six randomly sampled users have different answers. Right: The confidence score table for “Cat” and “Burmese cat.” More votes are needed to reach the same degree of confidence for “Burmese cat” images.

sity. Figure 2.9 shows our labeling interface. A user is to click on the images that contain a “bluebird.” Figure 2.10 shows a finished submission, where the selected images have been moved to the right panel.

While users are instructed to make accurate judgment, we need to set up a quality control system to ensure this accuracy. There are two issues to consider. First, human users make mistakes and not all users follow the instructions. Second, users do not always agree with each other, especially for more subtle or confusing synsets, typically at the deeper levels of the tree. Figure 2.8 (left) shows an example of how users’ judgments differ for “Burmese cat.”

The solution to these issues is to have multiple users independently label the same image. An image is considered positive only if it gets a convincing majority of the votes. We observe, however, that different categories require different levels of consensus among users. For example, while five users might be necessary for obtaining a good consensus on “Burmese cat” images, a much smaller number is needed for “cat” images. We develop a simple algorithm to dynamically determine the number of agreements needed for different categories of images. For each synset, we first randomly sample an initial subset of images. At least 10 users are asked to vote on each of these images. We then obtain a



Main | Instructions | Unsure? Look up in Wikipedia | Google | [*Additional input*] No good photos? Have expertise? comments? Click here!

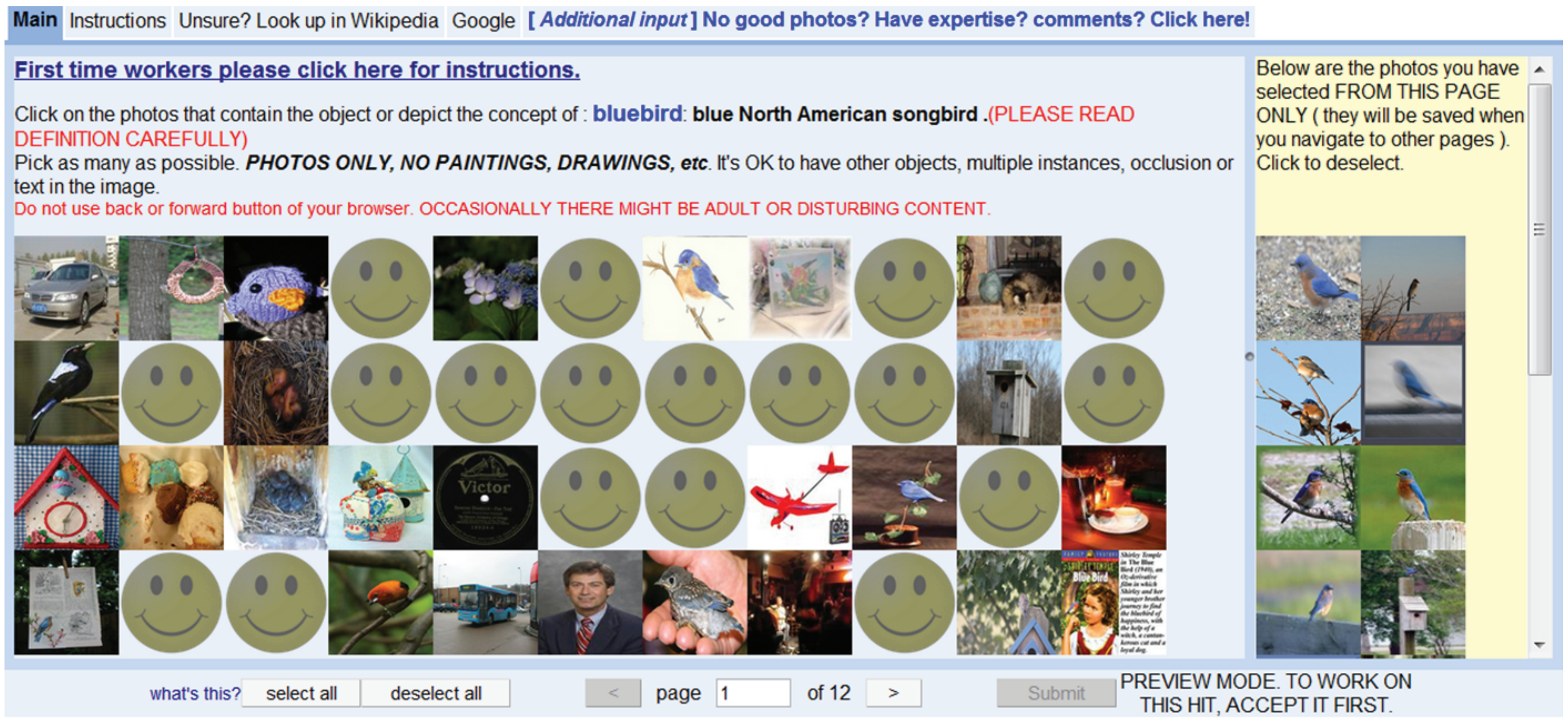
First time workers please click here for instructions.

Click on the photos that contain the object or depict the concept of : **bluebird**: **blue North American songbird** .(PLEASE READ DEFINITION CAREFULLY)
Pick as many as possible. ***PHOTOS ONLY, NO PAINTINGS, DRAWINGS, etc***. It's OK to have other objects, multiple instances, occlusion or text in the image.
Do not use back or forward button of your browser. OCCASIONALLY THERE MIGHT BE ADULT OR DISTURBING CONTENT.

Below are the photos you have selected FROM THIS PAGE ONLY (they will be saved when you navigate to other pages). Click to deselect.

PREVIEW MODE. TO WORK ON THIS HIT, ACCEPT IT FIRST.

Figure 2.9: The user interface for verifying the candidate images.



Main | Instructions | Unsure? Look up in Wikipedia | Google | [*Additional input*] No good photos? Have expertise? comments? Click here!

First time workers please click here for instructions.

Click on the photos that contain the object or depict the concept of : **bluebird**: **blue North American songbird** .(PLEASE READ DEFINITION CAREFULLY)
Pick as many as possible. ***PHOTOS ONLY, NO PAINTINGS, DRAWINGS, etc***. It's OK to have other objects, multiple instances, occlusion or text in the image.
Do not use back or forward button of your browser. OCCASIONALLY THERE MIGHT BE ADULT OR DISTURBING CONTENT.

Below are the photos you have selected FROM THIS PAGE ONLY (they will be saved when you navigate to other pages). Click to deselect.

PREVIEW MODE. TO WORK ON THIS HIT, ACCEPT IT FIRST.

Figure 2.10: A finished labeling task.

confidence score table, indicating the probability of an image being a good image given the user votes (Figure 2.8 (right) shows examples for "Burmese cat" and "cat"). For each of remaining candidate images in this synset, we proceed with the AMT user labeling until a pre-determined confidence score threshold is reached. Naturally, when a synset is more familiar to most of the average users, a smaller number of voters is needed to judge the images.

To obtain the table of confidence scores, we model the voting process as follows. Assume that the synset s is being considered. Each candidate image has an intrinsic parameter $p \in [0, 1]$ that indicates the difficulty of the image. The process of getting a new worker to vote $v \in \{1, 0\}$ on an image is a Bernoulli trial parametrized by the difficulty p of the image. That is, $\Pr(v = 1) = p$. The image is considered positive for synset s if the majority would vote yes, i.e., $p > 0.5$. Further we assume that p takes only discrete values $\{p_k\}, k = 1, \dots, l$ and is generated a multinomial distribution parametrized by $\mathbf{q}^{(s)} = \{q_k^{(s)}\}, k = 1, \dots, l$ such that $\Pr(p = p_k|s) = q_k^{(s)}$. The parameter $\mathbf{q}^{(s)}$ models the difficulty of the synset s. Given the current votes $\mathbf{v}$ and the synset s, the confidence of the image is then i.e., the probability that $p > 0.5$, that is,

$$\begin{aligned}\Pr(p > 0.5|\mathbf{v}, s) = & \frac{\Pr(\mathbf{v}|p > 0.5)\Pr(p > 0.5|s)}{\Pr(\mathbf{v}|s)} \\ = & \frac{\sum_{k:p_k>0.5}\Pr(\mathbf{v}|p = p_k)q_k^{(s)}}{\sum_k \Pr(\mathbf{v}|p = p_k)q_k^{(s)}},\end{aligned}$$

where

$$\Pr(\mathbf{v}|p = p_k) = \prod_i p_k^{v_i}(1 - p_k)^{(1-v_i)}.$$

We use the votes in the initial subset to estimate $\mathbf{q}^{(s)}$. Since there is a large number of votes per image, we can estimate the the difficulty p of each image by the percentage of positive votes. Then $\mathbf{q}^{(s)}$ is estimated by binning the estimates of p.

It is worth noting that the confidence table gives a natural measure of the "semantic difficulty" of the synset. For some synsets, users fail to reach a majority vote for any image, indicating that the synset cannot be easily illustrated by images. [3]

A final measure of quality control is to identify completely unacceptable submissions which can be rejected without a payment. We achieve this by embedding gold standard (known good/bad images obtained from the initial subset with high confidence) and comparing a worker's submission on these images with the ground truth.

With all the quality control mechanism in place, our algorithm successfully filters the candidate images, resulting in a high percentage of accurately labeled images per synset, as shown in Figure 2.4.

Since the start of the ImageNet project in 2007, we have processed 160 millions through Amazon Mechanical Turk. Our experience is that AMT is extremely scalable in terms of speed. For example, at our peak, we have up to 5,000 submissions each day; each submission verifies 250 images. This gives a speed of over 1 million images per day. Second, quality control is very important. We reject around 10% of the submissions due to spammers.

2.5 Discussion

In this section we discuss how we will further expand ImageNet and how ImageNet can be used to advance vision related research.

2.5.1 Expanding ImageNet

The current ImageNet constitutes around 25% of the WordNet synsets but has covered many common objects. Not all of the WordNet synsets can be visually illustrated—our preliminary estimate suggests that, for the remaining synsets of WordNet, a large portion is not visual. We will nevertheless keep expanding ImageNet to achieve full coverage of all visual synsets. Moreover, many parts of the WordNet hierarchy can be expanded to even

[3]An alternative explanation is that we did not obtain enough suitable candidate images. Given the extensiveness of our crawling scheme, this is a rare scenario.

more fine-grained sub-categories. For examples, the car categories in WordNet do not yet include different brands of cars.

Another direction of expanding ImageNet is to include more information such as localization, segmentation, cross-synset referencing of images, as well as expert annotation for difficult synsets.

2.5.2 Exploiting ImageNet

ImageNet can serve as a useful resource for a broad of range of vision related research.

First, ImageNet can be used as a training resource. Most of today's object recognition algorithms have focused on a small number of common objects, such as pedestrians, cars and faces. This is mainly due to the high availability of images for these categories. Figure 2.7 has shown that even the largest datasets today have a strong bias in their coverage of different types of objects. ImageNet, on the other hand, contains a large number of images for nearly all object classes including rare ones. One interesting research direction could be to transfer knowledge of common objects to learn rare object models.

Second, ImageNet is well suited as a benchmark dataset. The current benchmark datasets in computer vision such as Caltech101/256 and PASCAL have played a critical role in advancing object recognition and scene classification research. ImageNet is at least two orders of magnitude larger than these datasets and it can serve as a new and challenging benchmark dataset for future research.

Thrid, ImageNet can allow us to introduce new semantic relations for visual modeling. Because ImageNet is uniquely linked to all concrete nouns of WordNet, and those synsets are all richly interconnected, one could also exploit different semantic relations, e.g., to learn part models. To move towards total scene understanding, it is also helpful to consider different depths of the semantic hierarchy.

Finally, ImageNet can facilitate human vision research. ImageNet's rich structure and dense coverage of the image world may help advance the understanding of the human visual system. For example, the question of whether a concept can be illustrated by images is much more complex than one would expect at first. Also, how to empirically determine whether any given category is basic, super-ordinate, or subordinate remains an unexplored area.

2.6 Summary

We have presented ImageNet, a large scale ontology of images built upon the backbone of the WordNet structure. ImageNet aims to populate the majority of the 80,000 synsets of WordNet with an average of 500–1,000 human-verified and full resolution images. The Fall 2011 release of ImageNet has 22 thousands synsets and 14 million images. We have shown that ImageNet is much larger in scale and diversity and much more accurate than the previous image datasets. Constructing such a large scale database is a challenging task. We have also presented our data collection scheme with Amazon Mechanical Turk and discussed new research opportunities opened up by the scale, accuracy, diversity, and hierarchical structure of ImageNet.

Chapter 3

Benchmarking with ImageNet

3.1 Introduction

The construction of ImageNet paves the road toward large scale recognition. Its broad coverage of categories allows us to ask an important question that was impossible to ask before—how does the current state of the art algorithms work at the human scale? [1] Answers to this question can help us understand the limitations of current methods and shed light on future research directions.

Recent progress on image categorization has been impressive and has introduced a range of features, models, classifiers, and frameworks [18, 14, 15, 71, 36, 72, 73, 74, 75]. In this chapter we explore scaling up the number of categories considered in recognition experiments from hundreds to over 10 thousand, in order to map out the opportunities and challenges of large scale recognition and eventually move toward reducing the gap between machine performance and human abilities (Figure 3.1). Note that this is not simply a matter of training more and more classifiers (although that is a challenging task on its own). With such large numbers of categories there is a concomitant shift in the difficulty of discriminating between them as the categories sample the semantic space more densely.

[1]An early version of this chapter has been presented in [52].

The previously unexplored scale of the experiments in this work allow this effect to be measured.

Recognition encompasses a wide range of specific tasks, including classification, detection, viewpoint understanding, segmentation, verification and more. In this study we focus on category recognition, in particular the task of assigning a single category label to an image that contains one or more instances of a category of object following the work of [76, 77, 4, 5].

We conduct the first empirical study of image categorization at near human scale. Some results are intuitive – discriminating between thousands of categories is in fact more difficult that discriminating between hundreds – but other results reveal structure in the difficulty of recognition that was previously unknown. Our key contributions are as follows.

First, we provide the first in-depth study of image classification at such a large scale. Such experiments are technically challenging, and we present a series of techniques to overcome this difficulty (Section 3.4.1).

Second, we show that conventional wisdom obtained from current datasets does not necessarily hold in some cases at a larger scale. For example, the ordering by performance of techniques on hundreds of categories is not preserved on thousands of categories. Thus, we cannot solely rely on experiments on the Caltech [4, 5] and PASCAL [77] datasets to predict performance on large classification problems (Section 3.4.2).

Third, we propose a measure of similarity between categories based on WordNet [37] – a hierarchy of concepts developed for studying language. Experiments show a surprisingly strong correlation between this purely linguistic metric and the performance of visual classification algorithms. We also show that the categories used in previous object recognition experiments are relatively sparse – distinguishing them from each other is significantly less difficult than distinguishing many other denser subsets of the 10K categories (Section 3.4.3).

Finally, observing that object categories are naturally hierarchical, we propose and eval-

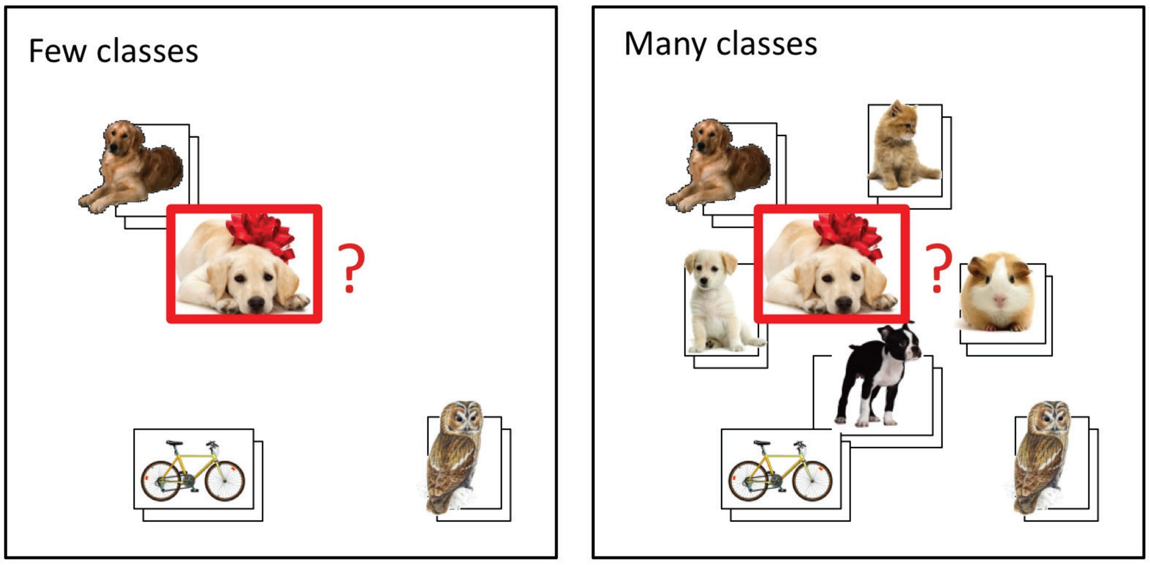


Figure 3.1: Given a query image, the task of "image classification" is to assign it to one of the classes (represented by a stack of images) that the algorithm has learned. Left: Most traditional vision algorithms have been tested on a small number of somewhat distinct categories. Right: Real world image classification problems may involve a much larger number of categories – so large that the categories can no longer be easily separated.

uate a technique to perform hierarchy aware classification, and show that more informative classification results can be obtained (Section 3.4.4).

3.2 Related Work

Much recent work on image classification has converged on bag of visual word models (BoW) [17] based on quantized local descriptors [13, 14, 15] and support vector machines [14, 18] as basic techniques. These are enhanced by multi-scale spatial pyramids (SPM) [15] on BoW or histogram of oriented gradient (HOG) [19, 15] features. In the current state of the art, multiple descriptors and kernels are combined using either ad hoc or multiple kernel learning approaches [78, 71, 79, 80]. Work in machine learning supports using winner-takes-all between 1-vs-all classifiers for the final multi-class classification decision [81]. We choose SPM using BoW because it is a key component of many of the best recognition results [78, 71, 79, 80] and is relatively efficient. Recent work allows fast approximation of the histogram intersection kernel SVM, used for SPM, by a linear SVM on specially encoded SPM features [82]. See Section 3.4.5 for the modifications necessary to allow even that very efficient solution to scale to very large problems.

Prior to this work there have been no recognition results on more than a few hundreds of categories. Previous work on Tiny Images [36] shows only proof of concept classification on fewer than 50 categories. Fergus et al. explore semi-supervised learning on 126 hand labeled Tiny Images categories [83] and Wang et al. show classification on a maximum of 315 categories ($< 5\%$) [84].

Recent work considering hierarchies for image recognition or categorization [22, 23, 24, 25] has shown impressive improvements in accuracy and efficiency, but has not studied classification minimizing hierarchical cost. Related to classification is the problem of detection, often treated as repeated 1-vs-all classification in sliding windows. In many cases such localization of objects might be useful for improving classification, but even the most efficient of the state of the art techniques [72, 79, 33] take orders of magnitude more time per image than the ones we consider in this study, and thus cannot be utilized given the scale of our experiments.

3.3 Approach

3.3.1 Datasets

The goals of this chapter are to study categorization performance on a significantly larger number of categories than the current state of the art, and furthermore to delve deeper toward understanding the factors that affect performance. The size and breadth of ImageNet allow us to perform multiple longitudinal probes of the classification problem. Specifically we consider the following subsets:

- **ImageNet10K.** 10,184 categories from the Fall 2009 release of ImageNet, including both internal and leaf nodes with more than 200 images each (a total of 9 million images).
- **ImageNet7K.** 7,404 leaf categories from ImageNet10K. Internal nodes may overlap with their descendants, so we also consider this leaf only subset.
- **ImageNet1K.** 1,000 leaf categories randomly sampled from ImageNet7K.
- **Rand200{a,b,c}.** Three datasets, each containing 200 randomly selected leaf categories. The categories in Rand200a are sampled from ImageNet1K while Rand200b and Rand200c are sampled directly from ImageNet7K.
- **Ungulate183, Fungus134, Vehicle262.** Three datasets containing all the leaf nodes that are descendants of particular parent nodes in ImageNet10K (named by the parent node and number of leaves).
- **CalNet200**. This dataset serves as a surrogate for the Caltech256 dataset – containing the 200 image categories from Caltech256 that exist in ImageNet.

Note that all datasets have non-overlapping categories except ImageNet10K. Following the convention of the PASCAL VOC Challenge, each category is randomly split 50%-50% into a set of training and test images, with a total of 4.5 million images for training and 4.5 million images for testing. All results are averaged over two runs by swapping training and test, except for ImageNet7K and ImageNet10K due to extremely heavy computational cost. In all cases we provide statistical estimates of the expected variation. The number of training images per category ranges from 200 to 1,500, with an average of 450.

3.3.2 Methodology

The main thrust of this study is image classification: given an image and K classes, the task is to select one class label. We employ two evaluation measures:

Mean accuracy. The accuracy of each class is the percentage of correct predictions, i.e., predictions identical to the ground truth class labels. The mean accuracy is the average accuracy across all classes.

Mean misclassification cost. To exploit the hierarchical organization of object classes, we also consider the scenario where it is desirable to have non-uniform misclassification cost. For example, misclassifying "dog" as "cat" might not be penalized as much as misclassifying "dog" as "microwave." Specifically, for each image $x_i^{(k)} \in X, i = 1, \dots, m$ from class k, we consider predictions $f(x_i^{(k)}) : X \rightarrow \{1, \dots, K\}$, where K is the number of classes (e.g., $K = 1000$ for ImageNet1K) and evaluate the cost for class k as $L_k = \frac{1}{m}\sum_{i=1}^{m} C_{f(x_i^{(k)}),k}$, where C is a $K \times K$ cost matrix and $C_{i,j}$ is the cost of classifying the true class j as class i. The mean cost is the average cost across all classes. Evaluation using a cost based on the ImageNet hierarchy is discussed in Section 3.4.4.

We use the following four algorithms in our evaluation experiments as samples of some major techniques used in object recognition:

- `GIST+NN` Represent each image by a single GIST [85] descriptor (a commonly accepted baseline descriptor for scene classification) and classify using *k-nearest-neighbors* (kNN) on L2 distance.

- `BOW+NN` Represent each image by a histogram of SIFT [13] codewords and classify using kNN on L1 distance, as a baseline for BoW NN-based methods.

- `BOW+SVM` Represent each image by a histogram of SIFT codewords, and train and classify using linear SVMs. Each SVM is trained to distinguish one class from the rest. Images are classified by the class with largest score (a 1-vs-all framework). This serves as a baseline for classifier-based algorithms.

- `SPM+SVM` Represent each image by a spatial pyramid of histograms of SIFT codewords [15]. Again a 1-vs-all framework is used, but with approximate histogram intersection kernel SVMs [82, 14, 15]. This represents a significant component of many state of the art classifiers [78, 71, 79, 80].

3.4 Results

3.4.1 Computation Matters

Working at the scale of 10,000 categories and 9 million images moves computational considerations to the forefront. Many common approaches become computationally infeasible at such large scale.

As a reference, for this data it takes 1 hour on a 2.66GHz Intel Xeon CPU to train *one* binary linear SVM on bag of visual words histograms (including a minimum amount of parameter search using cross validation), using the extremely efficient LIBLINEAR [86]. In order to perform multi-class classification, one common approach is 1-vs-all, which entails training 10,000 such classifiers – requiring more than 1 CPU year for training and 16 hours for testing. Another approach is 1-vs-1, requiring 50 million pairwise classifiers. Training takes a similar amount of time, but testing takes about 8 years due to the huge number of classifiers. A third alternative is the "single machine" approach, e.g., Crammer & Singer [87], which is comparable in training time but is not readily parallelizable. We choose 1-vs-all as it is the only affordable option.

Training `SPM+SVM` is even more challenging. Directly running intersection kernel SVM is impractical because it is at least $100\times$ slower (100+ years) than linear SVM [82]. We use the approximate encoding proposed by Maji & Berg [82] that allows fast training with LIBLINEAR. This reduces the total training time to 6 years. However, even this very efficient approach must be modified because memory becomes a bottleneck [2] – a direct application of the efficient encoding of [82] requires 75GB memory, far exceeding our

[2]While it is possible to use online methods, e.g., stochastic subgradient descent, they can be slower to converge [86].

memory limit (16GB). We reduce it to 12G through a combination of techniques detailed in Section 3.4.5.

For NN based methods, we use brute force linear scan. It takes 1 year to run through all testing examples for GIST or BOW features. It is possible to use approximation techniques such as locality sensitive hashing [55], but due to the high feature dimensionality (e.g., 960 for GIST), we have found relatively small speedup. Thus we choose linear scan to avoid unnecessary approximation.

In practice, all algorithms are parallelized on a computer cluster of 66 multicore machines, but it still takes weeks for a single run of all our experiments. Our experience demonstrates that computational issues need to be confronted at the outset of algorithm design when we move toward large scale image classification, otherwise even a baseline evaluation would be infeasible. Our experiments suggest that to tackle massive amount of data, distributed computing and efficient learning will need to be integrated into any vision algorithm or system geared toward real-world large scale image classification.

3.4.2 Size Matters

We first investigate the broad effects on performance and computation of scaling to ten-thousand categories. As the number of categories in a dataset increases, the accuracy of classification algorithms decreases, from a maximum of 34% for Rand200{a,b,c} to 6.4% for ImageNet10K (Figure 3.2). While the performance drop comes at no surprise, the speed of decrease is slower than might be expected – roughly a $2\times$ decrease in accuracy with $10\times$ increase in the number of classes, significantly better than the $10\times$ decrease of a random baseline.

There is a surprise from *k-nearest-neighbor* (*kNN*) classifiers, either using GIST features or BoW features. For Rand200{a,b,c}, these techniques are significantly worse than linear classifiers using BoW features, around 10% lower in accuracy. This is consistent with the experience of the field – methods that do use *kNN* must be augmented in order to

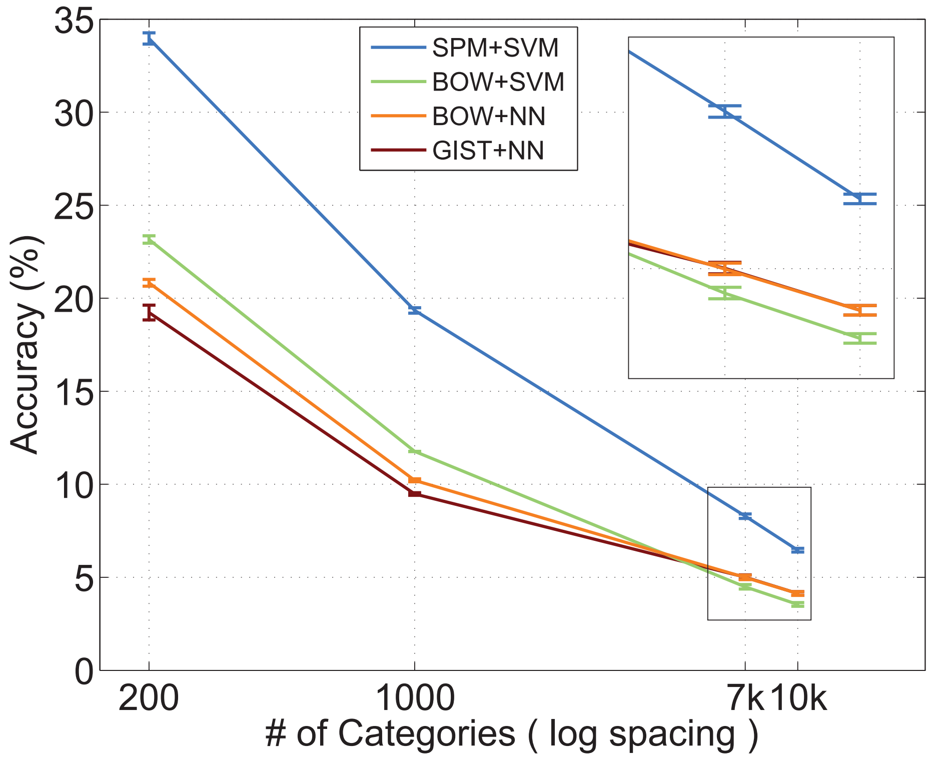


Figure 3.2: Mean classification accuracy of various methods on Rand200$\{a, b, c\}$, ImageNet1K, ImageNet7K and ImageNet10K.

provide competitive performance [18, 88]. But the picture is different for ImageNet7K or ImageNet10K categories, where simple *kNN* actually outperforms linear SVMs on BoW features (`BOW+SVM`), with 11-16% *higher* accuracy. The small absolute gain in mean accuracy, around 0.5%, is made significant by the very small expected standard deviation of the means 0.1%. [3] *A technique that significantly outperforms others on small datasets may actually underperform them on large numbers of categories*.

This apparent breakdown for 1-vs-all with linear classifiers comes despite a consistent line of work promoting this general strategy for multi-class classification [81]. It seems to reveal issues with calibration between classifiers, as the majority of categories have comparable discriminative power on ImageNet7K and Rand200a (Fig 3.3 *left*), but multi-way classification is quite poor for ImageNet7K (Fig 3.3 *right*). One explanation is that

[3]Stdev for ImageNet7K and ImageNet10K are estimated using the individual category variances, but are very small *cf* standard error and the central limit theorem.

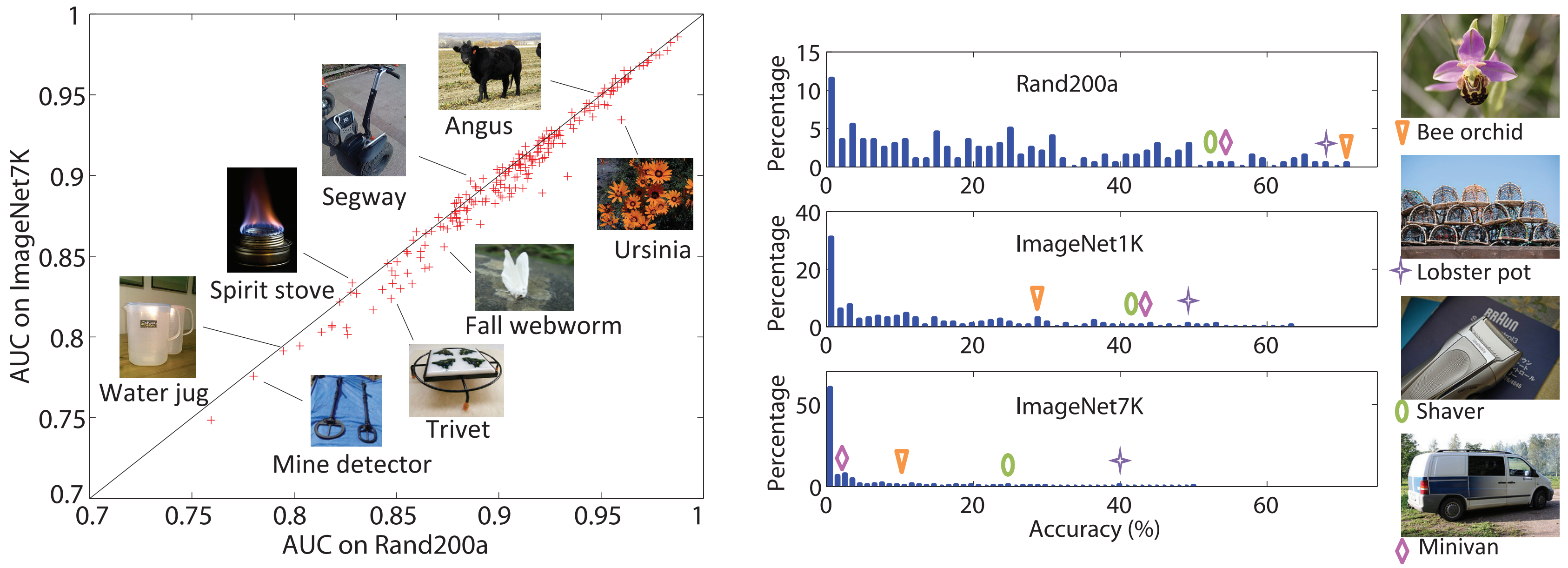


Figure 3.3: Left: Scatter plot comparing the area under ROC curve (AUC) of `BOW+SVM` for the 200 categories in Rand200a when trained and evaluated against themselves(x-axis) and when trained and evaluated against ImageNet7K(y-axis). Right: Histograms of accuracies for the same 200 categories in Rand200a, ImageNet1K, and ImageNet7K, example categories indicated with colored markers.

for the one-against-all approach, a correct prediction would require that the true classifier be more confident than any other classifiers, which becomes more difficult with a larger number of classes as the chance of false alarms from others greatly increases. Then the behavior starts to resemble kNN methods, which are only confident about close neighbors.

Looking in more detail at the confusion between the categories in ImageNet7K reveals additional structure (Figure 3.4). Most notable is the generally block diagonal structure, *indicating a correlation between the structure of the semantic hierarchy (by WordNet) and visual confusion between the categories*. The two most apparent blocks roughly align with "artifacts" and "animals," two very high level nodes in WordNet, suggesting the least amount of confusion between these two classes with more confusion within. This is consistent with both computational studies on smaller numbers of classes [25] and some human abilities [89]. Sections of the confusion matrix are further expanded in Figure 3.4. These also show roughly block diagonal structures at increasingly finer levels not available in other datasets. The pattern is roughly block diagonal, but by no means exact. There is a great deal of noise and a fainter "plaid," oscillating pattern of stripes, indicating that the ordering of categories in WordNet is not completely in agreement with the visual confusion between them.

The block patterns indicate that it is possible to speed up the classification by using a sublinear number of classifiers in a hierarchy, as Griffin & Perona have demonstrated on Caltech256 [25]. They built a hierarchy of classifiers directly from the confusion matrix. Here we confirm their findings by observing a much stronger pattern on a large number of classes. Moreover we note that such a grouping may actually be directly obtained from WordNet, in which case, the output of an internal classifier in the hierarchy would be semantically meaningful.

Also of note is that in scaling to many classes, only a small subset of the distractor classes are truly distracting, possibly explaining the smaller than expected performance drop. For example, to classify "German shepherd," most of the distractor classes are "easy"

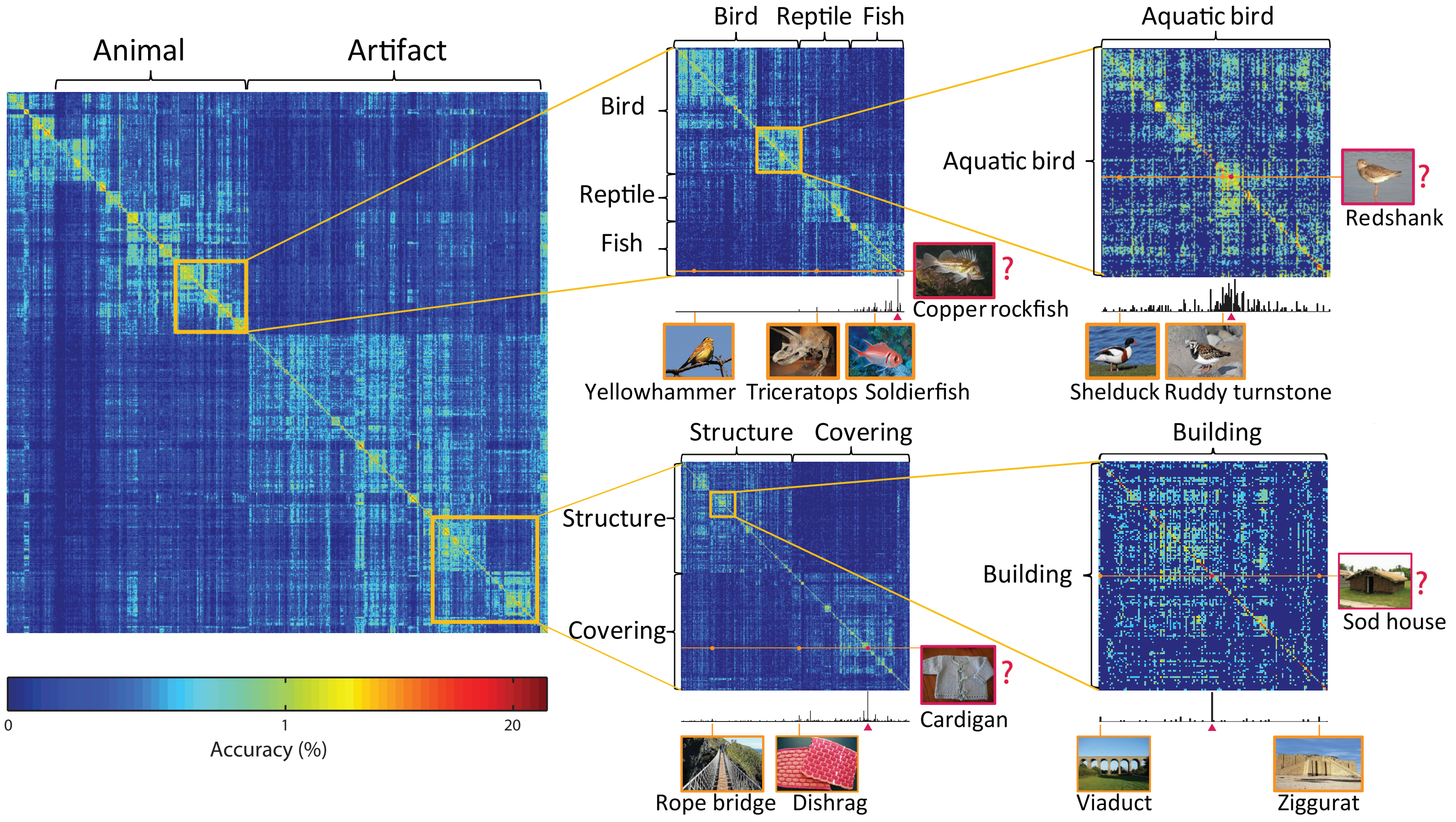


Figure 3.4: Confusion matrix and sub-matrices of classifying the 7,404 leaf categories in ImageNet7K, ordered by a depth first traversal of the WordNet hierarchy, using SPM+SVM. Left: Downsampled 7,404 $\times$ 7,404 confusion matrix, each pixel representing max confusion over 4 $\times$ 4 entries. Middle: Zoom-in to two sub-matrices, each pixel representing 2 $\times$ 2 entries. One row of the matrix is plotted below each matrix (corresponding to red outlined images). The correct class is indicated by a red triangle. Examples of other classes are also shown. Right: Further zoom-in, each pixel representing the confusion between two individual categories.

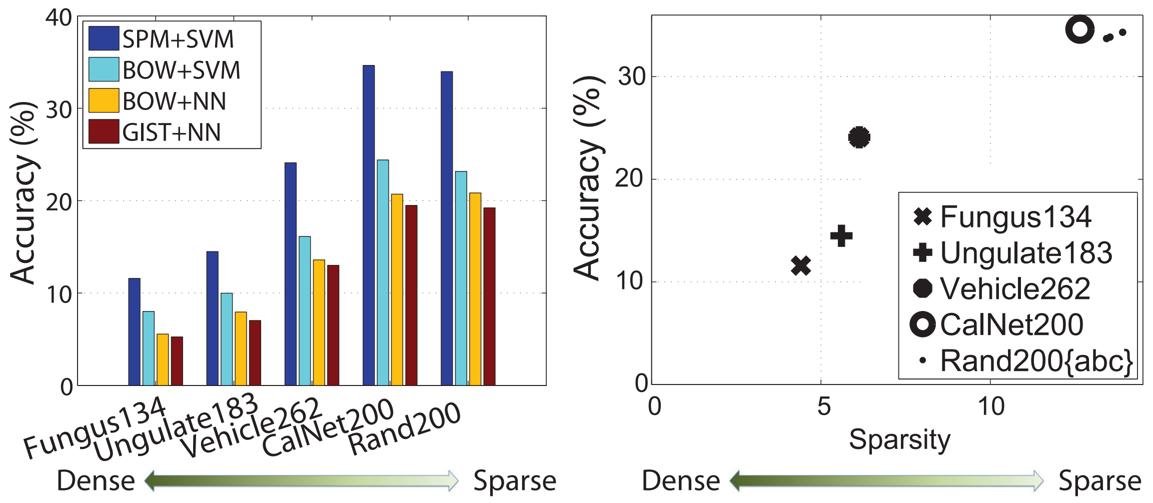


Figure 3.5: Left: Accuracy on datasets of varying density. Note that CalNet200 (the Caltech 256 categories in ImageNet) has low density and difficulty on par with a set of 200 randomly sampled categories. Right: Accuracy (using `SPM+SVM`) versus dataset density measured by mean distance in WordNet (see Section 3.4.3).

ones like “dishrag,” while only a few semantically related classes like “husky” add to the difficulty. It suggests that one key to improving large scale classification is to focus on those classes, whose difficulty correlates with semantic relatedness. We quantify this correlation in Section 3.4.3.

3.4.3 Density Matters

Our discussion so far has focused on the challenges arising from the sheer number of categories. Figure 3.4 reveals that the difficulty of recognition varies significantly over different parts of the semantic space. Some classifiers must tackle more semantically related, and possibly visually similar, categories. Accurate classification of such categories leads to useful applications, e.g., classifying groceries for assisting the visually impaired, classifying home appliances for housekeeping robots, or classifying insect species for environmental monitoring [90]. We refer to sets of such categories as *dense* and study the effect of density on classification.

We begin by comparing mean classification accuracy for classifiers trained and tested on each of the small datasets – Fungus134, Ungulate183, Vehicle262, CalNet200, Rand200 –

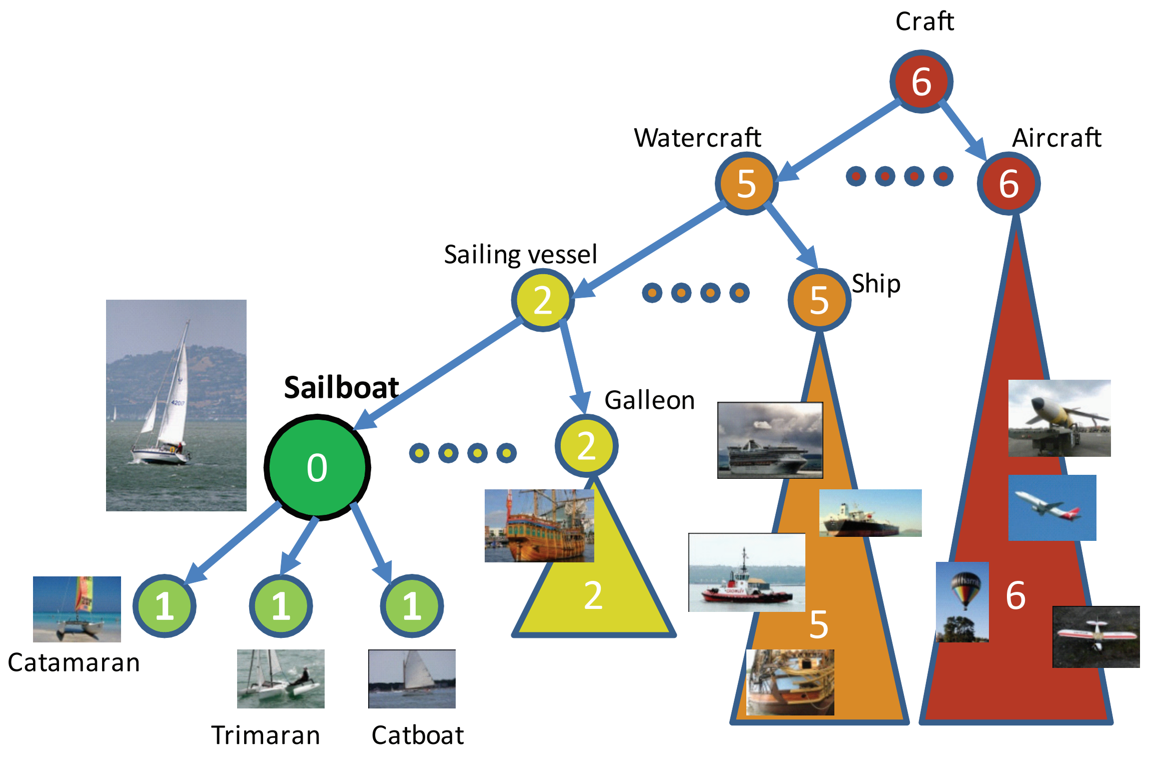


Figure 3.6: Illustration of the inter-class distance (indicated by the numbers) between "sailboat" and other classes, as defined in Section 3.4.3. Any descendant of ship is further from sailboat than gallon but closer than those in aircraft. Note that one step up the hierarchy may increase the distance by more than one as the tree height is the length of the longest path to a leaf node.

across descriptors and classifiers in Figure 3.5. Note that while `SPM+SVM` produces consistently higher accuracies than the other approaches, the ordering of datasets by performance is exactly the same for each approach. [4] This indicates that *there is a significant difference in difficulty between different* datasets*, independent of feature and classifier choice*.

Next we try to predict the difficulty of a particular dataset by measuring the density of the categories, based on the hierarchical graph structure of WordNet. We define the distance, $h(i, j)$, between categories i and j, as the height of their lowest common ancestor. The height of a node is the length of the longest path down to a leaf node (leaf nodes have height 0). We measure the density of a dataset as the mean $h(i, j)$ between all pairs of

[4]Ordering of datasets is consistent, but ordering of methods may change between datasets as noted in Section 3.4.2 where `BOW+SVM` and the kNN approaches switch order.

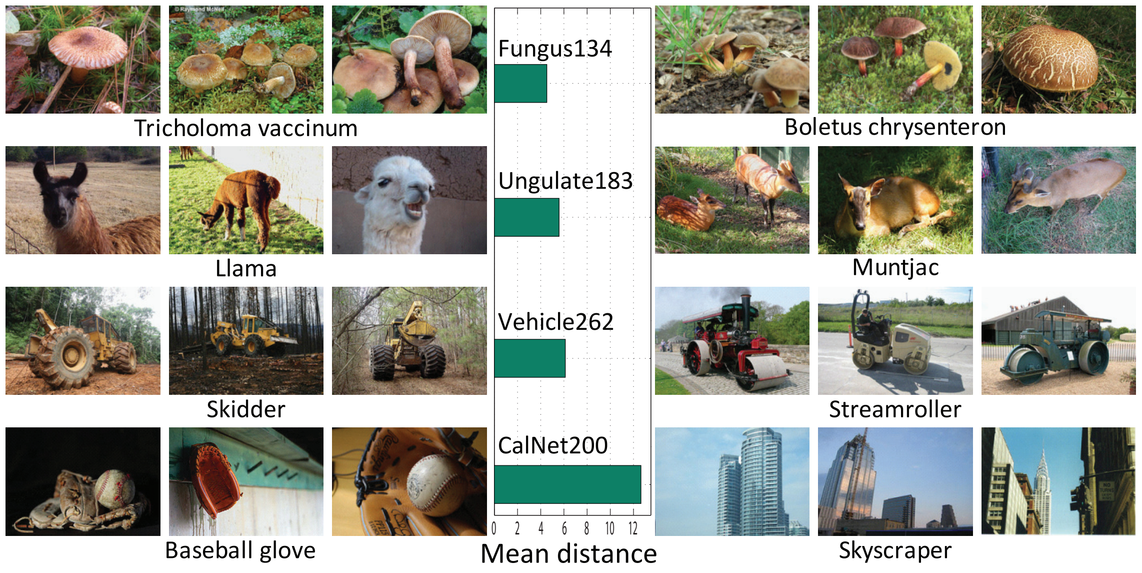


Figure 3.7: Each row shows a pair of example categories from the dataset indicated in the center column. The pairs are chosen to have distance near the mean distance in WordNet (Section 3.4.3) between categories in the dataset, indicated by the bars in the center column.

categories – smaller implies denser. See Figure 3.6 for an illustration and Figure 3.7 for examples of pairs of categories from each dataset that have distance closest to the mean for that dataset. There is a very *clear correlation between the density in WordNet and accuracy of visual classification;* denser *datasets predict lower accuracy* (Figure 3.5). This is despite the fact that WordNet was not created as a visual hierarchy!

Classification accuracy on 200 randomly chosen categories (Rand200{a,b,c}) is more than 3 times higher than on the 134 categories from Fungus134. The large gap suggests that the methods studied here are not well equipped for classifying dense sets of categories. In fact, there have been relatively few efforts on "dense classification" with some notable exceptions, e.g., [91, 92, 90]. The results seem to call for perhaps more specialized features and models, since it is one key to improving large scale classification performance as discussed in Section 3.4.2

Also of note is that the Caltech256 categories that occur in ImageNet (CalNet200) have very low density and relatively high accuracy – in almost exactly the same range as random sets of categories. *The Caltech categories are very sparse and do not exhibit* *the difficulty of dense sets of categories, making Caltech-like datasets incomplete as an evaluation resource towards some of the real-world image classification problems.*

Finally we note that our WordNet based measure is not without limitations, e.g., “food tuna” and “fish tuna” are semantically related but belong to “food” and “fish” subtrees respectively, so are far away from each other. Nonetheless as a starting point for quantifying semantic density, the results are encouraging.

3.4.4 Hierarchy Matters

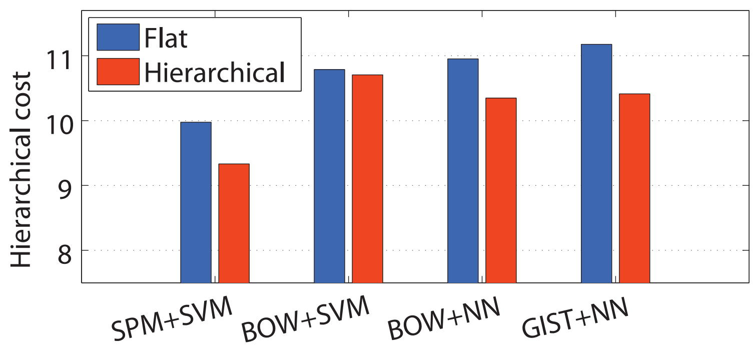
For recognition at the scale of human ability, categories will necessarily overlap and display a hierarchical structure [76]. For example, a human may label “redshank” as “shorebird,” “bird,” or “animal,” all of which are correct but with a decreasing amount of information. Humans make mistakes too, but to different degrees at different levels – a “redshank” might be mistaken as a “red-backed sandpiper,” but almost never as anything under “home appliance.”

The implications for real world object classification algorithms are two fold. First a learning algorithm needs to exploit real world data that inevitably has labels at different semantic levels. Second, it is desirable to output labels as informative as possible while minimizing mistakes at higher semantic levels.

Consider an automatic photo annotator. If it cannot classify “redshank” reliably, an answer of “bird” still carries much more information than “microwave.” However, our classifiers so far, trained to minimize the 0-1 loss, [5] have no incentive to do so – predicting “microwave” costs the same as predicting “bird.”

Here we explore ways to make classifiers more informative. We define a hierarchical cost $C_{i,j}$ for classifying an image of class j as class i as $C_{i,j} = 0$ when $i = j$ or when i is a descendant of j, and $C_{i,j} = h(i, j)$, the height of their lowest common ancestor in

[5]The standard loss function for classification, where a correct classification costs zero and any incorrect classification costs 1.



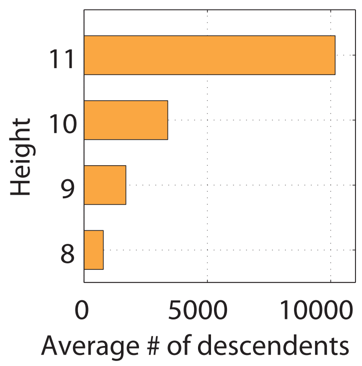


Figure 3.8: Left: Hierarchical cost of flat classification and hierarchical classification on ImageNet10K across different methods. Right: Mean number of descendants for nodes at each height, indicating the effective log-scale for hierarchical cost.

WordNet, otherwise. This cost definition directly measures the semantic level at which a misclassification occurs – a more informative classifier, one able to discriminate finer details, would have lower cost. It also takes care of the overlapping categories – there is penalty for classifying an image in an internal node as its (more general) ancestor but no cost for classifying it as any of its (more specific) descendants. As an example, in Figure 3.6, for an image labeled as "sailboat," classifying it as "catamaran" or any other descendant incurs no cost [6] while classifying as any descendant of "aircraft" incurs cost 6.

We can make various classification approaches cost sensitive by obtaining probability estimates (Section 3.4.5). For a query image x, given posterior probability estimates $\hat{p}_j(x)$ for class j, $j \in \{1, \ldots K\}$, according to Bayesian decision theory, the optimal prediction is obtained by predicting the label that minimizes the expected cost $f(x) = \arg\min_{i=1,\ldots,K} \sum_{j=1}^{K} C_{i,j}\hat{p}_j(x)$.

Comparing the mean hierarchical cost for the original (flat) classifier with the mean cost for the cost sensitive (hierarchical) classifier, we find a consistent reduction in cost on ImageNet10K (Figure 3.8). It shows that the hierarchical classifier can discriminate at more informative semantic levels. While these reductions may seem small, the cost is effectively on a log scale. It is measured by the height in the hierarchy of the lowest common ancestor,

[6] The image can in fact be a "trimaran," in which case it is not entirely correct to predict "catamaran." This is a limitation of intermediate level ground truth labels.

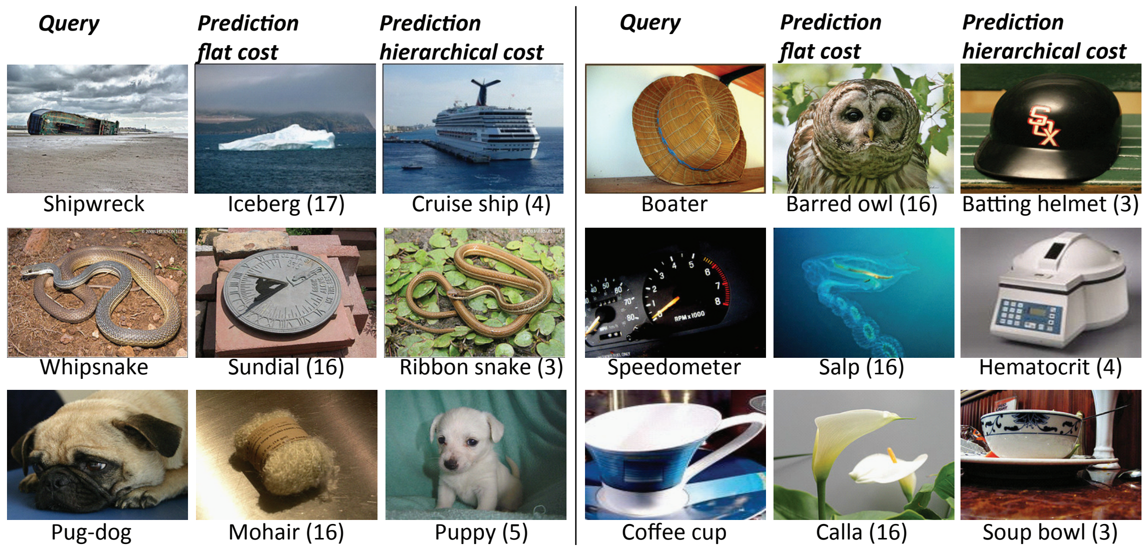


Figure 3.9: Example errors using a flat vs. hierarchical classifier with `SPM+SVM` on ImageNet10K, shown in horizontal groups of three: a query, prediction by a flat classifier (minimizing 0-1 loss), and by a hierarchical classifier (minimizing hierarchical cost). Numbers indicate the hierarchical cost of that misclassification.

and moving up a level can more than double the number of descendants (Figure 3.8 *right*).

The reduction of mean cost on its own would not be interesting without a clear benefit to the results of classification. The examples in Figure 3.9 show query images and their assigned class for flat classification and for classification using hierarchical cost. While a whipsnake is misclassified as ribbon snake, it is still correct at the "snake" level, thus giving a more useful answer than "sundial." It demonstrates that *classification based on hierarchical cost can be significantly more informative.*

3.4.5 Experimental Details

We obtain BoW histograms (L1-normalized) using dense SIFT [93] on 20x20 overlapping patches with a spacing of 10 pixels at 3 scales on images resized to a max side length of 300, and a 1K codebook from KMeans on 10 million SIFT vectors. We use the same codewords to obtain spatial pyramid histograms (3 levels), ϕ_2 encoded [82] to approximate the intersection kernel with linear SVMs. Due to high dimensionality (21K), we only encode nonzeros (but add a bias term). This preserves the approximation for our, non-negative, data, but with slightly different regularization. We found no empirical performance difference testing up to 1K categories. To save memory, we use only two bytes for each entry of encoded vectors (sparse) by delta-coding its index (1 byte) and quantizing its value to 256 levels (1 byte). We further reduce memory by only storing every other entry, exploiting redundancy in consecutive entries. We use LIBLINEAR [86] to train linear SVMs, parameter C determined by searching over 3 values (0.01, 0.1, 1 for ImageNet10K) with 2-fold cross validation. We use smaller weight for negative examples ($100\times$ smaller for ImageNet10K) than positives. We obtain posterior probability estimates by fitting a sigmoid function to the outputs of SVMs [94], or by taking the percent of neighbors from a class for NN.

3.5 Summary

We have presented the first large scale recognition experiments on 10,184 categories and 9 million images. We show that challenges arise from the size and density of the semantic space. The performance of the current state of the art algorithm is still low, 6.4% on 10K categories. Surprisingly the ordering of NN and Linear classification approaches swap from previous datasets to our very large scale experiments – we cannot always rely on experiments on small datasets to predict performance at large scale. We produce a measure of category distance based on the WordNet hierarchy and show that it is well correlated with the difficulty of various datasets. We present a hierarchy-aware cost function for classification and show that it produces more informative classification results. These experiments point to future research directions in large scale image classification, as well as critical dataset and highlight potential benchmarking issues for evaluating different algorithms.

Chapter 4

Efficient Large Scale Classification

4.1 Introduction

This chapter addresses the computational challenges of learning to recognize tens of thousands of visual object categories. [1] The large number of classes renders the standard one-versus-all multiclass approach too costly, as the complexity grows linearly with the number of classes, for both training and testing. This issue is especially relevant for practical applications that require low latency or high throughput, such as those in robotics or in image retrieval.

Classification with many classes has received increasing attention recently, and most approaches appear to have converged to tree based models [42, 43, 25, 26]. In particular, Bengio et al. [26] proposes a *label tree* model, which has been shown to achieve state of the art performance in testing. In a label tree, each node is associated with a subset of class labels and a linear classifier that determines which branch to follow. In performing the classification task, a test example travels from the root of the tree to a leaf node associated with a single class label. Therefore for a well balanced tree, the time required for evaluation is reduced from $O(DK)$ to $O(D \log K)$, where K is the number of classes and D is the

[1] An early version of this chapter has been presented in [53].

feature dimensionality. The technique can be combined with an embedding technique, so that the evaluation cost can be further reduced to $O(\tilde{D} \log K + D\tilde{D})$ where $\tilde{D} \ll D$ is an embedded label space.

Despite the success of label trees in addressing testing efficiency, the learning technique, critical to ensuring good testing accuracy and efficiency, has several limitations. Learning the tree structure (determining how to split the classes into subsets) involves first training one-vs-all classifiers for all K classes to obtain a confusion matrix, and then using spectral clustering to split the classes into disjoint subsets. First, learning one-vs-all classifiers is costly for large number of classes. Second, the partitioning of classes does not allow overlap, which can be unnecessarily difficult for classification. Third, the tree structure may be unbalanced, which can result in sub-optimal test efficiency.

In this chapter, we address these issues by observing that (1) determining the partition of classes and learning a classifier for each child node in the tree can be performed jointly, and (2) allowing overlapping of class labels among children leads to an efficient optimization that also enables precise control of the accuracy vs efficiency trade-off, which can in turn guarantee balanced trees. This leads to a novel label tree learning technique that is more efficient and effective. Specifically, we eliminate the one-vs-all training step while improving both efficiency *and* accuracy in testing.

4.2 Related Work

Our approach is directly motivated by the label tree embedding technique proposed by Bengio et al. in [26], which is among the few approaches that address sublinear testing cost for multi-class classification problems with a large number of classes and has been shown to outperform alternative approaches including Filter Tree [42] and Conditional Probability Tree (CPT) [43]. Our contribution is a new technique to achieve more efficient and effective learning for label trees. For a comprehensive discussion on multi-class classification techniques, we refer the reader to [26].

Classifying a large number of object classes has received increasing attention in computer vision as datasets with many classes such as ImageNet become available. One line of work is concerned with developing effective feature representations [95, 96, 97, 98] and achieving state of the art performances. Another direction of work, explores methods for exploiting the structure *between* object classes. In particular, it has been observed that object classes can be organized in a tree-like structure both semantically and visually [25, 36, 52], making tree based approaches especially attractive. Our work follows this direction, focusing on effective learning methods for building tree models.

Our framework of explicitly controlling accuracy or efficiency is connected to Weiss et al.'s work [99] on building a cascade of graphical models with increasing complexity for structured prediction. Our work differs in that we reduce the *label* space instead of the model space.

4.3 Label Tree and Label Tree Learning by Bengio et al.

Here we briefly review the label tree learning technique proposed by Bengio et al. and then discuss the limitations we attempt to address.

A label tree is a tree $T = (V, E)$ with nodes V and edges E. Each node $r \in V$ is associated with a set of class labels $\kappa(r) \subseteq \{1, \ldots, K\}$. Let $\sigma(r) \subset V$ be the its set of children. For each child c, there is a linear classifier $w_c \in \mathbb{R}^D$ and we require that its label set is a subset of its parent's, that is, $\kappa(c) \subseteq \kappa(r), \forall c \in \sigma(r)$.

To make a prediction given an input $x \in \mathbb{R}^D$, we use Algorithm 1. We travel from the root until we reach a leaf node, at each node following the child that has the largest classifier score. There is a slight difference than the algorithm in [26] in that the leaf node is not required to have only one class label. If there is more than one label, an arbitrary label from the set is predicted.

Algorithm 1 Predict the class of x given the root node r

```
$s \leftarrow r$.
while $\sigma(s) \neq \emptyset$ do
    $s \leftarrow \arg\max_{c \in \sigma(s)} w_c^T x$
end while
return an arbitrary $k \in \kappa(s)$ or NULL if $\kappa(s) = \emptyset$.
```

Learning the tree structure is a fundamentally hard problem because brute force search for the optimal combination of tree structure and classifier weights is intractable. Bengio et al. [26] instead propose to solve two subproblems: learning the tree structure and learning the classifier weights. To learn the tree structure, K one versus all classifiers are trained first to obtain a confusion matrix $C \in \mathbb{R}^{K \times K}$ on a validation set. The class labels are then clustered into disjoint sets by spectral clustering with the confusion between classes as affinity measure. This procedure is applied recursively to build a complete tree. Given the tree structure, all classifier weights are then learned jointly to optimize the misclassification loss of the tree.

We first analyze the cost of learning by showing that training, with m examples, K classes and D dimensional feature, costs $O(mDK)$. Assume optimistically that the optimization algorithm converges after only one pass of the data and that we use first order methods that cost $O(D)$ at each iteration, with feature dimensionality D. Therefore learning one versus all classifiers costs $O(mDK)$. Spectral clustering only depends on K and does not depend on D or m, and therefore its cost is negligible. In learning the classifier weights on the tree, each training example is affected by only the classifiers on its path, i.e., $O(Q \log K)$ classifiers, where $Q \ll K$ is the number of children for each node. Hence the training cost is $O(mDQ \log K)$. This analysis indicates that learning K one versus all classifiers dominates the cost. This is undesirable in large scale learning because with bounded time, accommodating a large number of classes entails using less expressive and lower dimensional features.

Moreover, spectral clustering only produces disjoint subsets. It can be difficult to learn a classifier for disjoint subsets when examples of certain classes cannot be reliably classified to one subset. If such mistakes are made at higher level of the tree, then it is impossible to recover later. Allowing overlap potentially yields more flexibility and avoids such errors. In addition, spectral clustering does not guarantee balanced clusters and thus cannot ensure a desired speedup. We seek a novel learning technique that overcomes these limitations.

4.4 New Label Tree Learning

To address the limitations, we start by considering simple and less expensive alternatives of generating the splits. For example, we can sub-sample the examples for one-vs-all training, or generate the splits randomly, or use a human constructed semantic hierarchy (e.g., WordNet [37]). However, as shown in [26], improperly partitioning the classes can greatly reduce testing accuracy and efficiency. To preserve accuracy, it is important to split the classes such that they can be easily separated. To gain efficiency, it is important to have balanced splits.

We therefore propose a new technique that jointly learns the splits and classifier weights. By tightly coupling the two, this approach eliminates the need of one-vs-all training and brings the total learning cost down to $O(mDQ \log K)$. By allowing overlapping splits and explicitly modeling the accuracy and efficiency trade-off, this approach also improves testing accuracy and efficiency.

Our approach processes one node of the tree a time, starting with the root node. It partitions the classes into a fixed number of child nodes and learns the classifier weights for each of the children. It then recursively repeats for each child.

In learning a tree model, accuracy and efficiency are inherently conflicting goals and some trade-off must be made. Therefore we pose the optimization problem as maximizing efficiency given a constraint on accuracy, i.e., requiring that the error rate cannot exceed a certain threshold. Alternatively one can also optimize accuracy given efficiency constraints. We will first describe the accuracy constrained optimization and then briefly discuss the efficiency constrained variant. In practice, one can choose between the two formulations depending on convenience.

For the rest of this section, we first express all the desiderata in one single optimization problem (Section 4.4.1), including defining the optimization variables (classifier weights and partitions), objectives (efficiency) and constraints (accuracy). Then in Section 4.4.2& 4.4.3 we show how to solve the main optimization by alternating between learning the classifier weights and determining the partitions. We then summarize the complete algorithm (Section 4.4.4) and conclude with an alternative formulation using efficiency constraints (Section 4.4.5).

4.4.1 Main Optimization

Formally, let the current node r represent classes labels $\kappa(r) = \{1, \dots, K\}$ and let Q be the specified number of children we wish to follow. The goal is to determine: (1) a partition matrix $P \in \{0, 1\}^{Q \times K}$ that represents the assignment of classes to the children, i.e., $P_{qk} = 1$ if class label k appear in child q and $P_{qk} = 0$ otherwise; (2) the classifier weights $w \in \mathbb{R}^{D \times Q}$, where a column w_q is the classifier weights for child $q \in \sigma(r)$,

We measure accuracy by examining whether an example is classified to the correct child, i.e., a child that includes its true class label. Let $x \in \mathbb{R}^D$ be a training example and $y \in \{1, \dots, K\}$ be its true label. Let $\hat{q} = \arg\max_{q \in \sigma(r)} w_q^T x$ be the child that x follows. Given w, P, x, y, the classification loss at the current node r is then

$$L(w, x, y, P) = 1 - P(\hat{q}, y). \tag{4.1}$$

Note that the final prediction of the example is made at a leaf node further down the tree, if the child to follow is not already a leaf node. Therefore L is a lower bound of the actual loss. It is thus important to achieve a smaller L because it could be a bottleneck of the final accuracy.

We measure efficiency by how fast the set of possible class labels shrinks. Efficiency is maximized when each child has a minimal number of class labels so that an unambiguous prediction can be made, otherwise we incur further cost for traveling down the tree. Given a test example, we define *ambiguity* as our efficiency measure, i.e., the size of label set of the child that the example follows, relative to its parent's size. Specifically, given w and P, the ambiguity for an example x is

$$A(w, x, P) = \frac{1}{K} \sum_{k=1}^{K} P(\hat{q}, k). \tag{4.2}$$

Note that $A \in [0, 1]$. A perfectly balanced K-nary tree would result in an ambiguity of $1/K$ for all examples at each node.

One important note is that the classification loss (accuracy) and ambiguity (efficiency) measures as defined in Eqn. 4.1 and Eqn. 4.2 are *local* to the current node being considered in greedily building the tree. They serve as proxies to the *global* accuracy and efficiency of the entire tree. For the rest of this chapter, we will omit the "local" and "global" qualifications if it is clear according to the context.

Let $\epsilon > 0$ be the maximum classification loss we are willing to tolerate. Given a training set $(x_i, y_i), i = 1, \ldots, m$, we seek to minimize the average ambiguity of all examples while keeping the classification loss below ϵ, which leads to the following optimization problem:

OP1. Optimizing efficiency with accuracy constraints.

$$\begin{aligned} \underset{w,P}{\text{minimize}} \quad & \frac{1}{m} \sum_{i=1}^{m} A(w, x_i, P) \\ \text{subject to} \quad & \frac{1}{m} \sum_{i=1}^{m} L(w, x_i, y_i, P) \leq \epsilon \\ & P \in \{0, 1\}^{Q \times K}. \end{aligned}$$

There are no further constraints on P other than that its entries are integers 0 and 1. We do not require that the children cover all the classes in the parent. It is legal that one class in the parent can be assigned to none of the children, in which case we give up on the training examples from the class. In doing so, we pay a price on accuracy, i.e., those examples will have a misclassification loss of 1. Therefore a partition P with all zeros is unlikely to be a good solution. We also allow overlap of label sets between children. If we cannot classify the examples from a class perfectly into one of the children, we allow them to go to more than one child. We pay a price on efficiency since we make less progress in eliminating possible class labels. This is different from the disjoint label sets in [26]. Overlapping label sets gives more flexibility and in fact leads to simpler optimization, as will become clear in Section 4.4.3.

Directly solving OP1 is intractable. However, with proper relaxation, we can alternate between optimizing over w and over P where each is a convex program.

4.4.2 Learning Classifier Weights w Given Partitions P

Observe that fixing P and optimizing over w is similar to learning a multi-class classifier except for the overlapping classes. We relax the loss L by a convex loss $\tilde{L}$ similar to the hinge loss.

$$\tilde{L}(w, x_i, y_i, P) = \max\{0, 1 + \max_{q \in A_i, r \in B_i} \{w_r^T x_i - w_q^T x_i)\}\}$$

where $A_i = \{q | P_{q,y_i} = 1\}$ and $B_i = \{r | P_{r,y_i} = 0\}$. Here A_i is the set of children that contain class y_i and B_i is the rest of the children. The responses of the classifiers in A_i are encouraged to be bigger than those in B_i, otherwise the loss $\tilde{L}$ increases. It is easily verifiable that $\tilde{L}$ upperbounds L. We then obtain the following convex optimization problem.

OP2. Optimizing over w given P.

$$\underset{w}{\text{minimize}} \quad \lambda \sum_{q=1}^{Q} \|w_q\|_2^2 + \frac{1}{m} \sum_{i=1}^{m} \tilde{L}(w, x_i, y_i, P)$$

Note that here the objective is no longer the ambiguity A. This is because the influence of w on A is typically very small. When the partition P is fixed, w can lower A by classifying examples into the child with the smallest label set. However, the way w classifies examples is mostly constrained by the accuracy cap ϵ, especially for small ϵ. Empirically we also found that in optimizing $\tilde{L}$ over w, A remains almost constant. Therefore for simplicity we assume that A is constant w.r.t w and the optimization becomes minimizing classification loss to move w to the feasible region. We also added a regularization term $\sum_{q=1}^{Q} \|w_q\|_2^2$.

4.4.3 Determining Partitions P Given Classifier Weights w

If we fix w and optimize over P, rearranging terms gives the following integer program.

OP3. Optimizing over P.

$$\begin{aligned} \underset{P}{\text{minimize}} \quad & A(P) = \sum_{q,k} P_{qk} \frac{1}{mK} \sum_{i=1}^{m} \mathbf{1}(\hat{q}_i = q) \\ \text{subject to} \quad & 1 - \sum_{q,k} P_{qk} \frac{1}{m} \sum_{i=1}^{m} \mathbf{1}(\hat{q}_i = q \wedge y_i = k) \leq \epsilon \\ & P_{qk} \in \{0,1\}, \forall q, k. \end{aligned}$$

Integer programming in general is NP-hard. However, for this integer program, we can solve it by relaxing it to a linear program and then taking the ceiling of the solution. We show that this solution is in fact near optimal by showing that the number of non-integers can be very few, due to the fact that the LP has few constraints other than that the variables lie in $[0,1]$ and most of the $[0,1]$ constraints will be active. Specifically we use Lemma 4.4.1 to bound the rounded LP solution in Theorem 4.4.2.

Lemma 4.4.1. *For LP problem*

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & c^T x \\ \text{subject to} \quad & Ax \le b \\ & 0 \le x \le 1, \end{aligned}$$

where $A \in \mathbb{R}^{m \times n}, m < n$, if it is feasible, then there exists an optimal solution with at most m non-integer entries and such a solution can be found in polynomial time.

Proof. Let $x*$ be an optimal solution that an LP solver returns. Let B be the set of indices of entries in $x*$ that are non-integers, $B = \{i : x_i^* \in (0, 1)\}$ and let E be the rest of the indices.

If $|B| \le m$, then we are done. We now consider the case when $|B| > m$.

Let H be the polyhedron $H = \{x_B : c_B^T x_B = c^T x_B^*, A_B x_B = A_B x_B^*, 0 \le x_B \le 1\}$, where A_B is the columns indexed by B. Observe that any x such that $x_B \in H$ and $x_E = x_E^*$, is an also an optimal solution of the LP. That is, replacing the non-integer entries of x^* with those in H still gives an optimal solution.

Since $x_B^* \in H$, therefore H is non-empty. Also H is bounded. Hence there exists at least one *basic feasible solution* x_B' of H [100], for which there are $|B|$ linearly independent constraints that are active. Such a basic feasible solution can be found in polynomial time by solving an auxiliary LP by introducing additional artificial variables, the same as the Phase 1 of the simplex method. Details can be found in [100].

We now show that x_B' has at most m non-integer entries.

We first show that $\forall x_B \in \text{null}(A_B)$, $c_B^T x_B = 0$. Assume to the contrary that there exists $\hat{x}_B \in \text{null}(A_B)$ such that $c_B^T \hat{x}_B < 0$. Let $y^* \in \mathbb{R}^n$ be such that $y_B^* = x_B^* + \theta \hat{x}_B$ and $y_E^* = x_E^*$, where $\theta > 0$. It follows that for sufficiently small θ, y^* satisfies all contraints of the LP, since $Ay^* = Ax^* + \theta A_B \hat{x}_B = Ax^* \le b$ and $0 \le y_B^* = x_B^* + \theta \hat{x}_B \le 1, 0 \le y_E^* = x_E^* \le 1$. Also the LP has a smaller value, since $c^T y^* = c^T x^* + \theta c_B^T \hat{x}_B < c^T x^*$, which is contradition.

If follows that $c_B \in \text{null}(A_B)^\perp = \text{row}(A_B)$. Therefore the number of linearly independent vectors among c_B and rows of A_B is at most m. Since x'_B has $|B| > m$ linearly independt constraints that are active, at least $|B| - m$ constraints from $0 \leq x'_B \leq 1$ must be active and therefore at least $|B| - m$ entries of x'_B are integers. Hence x'_B has at most m non-integer entries.

We then replace the entries x^*_B in x^* with x'_B and obtain an optimal solution with at most m non-integer entries. □

Theorem 4.4.2. *Let A^* be an optimal value of OP3. A solution P' can be computed within polynomial time such that $A(P') \leq A^* + \frac{1}{K}$.*

Proof. We relax OP3 to an LP by replacing the constraint $P_{qk} \in \{0, 1\}, \forall q, k$ with $P_{qk} \in [0, 1], \forall q, k$. Apply Lemma 4.4.1 and we obtain an optimal solution P'' of the LP with at most 1 non-integer. We take the ceiling of the fraction and obtain an integer solution P' to OP3. The value of the LP, a lower bound of A^*, increases by at most $\frac{1}{K}$, since $\frac{1}{mK} \sum_{i=1}^{m} \mathbf{1}(\hat{q}_i = q) \leq \frac{1}{K}, \forall q$. □

Note that the ambiguity is a quantity in $[0, 1]$ and K is the number of classes. Therefore for large numbers of classes the rounded solution is almost optimal.

4.4.4 Summary of Algorithm

Now all ingredients are in place for an iterative algorithm to build the tree, except that we need to initialize the partition P or the weights w. We find that a random initialization of P works well in practice. Specifically, for each child, we randomly pick one class, without replacement, from the label set of the parent. That is, for each row of P, randomly pick a column and set the column to 1. This is analogous to picking the cluster seeds in the K-means algorithm.

We summarize the algorithm for building one level of tree nodes in Algorithm 2. The procedure is applied recursively from the root. Note that each training example only affects classifiers on one path of the tree, hence the training cost is $O(mD \log K)$ for a balanced tree.

Algorithm 2 Grow a single node r

```
Input: $Q$,$\epsilon$ and training examples classified into node $r$ by its ancestors.
Initialize $P$. For each child, randomly pick one class label from the parent, without
replacement.
for $t = 1 \to T$ do
    Fix $P$, solve OP2 and update $w$.
    Fix $w$, solve OP3 and update $P$.
end for
```

4.4.5 Efficiency Constrained Formulations

As mentioned earlier, we can also optimize accuracy given explicit efficiency constraints. Let δ be the maximum ambiguity we can tolerate. Let OP1', OP2', OP3' be the counterparts of OP1, OP2 and OP3. We obtain OP1' by replacing ϵ with δ and switching $L(w, x_i, y_i, P)$ and $A(w, x_i, p)$ in OP1. OP2' is the same as OP2 because we also treat A as constant and minimize the classification loss L unconstrained. OP3' can also be formulated in a straightforward manner, and solved nearly optimally by rounding from LP (Theorem 4.4.3).

Theorem 4.4.3. *Let L^* be the optimal value of OP3'. A solution P' can be computed within polynomial time such that $L(P') \leq L^* + \max_k \psi_k$, where $\psi_k = \frac{1}{m}\sum_{i=1}^{m} \mathbf{1}(y_i = k)$, is the percentage of training examples from class k.*

Proof. We relax OP3' to an LP. Apply Lemma 4.4.1 and obtain an optimal solution P'' with at most 1 non-integer. We take the floor of P'' and obtain a feasible solution P' to $OP3'$. The value of the LP, a lower bound of L^*, increases by at most $\max_k \psi_k$, since $\frac{1}{m}\sum_i \mathbf{1}(\hat{q}_i = q \wedge y_i = k) \leq \frac{1}{m}\sum_{i=1}^{m} \mathbf{1}(y_i = k) \leq \max_k \psi_k, \forall k, q$. □

For uniform distribution of examples among classes, $\max_k \psi_k = 1/K$ and the rounded solution is near optimal for large K. If the distribution is highly skewed, for example, a heavy tail, then the rounding can give poor approximation. One simple workaround is to split the big classes into artificial subclasses or treat the classes in the tail as one big class, to "equalize" the distribution. Then the same learning techniques can be applied. In this work we focus on the near uniform case and leave further discussion on the skewed case as future work.

4.5 Experiments

We use two datasets for evaluation: ILSVRC2010 (see Section 5.4.1) and ImageNet10K (see Section 3.3.1). ILSVRC2010 is created for a large scale recognition challenge held in conjunction of PASCAL [101]. There are 1.2M images from 1K classes for training, 50K images for validation and 150K images for test. For each image in ILSVRC2010 we compute the LLC [95] feature with SIFT on a 10K codebook and use a two level spatial pyramid (1x1 and 2x2 grids) to obtain a 50K dimensional feature vector. In ImageNet10K, there are 9M images from 10184 classes. We use 50% for training, 25% for validation, and the rest 25% for testing. For ImageNet10K, We compute LLC similarly except that we use no spatial pyramid, obtaining a 10K dimensional feature vector.

We use parallel stochastic gradient descent (SGD) [102] for training. SGD is especially suited for large scale learning [38] where the learning is bounded by the time and the features can no longer fit into memory (the LLC features take 80G in sparse format). Parallelization makes it possible to use multiple CPUs to improves wall time.

We compare our algorithm with the original label tree learning method by Bengio et al. [26]. For both algorithms, we fix two parameters, the number of children Q for each node, and the maximum depth H of the tree. The depth of each node is defined as the maximum distance to the root (the root has depth 0). We require every internal node to split into Q children, with two exceptions: nodes at depth $H-1$ (parent of leaves) and nodes with fewer than Q classes. In both cases, we split the node fully, i.e., grow one child node per class. We use $T_{Q,H}$ to denote a tree built with parameters Q and H. We set Q and

H such that for a well balanced tree, the number of leaf nodes Q^H approximate the number of classes K.

We evaluate the *global* classification accuracy and computational cost in both training and test. The main costs of learning consist of two operations, evaluating the gradient and updating the weights, i.e., vector dot products and vector additions (possibly with scaling). We treat both operations as costing the same. [2] To measure the cost, we count the number of *vector* operations performed per training example. For instance, running SGD one-versus-all (either independent or single machine SVMs [103]) for K classes costs $2K$ per example for going through data once, as in each iteration all K classifiers are evaluated against the feature vector (dot product) and updated (addition).

For both algorithms, we build three trees $T_{32,2}$, $T_{10,3}$, $T_{6,4}$ for the ILSVRC2010 1K classes and build one tree $T_{101,2}$ for ImageNet10K classes. For the Bengio et al. method, we first train one-versus-all classifiers with one pass of parallel SGD. This results in a cost of $2{,}000$ per example for ISVRC2010 and $20{,}368$ for ImageNet10K. After forming the tree skeleton by spectral clustering using confusion matrix from the validation set, we learn the weights by solving a joint optimization (see [26]) with two passes of parallel SGD. For our method, we do three iterations in Algorithm 2. In each iteration, we do one pass of parallel SGD to solve OP3', such that the computation is comparable to that of Bengio et al. (excluding the one-versus-all training). We then solve OP3' on the validation set to update the partition. To set the efficiency constraint, we measure the average (local) ambiguity of the root node of the tree generated by the Bengio et al. approach, on the validation set. We use it as our ambiguity cap throughout our learning, in an attempt to produce a similarly structured tree.

We report the test results in Table 4.1. The results show that for all types of trees, our method achieves comparable or significantly better accuracy while achieving better speedup with much less training cost, even after excluding the 1-versus-all training in Ben-

[2]This is inconsequential as a vector addition always pairs with a dot product for all training in this work.

	$T_{32,2}$			$T_{10,3}$			$T_{6,4}$			$T_{101,2}$		
	$Acc\%$	C_{tr}	S_{te}	$Acc\%$	C_{tr}	S_{te}	$Acc\%$	C_{tr}	S_{te}	$Acc\%$	C_{tr}	S_{te}
Ours	**11.9**	**259**	**10.3**	**8.92**	**104**	**18.2**	5.62	**50.2**	**31.3**	**3.4**	**685**	**32.4**
[26]	8.33	321	**10.3**	5.99	193	15.2	**5.88**	250	9.32	2.7	1191	**32.4**

Table 4.1: Global accuracy (Acc), training cost (C_{tr}), and test speedup (S_{te}) on ILSVRC2010 1K classes ($T_{32,2}, T_{10,3}, T_{6,4}$) and on ImageNet10K($T_{101,2}$) classes. Training and test costs are measured as the average number of vector operations performed per example. Test speedup is the one-vs-all test cost divided by the label tree test cost. Ours outperforms the Bengio et al. [26] approach by achieving comparable or better accuracy and efficiency, with less training cost, compared with the training cost for Bengio et al. [26] with the one-vs-all training cost excluded.

Tree		$T_{32,2}$		$T_{10,3}$			$T_{6,4}$			
Depth		0	1	0	1	2	0	1	2	3
Classification loss (%)	Ours	49.9	76.1	34.6	52.6	71.2	30.0	48.8	55.9	64.4
	Bengio [26]	76.6	64.8	62.8	53.7	65.3	56.2	34.8	37.3	65.8
Ambiguity (%)	Ours	6.49	1.55	18.9	18.4	2.96	24.7	24.1	23.5	7.15
	Bengio [26]	6.49	1.87	19.0	25.9	2.95	24.7	59.6	56.5	2.02

Table 4.2: Local classification loss (Eqn. 4.1) and ambiguity (Eqn. 4.2) measured at different depth levels for all trees on the ILSVRC2010 test set (1K classes). $T_{6,4}$ of Bengio et al. is less balanced (large ambiguity). Our trees are more balanced as efficiency is explicitly enforced by capping the ambiguity throughout all levels.

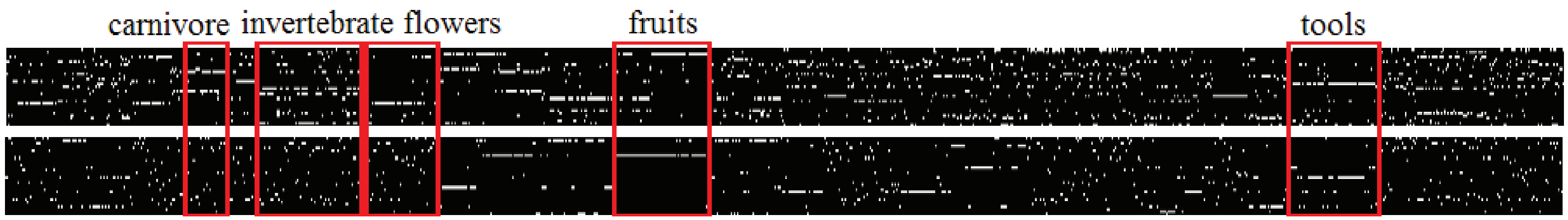


Figure 4.1: Comparison of partition matrices ($32 \times 1{,}000$) of the root node of $T_{32,2}$ for our approach (top) and the Bengio et al. approach (bottom). Each entry represents the membership of a class label (column) in a child (row). The columns are ordered by a depth first search of WordNet. Columns belonging to certain WordNet subtrees are marked by red boxes.

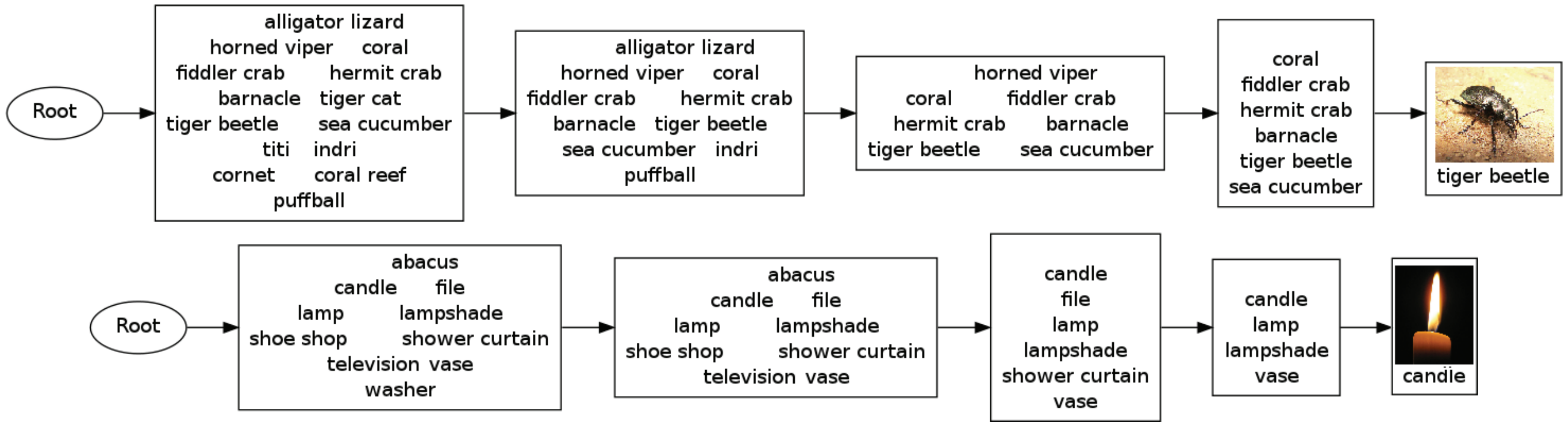


Figure 4.2: Paths of the tree $T_{6,4}$ taken by two test examples. The class labels shown are randomly subsampled to fit into the space.

gio et al.'s. In particular, our training is 31 times faster on 10K classes than Bengio et al.'s—our cost is 685 per examples whereas theirs is $20{,}386$ for one-versus-all training plus $1{,}191$ for the label tree learning . It's worth noting that for the Bengio et al. approach, $T_{6,4}$ fails to further speedup testing compared to the other shallower trees. The reason is that at depth 1 (one level down from root), the splits became highly imbalanced and does not shrink the class sets faster enough until the height limit is reached. This is revealed in Table 4.2, where we measure the average local ambiguity (Eq. 4.2) and classification loss (Eq. 4.1) at each depth on the test set to shed more light on the structure of the trees. Observe that our trees have almost constant average ambiguity at each level, as enforced in learning. This shows an advantage of our algorithm since we are able to explicitly enforce balanced tree while in Bengio et al. [26] no such control is possible, although spectral clustering encourages balanced splits.

In Figure 4.1, we visualize the partition matrices of the root of $T_{32,2}$, for both algorithms. The columns are ordered by a depth first search of the WordNet tree so that neighboring columns are likely to be semantic similar classes. We observe that for both methods, there is visible alignment of the WordNet ordering. We further illustrate the semantic alignment by showing with the paths of our $T_{6,4}$ traveled by two test examples. Also observe that our partition is notably "noisier," despite that both partitions have the same average ambiguity. This is a result of overlapping partitions, which in fact improves accuracy (as shown in Table 4.2) because it avoids the mistakes made by forcing all examples of a class commit to one child.

Also note that Bengio et al. showed in [26] that optimizing the classifiers on the tree jointly is significantly better than independently training the classifiers for each node, as it encodes the dependency of the classifiers along a tree path. This does not contradict our results. Although we have no explicit joint learning of classifiers over the entire tree, we train the classifiers of each node using examples already filtered by classifiers of the ancestors, thus implicitly enforcing the dependency.

Finally it's worth noting that the accuracies achieved here are not directly comparable to those in Chapter 3. The computational resource in these experiments is much more restricted than those in Chapter 3. Moreover, these experiments are coded in Matlab while those in Chapter 3 in C. Nonetheless the experiments demonstrate that the new learning algorithm improves Bengio et al. [26], the previous state of the art approach.

4.6 Summary

We have presented a novel approach to efficiently learning a label tree for large scale classification with many object classes. The key contribution of the approach is a technique to simultaneously determine the structure of the tree and learn the classifiers for each node in the tree. This approach also allows fine-grained control over the efficiency vs accuracy trade-off in designing the label tree. Experiments are performed on large scale image classification with 10,184 classes and 9 million images. We have demonstrated significant improvements in test accuracy and efficiency with a 31 times reduction of training time and more balanced trees compared to the previous state of the art by Bengio et al [26].

Chapter 5

Hierarchy-Aware Large Scale Retrieval

5.1 Introduction

This chapter studies the problem of similarity retrieval – given a query image, the task is to find similar images from a large image collection. [1] It is closely related to large scale visual recognition, especially when semantic similarity is considered.

In this study we focus on exploiting the semantic relations between categories to improve large scale retrieval, as depicted in Figure 5.1. As illustrated there, results show that exploiting hierarchical relationships can significantly improve retrieval accuracy. Incorporating hierarchical relationships is becoming more important as datasets grow larger. The potential benefit is largest when categories are sampled "densely" and fine-grained distinctions must be made (e.g., [104, 52]). In order to handle such large scale data, computational efficiency and scalability is a critical aspect of effective using hierarchy in retrieval.

Our approach demonstrates how to effectively incorporate prior human knowledge in the form of a hierarchical structure defined on semantic attributes of images. For instance given semantic attributes like containing a horse, dog, or windmill, a predefined hierarchy might let us know that an image containing a horse would be more similar to one contain-

[1]An early version of this chapter has been presented in [54].

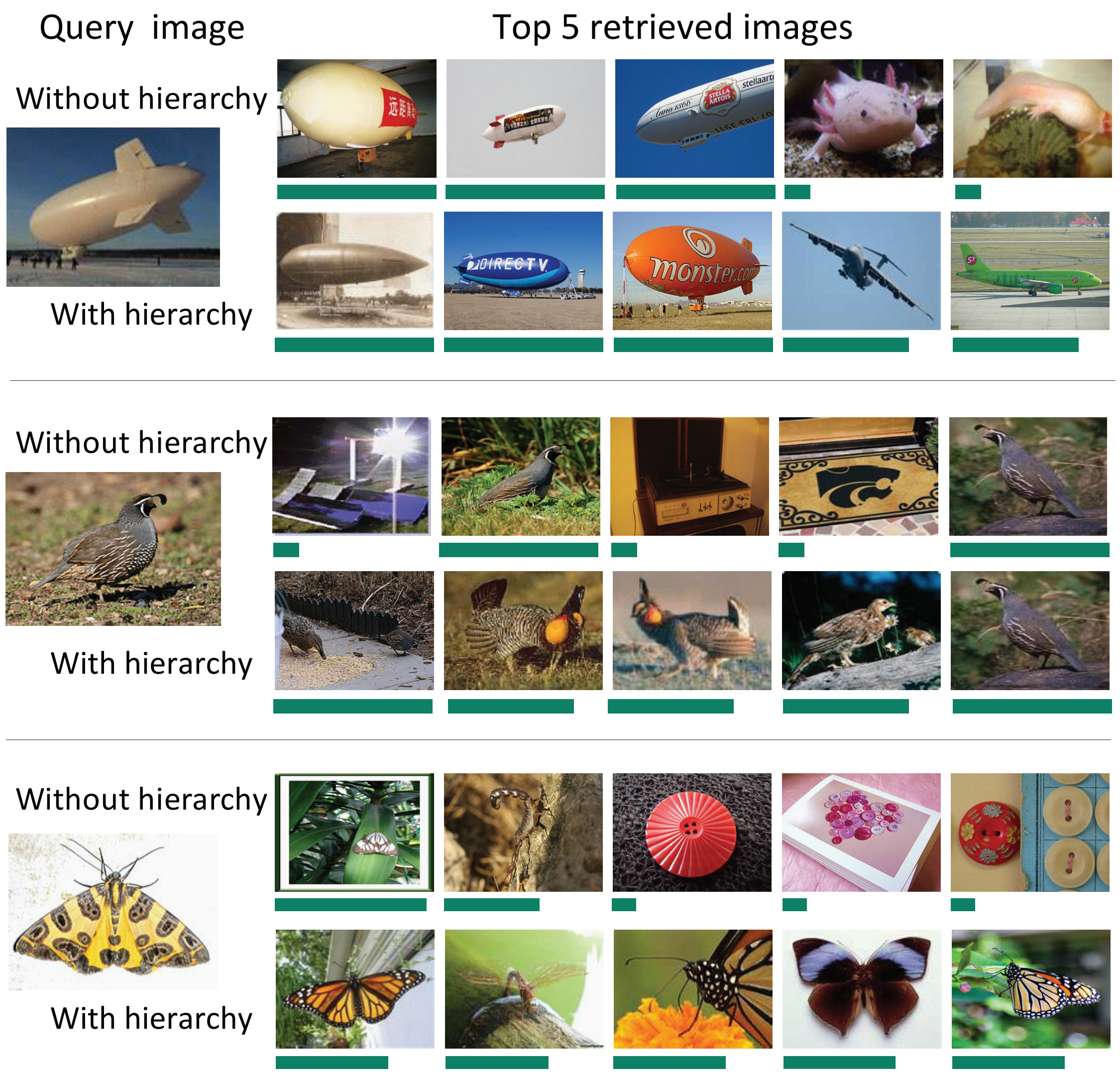


Figure 5.1: Images retrieved by exploiting hierarchy versus those without considering hierarchy. Green bars show ground truth similarity to the query, defined based on the category hierarchy (see Section 5.4.2). Longer bars indicate more similarity.

ing a dog than to an image containing a windmill. It is feasible to specify a hierarchical structure in terms of semantic attributes, but may be quite difficult to do so directly in terms of low level features.

The current state of the art for similar image retrieval stems from a strong line of work on learning the underlying similarity function used for retrieval [105, 106, 107]. In that work, the goal is to learn a function that computes similarity directly from low level feature vectors extracted from images, and does not allow variable measures of similarity that could encode hierarchical structure. It may be possible to adapt some of those strategies to take into account variable similarities for hierarchical structure, but would require modification of the techniques, and would not necessarily improve the scalability or parallelism of the approaches.

Our approach takes a different track, *learning to recognize semantic attributes of images, and then using a predefined comparison function – based on a known hierarchal structure – to produce a similarity score for retrieval.* Learning to recognize semantic attributes can be easily parallelized, making this approach very scalable. That this approach requires labeled data for semantic attributes is potentially limiting, but in practice almost every single experiment in the related work on similarity learning begins from data with labels, such as the categories in Caltech256, or queries that produced the images in OASIS [105]. Furthermore, for non-overlapping categories, it is possible to reconstruct the category labels directly from the training data used for OASIS [105], LMNN [106], MCML [107], and other techniques. We show that significant improvements over the state of the art are possible when labels and a hierarchy are known or when labels can be inferred but hierarchy is not available. [2] Nevertheless, when labels are truly un-available and cannot be inferred, the proposed technique will not be appropriate or optimal.

Once a similarity function is determined the next challenge is efficient retrieval of the most similar database images for a query, with respect to the hierarchical similarity. This chapter presents *a novel hashing strategy that provides a sublinear time solution for retrieval and forms a generally usable component on its own.* When combined with the training for the semantic classifiers that is linear in the input data and inherently parallelizable, the overall system is very scalable. To make this concrete, our semantic index structure can be built on 600,000 images in 14 days on a single CPU, or because of the easy parallelizability, in 20 minutes of wall clock time using 1,000 CPUs. Using hashing, retrieval of similar images in the resulting index can be performed in 3 milliseconds per

[2]This is actually the case in most work on similarity learning [106, 107, 105].

query with accuracy close to 90% of brute force search and computational cost less than 0.001 times that of brute force.

5.2 Related Work

We review closely related work on hierarchy, similarity learning, semantic indexing, and hashing for retrieval.

Work on recognition in computer vision has reached the scale – in terms of number of classes – where hierarchical relationships between classes begin to 1) be non-trivial, 2) have an impact on recognition performance, and 3) have the potential to improve recognition accuracy. This has been demonstrated by work putting existing datasets into hierarchies [25], and building large new datasets – e.g., TinyImages [36] and the ImageNet dataset presented in this dissertation– based on the hierarchical semantic structure in WordNet [37] a major project of the linguistics and natural language processing community. This line of work has begun to reveal both the effects of hierarchical structure on classification accuracy [104, 36, 52],and hints at the promise of exploiting such structure for classification when evaluated in terms of the hierarchy [52] as well as showing improved classification given very small amounts of training data [30].

In this work we demonstrate that it is possible to exploit hierarchical structure for a different but related task – similar image retrieval – gaining significant improvements in accuracy. This complements recent work advancing the learning of similarity functions, especially for retrieval, e.g. [107, 106, 105], that does not yet address hierarchy. Some of our experiments compare with OASIS from Chechik et al. [105], the current state of the art in learning similarity functions for retrieval, [3] and we demonstrate significant improvement by adding hierarchy. Furthermore the proposed techniques are easily parallelizable, allowing better scaling than [105] (even without hierarchy) which already improved com-

[3]Closely related in technique to [108] for classification.

putational efficiency significantly over other techniques [107, 106].

Many of the improvements shown stem from exploiting high level knowledge in the form of a semantic hierarchy. This is related to recent research in explicitly estimating high level semantic attributes for recognition [109, 110, 111, 112, 113]. In particular [109] allows retrieval queries using language to refer to the semantic attributes of faces. We consider queries specified by an image, and add a hierarchical relationship between semantic attributes. Recent work [113] considers a representation similar in spirit to the semantic representation we use, but focuses on using the representation for classification – using multiple training examples for a class specific query, as opposed to a single example as a query – instead of retrieval. There is also related work from the multimedia and information retrieval community, especially on TrecVid, e.g., [114, 115] (and references therein), that explicitly train object or concept "detectors" and use their output as features for retrieval based on high level (or textual) queries.

Efficient retrieval with respect to a similarity function is important for very large scale settings. Significant work has been done on hashing for the related problem of finding approximate nearest neighbors [55, 116]. In our setting, retrieval using bilinear similarity on vectors of probabilities is a core subroutine, and we introduce a novel hashing scheme to accomplish this. Note that this is a data independent hashing approach in contrast to recent approaches based on learning hashing functions for vision [117, 118, 119].

5.3 Approach

5.3.1 Exploiting Hierarchy

In order to exploit hierarchical knowledge in retrieval, we consider the core subroutine of evaluating the similarity Sim between two images. Retrieval consists of finding the images from a large database with greatest similarity to a query image. In previous work, much of the effort in building a system for retrieval was in learning the similarity func-

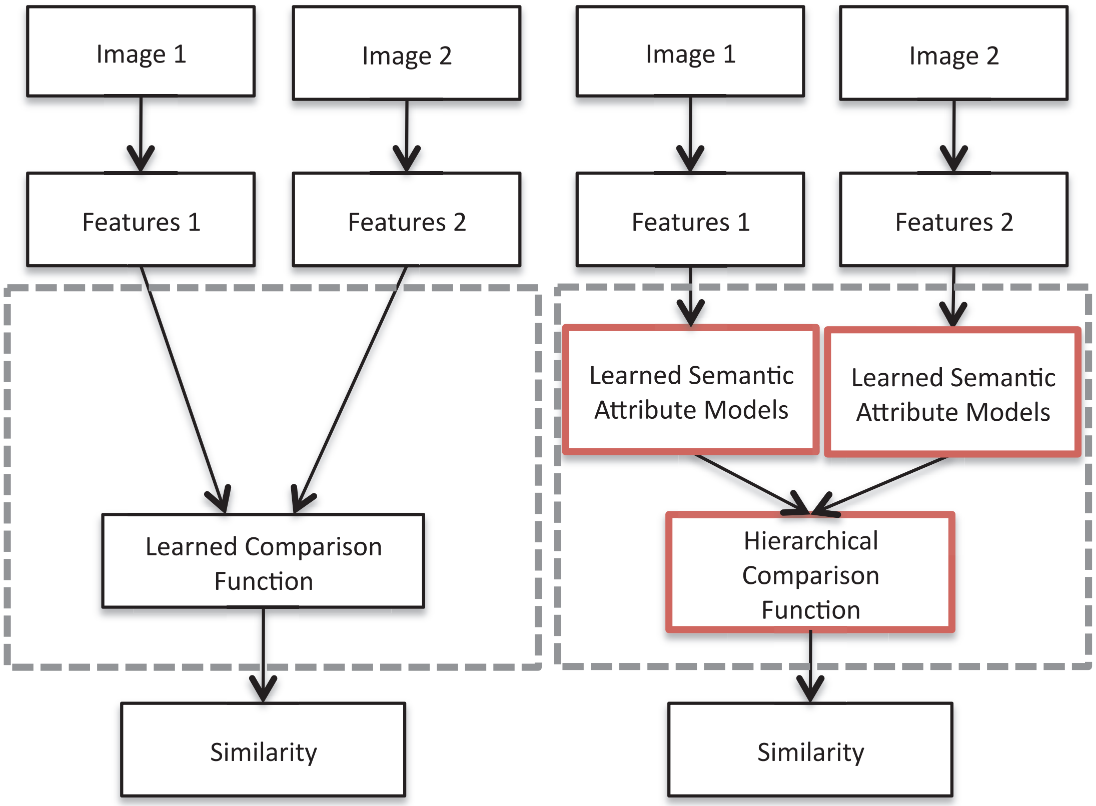


Image 1
Image 2
Image 1
Image 2
Features 1
Features 2
Features 1
Features 2
Learned Semantic Attribute Models
Learned Semantic Attribute Models
Learned Comparison Function
Hierarchical Comparison Function
Similarity
Similarity

Figure 5.2: Compared to previous work (left) our approach to learning similarity functions (right) separates the learned similarity function into two parts, an estimate of probabilities for semantic labels, and a hierarchical comparison function. The hierarchical comparison function is deterministically built from prior knowledge and only the semantic models are learned.

tion that mapped low-level image features computed from image to a similarity value. For images a,b,c, and d, let $f(\cdot)$ denote their low level features, then training can be performed by considering constraints on pairs, e.g., a and b are more similar than c and d so $Sim(f(a), f(b)) > Sim(f(c), f(d)) + 1$ as in [106]. The state of the art OASIS [105] considers triples of images, where a was supposed to be more similar to b than to c, yielding the constraint $Sim(f(a), f(b)) < Sim(f(a), f(c)) + 1$. These constraints were used to *learn* a matrix L so that $Sim(f(a), f(b)) = f(a)'Lf(b)$.

As illustrated in Figure 5.2, our approach computes similarity by first estimating probabilities of semantic attributes for an image $s(f(a))$, based on low level image features.

Then we use the prior knowledge of the semantic hierarchical relationship to deterministically compute a hierarchical similarity matrix, S, and the similarity function is $Sim(a, b) = s(f(a))^T Ss(f(b))$ (Section 5.3.1.1 & 5.3.1.2). Not only does this allow exploiting hierarchical knowledge, but learning is only needed to build the models for semantic attributes – a process that is much easier to parallelize than previous approaches to learning similarity.

5.3.1.1 Encoding hierarchy in semantic similarity

The core of our approach is to use prior knowledge of a hierarchy between semantic attributes to compute similarity for retrieval. We start by discussing a non-probabilistic version of such a similarity, and describe an image a by a set of binary semantic attributes $\{1 \dots K\}$. The attributes can be object categories ("is dog"), part relations ("has legs"), visual descriptions ("is black") or in fact any predicate about the image. We will mainly focus on object class category attributes as they are the dominant type of attributes currently used and have been extensively studied, but the approach will extend to arbitrary attributes.

Given the attributes, the similarity between two images a and b can then be measured as how well their attributes match. Specifically, let $\delta_i(a) \in \{0, 1\}$ be the indicator function of image a having attribute i. We define the similarity as $Sim^*(a, b) = \sum_{i,j} \delta_i(a) S_{ij} \delta_j(b)$, where $S \in \mathbb{R}^{K \times K}$ and S_{ij} is a "matching score" between attribute i and j, i.e., the semantic similarity based on prior human knowledge. We refer to S as *the prior matrix*.

This is a very general form and encompasses a large class of semantic similarities. For object category attributes, a dominate relationship is the "is a" relation that naturally organizes them into a semantic hierarchy. In this case, S can be derived by measuring the closeness of categories relative to the hierarchy. For instance, let $S(i, j) = \xi(\pi(i, j))$, where $\pi(i, j)$ is the lowest common ancestor of category i and j and $\xi(\cdot) : \{1 \dots K\} \to \mathbb{R}$ is some real function that is non-decreasing going down the hierarchy,i.e., the lower the lowest shared ancestor, the more similar categories i and j are. For example, "is donkey" is much more similar to "is horse" than "is keyboard" because "donkey" shares a lower level common ancestor “equine” with “horse” than “object” with “keyboard.” More concretely $\xi(\cdot)$ can be based on the height of the node [46]. Note that there are other possible ways to obtain similarity between attributes such as automatic text mining[120] when such a manually constructed hierarchy is not available.

A special case is when the attributes are mutually exclusive categories and S is the identity matrix, so $Sim^*(a, b)$ simply indicates whether a and b belong to the same category. Refer to this as a “flat” setting as there is no hierarchical relationship between the attributes. The attributes are treated as either identical or different and a retrieval system optimized for this similarity would be incapable of ranking “horse” higher than “keyboard” given a query “donkey.” This is setting where most existing techniques were developed and evaluated [107, 106, 105].

So far our similarity employs hard assignment of binary attributes. However there is often uncertainty in representing images with semantic attributes. On one hand, natural language is inherently ambiguous and categories overlap. There will always be objects that evade exact categorization. Also perfect classification of semantic attributes is unrealistic. Thus instead of using binary indicators, we represent an image a as a vector $s(f(a)) = x \in \mathbb{R}^K$ where $x_i = \Pr(\delta_i(a) = 1|a)$, i.e., the probability that image x has attribute i. Given image a, b and their vector of probabilities x, y, we redefine the similarity to be the expectation of the non-probabilistic version, i.e. $Sim(a, b) = \mathbb{E}\, Sim^*(a, b) = \sum_{i,j} x_i S_{ij} y_j$, or simply $Sim(a, b) = x^T Sy$. This is assuming that image a and b are drawn independently, as is valid for most retrieval settings. We will refer to this form of similarity $x^T Sy$ as *bilinear similarity*.

Note that although we have mainly used mutually exclusive object categories as attributes in our discussion and will also focus on this case in the experiments due to availability of datasets, our formulation is not restricted to mutually exclusive categorization of images. An image can have multiple objects and thus any number of attributes “turned on.”

5.3.1.2 Learning semantic attributes

Once the prior matrix S is given, to compute the bilinear similarity $x^T Sy$, a critical step is to learn models of semantic attributes and obtain probabilities. For large scale retrieval, important considerations are scalability and efficiency of learning, as real world retrieval systems need to handle tens of thousands of semantic attributes and to train from very large datasets.

To obtain the probabilistic attribute representation, we first learn binary classifiers for each semantic attribute independently. For example, in the case of category attributes, we train 1-vs-all linear SVM for each category. Then we calibrate the outputs of the classifiers into probabilities. In our experiments, we fit a sigmoid function to each SVM classifier [121] to convert the output into a probability. For non-overlapping categories, we further normalize the probabilities to form a vector whose entries sum to one. [4]

Note that both steps are easily parallelizable as the classifiers and sigmoid functions can be learned independently. Also learning the semantic attributes is decoupled from the specification of the prior matrix S. In contrast to existing similarity learning algorithms that learn similarity from low level features, our scheme can be adapted to new similarity measures by simply replacing the prior matrix in retrieval time, without relearning of the attribute models.

5.3.2 Efficient indexing

Efficiency is a major challenge for large scale retrieval Merely considering object categories as attributes may result in a probability vector of tens of thousands dimensions [2]. Computing the similarity between a query and each database image thus becomes prohibitively expensive.

[4]Note that the probabilities can be made more accurate by joint calibration. For example, for non-overlapping categories one can use random forest of probability estimation trees (PET) [122] to obtain more accurate multiclass probabilities. We find that for large training data the simplicity and efficiency outweighs the marginal accuracy gain from PETs.

We introduce a novel technique based on locality sensitive hashing (LSH) [55] to achieve sublinear retrieval time for bilinar similarity. The key is to construct on a family of hash functions $\mathcal{H}$ such that $\Pr_{h \in \mathcal{H}}(h(x) = h(y)) = Sim(x, y)$ [123] or $\Pr(h_1(x) = h_2(y)) = Sim(x, y)$ with h_1 and h_2 drawn independently [116]. For a query point y, one retrieves the database points from the bin of $h(y)$ and rerank them to produce the final results. In practice, one may concatenate multiple hash functions to reduce false collision and use multiple hash tables to increase recall.

For our bilinear similarity $Sim(x, y) = x^T S y$ where x and y are vectors of probabilities, we provide theoretical results on sufficient conditions of S and corresponding construction techniques, informally:

- If S is element-wise non-negative, symmetric and diagonally dominant, then there exists a construction (Lemma 5.3.2).
- If S is derived from a hierarchy such that classes sharing lower common ancestors have higher similarity, then there exists a construction (Lemma 5.3.8).

We present the details of the hashing constructions in Section 5.3.2.1. It is worth noting that the hashing construction for a special case, where the semantic attributes are non-overlapping category labels and S is identity matrix, is especially simple: $h(x)$ is an integer from 1 to K sampled according to the multinomial distribution x. In implementation, h is parametrized by a uniformly drawn real number $p \in [0, 1)$ and returns the index of the interval where p falls in x.

One closely related existing technique is the random hyperplane LSH [123] for approximating cosine similarity $Sim(x, y) = \frac{x^T y}{\|x\| \|y\|}$ that measures the angle between x and y, different from ours due to the L2 normalization. We compare empirical performance with it in Section 5.4.5.

5.3.2.1 Hashing Constructions

In this section we first provide proofs and hashing constructions for probability vectors of non-overlapping categories (Lemma 5.3.2– 5.3.8), i.e., $x \in \mathbb{R}^K, \sum_i x_i = 1, 0 \le x_i \le 1$ for $i = 1, \ldots, K$. We use Δ^{K-1} to denote the set of all such vectors. In Lemma 5.3.10, we show extension to the general case where $x \in \mathbb{R}^K, 0 \le x_i \le 1$ for $i = 1, \ldots, K$ (but does not necessarily sum to one). We use $\tilde{\Delta}^{K-1}$ to denote the set of all such vectors.

Definition 5.3.1. A matrix $S \in \mathbb{R}^{K \times K}$ is *hashable*, if there exists a $\lambda_S > 0$ and, for any $\epsilon > 0$, a distribution on a family $\mathcal{H}(S, \epsilon)$ of hash functions $h(\cdot; S, \epsilon)$ such that for any $x, y \in \Delta^{K-1}$,

$$0 \le \Pr\left(h_1(x; S, \epsilon) = h_2(y; S, \epsilon)\right) \; - \; \lambda_S \cdot x^T S y \le \epsilon$$

where h_1 and h_2 are drawn independently from $\mathcal{H}(S, \epsilon)$.

Here we relax the equality in the LSH condition $\Pr(h_1(x) = h_2(y)) = Sim(x, y)$ to equality up to ϵ. This has virtually no practical impact because in all of our constructions ϵ can be easily made negligibly small, without incurring any additional computational cost. Also note that scaling S does not affect the ranking induced by the similarity $x^T S y$.

Lemma 5.3.2. *If S is symmetric, element-wise non-negative and diagonally dominant, that is,*
$\forall i = 1, \ldots, K, \; s_{ii} \ge \sum_{j \ne i} s_{ij}$*, then S is hashable.*

Proof. Define a $K \times (K+1)$ matrix $\Theta = (\theta_{ij})$, where

$$\begin{aligned}
\theta_{ij} &= \sqrt{\hat{s}_{ij}}, \; \forall i = 1, \ldots, K, \; \forall j = 1, \ldots, K, \; i \ne j. \\
\theta_{ii} &= \sqrt{\hat{s}_{ii} - \sum_{j \ne i} \hat{s}_{ij}}, \; \forall i = 1, \ldots, K. \\
\theta_{i,K+1} &= 1 - \sum_{j=1}^{K} \theta_{ij}, \; \forall i = 1, \ldots, K.
\end{aligned}$$

where $\hat{S} = \lambda_S \cdot S$ with λ_S chosen to ensure $\theta_{i,K+1} \geq 0$. Note that each row of Θ sums to one. Also note that $\theta_{ij} = \theta_{ji}, \forall i, j \leq K$ due to the symmetry of S.

Consider hash functions $h(x)$ that map a probability vector to a set of positive integers, that is, $h : \Delta^{K-1} \rightarrow 2^{\mathbb{N}}$ where $2^{\mathbb{N}}$ is all subsets of natural numbers. Note that $h(x) = h(y)$ is defined as *set equality*, that is, the ordering of elements does not matter.

To construct $\mathcal{H}(S, \epsilon)$, let $N \geq 1/\epsilon$. Then $h(x; S, \epsilon)$ is computed as follows:

1. Sample $\alpha \in \{1, \ldots, K\} \sim multi(x)$
2. Sample $\beta \in \{1, \ldots, K+1\} \sim multi(\theta_\alpha)$ where θ_α is the α^{th} row of Θ.
3. If $\beta \leq K$, return $\{\alpha, \beta\}$
4. Randomly pick γ from $\{K+1, \ldots, K+N\}$, return $\{\gamma\}$.

In implementation, h is parametrized by three uniformly drawn values $p, q \in [0, 1]$ and $r \in \{1 \ldots N\}$, used respectively in the sampling process for α, β and γ.

Let x, y be probability vectors, $x, y \in \Delta^{K-1}$. Let $\alpha_x, \beta_x, \gamma_x$ be the values sampled when computing $h(x)$, and similarly for $\alpha_y, \beta_y, \gamma_y$. To compute $\Pr(h(x) = h(y))$, consider two cases below.

Case 1: Suppose $\alpha_x = i \in \{1, \ldots, K\}$, $\alpha_y = j \in \{1, \ldots, K\}$, $i \neq j$. Then

$$\begin{aligned}
&\Pr(h(x) = h(y) \mid \alpha_x = i \wedge \alpha_y = j) \\
&= \Pr(\beta_x = j \wedge \beta_y = i \mid \alpha_x = i \wedge \alpha_y = j) + \\
&\quad \Pr(\gamma_x = \gamma_y \wedge \beta_x = K+1 \wedge \beta_y = K+1 \mid \alpha_x = i \wedge \alpha_y = j) \\
&= \Pr(\beta_x = j \mid \alpha_x = i) \times \Pr(\beta_y = i \mid \alpha_y = j) + \\
&\quad \Pr(\gamma_x = \gamma_y \mid \beta_x = K+1, \beta_y = K+1) \times \\
&\quad \Pr(\beta_x = K+1 \mid \alpha_x = i) \times \Pr(\beta_y = K+1 \mid \alpha_y = j) \\
&= \theta_{ij}\theta_{ji} + \frac{1}{N}\,\theta_{i,K+1}\theta_{j,K+1} \\
&= \hat{s}_{ij} + \frac{1}{N}\,\theta_{i,K+1}\theta_{j,K+1}
\end{aligned}$$

Case 2: Suppose $\alpha_x = \alpha_y = i \in \{1, \ldots, K\}$. Then

$$\begin{aligned}
&\Pr(h(x) = h(y) \mid \alpha_x = \alpha_y = i) \\
&= \Pr(\beta_x = \beta_y \leq K \mid \alpha_x = \alpha_y = i) + \\
&\quad \Pr(\gamma_x = \gamma_y \wedge \beta_x = K+1 \wedge \beta_y = K+1 \mid \alpha_x = \alpha_y = i) \\
&= \sum_{j=1}^{K} \Pr(\beta_x = \beta_y = j \mid \alpha_x = \alpha_y = i) + \\
&\quad \Pr(\gamma_x = \gamma_y \mid \beta_x = K+1, \beta_y = K+1) \times \\
&\quad \Pr(\beta_x = K+1 \mid \alpha_x = i) \times \Pr(\beta_y = K+1 \mid \alpha_y = j) \\
&= \sum_{j=1}^{K} \theta_{ij}^2 + \frac{1}{N}\,\theta_{i,K+1}^2 \\
&= \hat{s}_{ii} + \frac{1}{N}\,\theta_{i,K+1}^2
\end{aligned}$$

Summing up the above conditional probabilities, we get

$$\begin{aligned}
&\Pr(h(x) = h(y)) \\
&= \sum_{i \neq j} x_i y_j \Pr(h(x) = h(y) | \alpha_x = i \wedge \alpha_y = j) + \\
&\quad \sum_{i} x_i y_i \Pr(h(x) = h(y) | \alpha_x = \alpha_y = i) \\
&= \sum_{i,j} x_i \hat{s}_{ij} y_j + \frac{1}{N} \sum_{i \neq j} x_i y_j \theta_{i,K+1} \theta_{j,K+1} + \frac{1}{N} \sum_{i} x_i y_i \theta_{i,K+1}^2 \\
&= \lambda_S x^T S y + \frac{1}{N} \sum_{i,j} x_i y_j \theta_{i,K+1} \theta_{j,K+1}
\end{aligned}$$

To conclude the proof, observe that

$$0 \leq \frac{1}{N} \sum_{i,j} x_i y_j \theta_{i,K+1} \theta_{j,K+1} \leq \frac{1}{N} \left(\sum_{i} x_i \theta_{i,K+1} \right) \left(\sum_{j} x_j \theta_{j,K+1} \right) \leq \epsilon$$

□

For the special case where S is the identity matrix, $h(x; S)$ reduces to $h(x; I)$, which returns an $\alpha \in \{1, \dots, K\}$ sampled from $multi(x)$.

Lemma 5.3.3. *If S is a matrix of all ones, then S is hashable.*

Proof. Note that $x^T S y = 1$ in this case since $x, y \in \Delta^{K-1}$. Simply let $\mathcal{H}$ consist of one constant function. $\square$

Definition 5.3.4. A matrix $Q \in \mathbb{R}^{m \times m}$ is a *zero padded extension* of $S \in \mathbb{R}^{n \times n}$ if there exists an one-to-one function f that maps the indices $\tau = \{1 \dots n\}$ to $\{1 \dots m\}$ such that $Q_{i,j} = S_{f^{-1}(i), f^{-1}(j)}$ for any $i, j \in f(\tau)$ and $Q_{i,j} = 0$ otherwise.

In other words, Q is obtained by symmetrically inserting rows and columns of zeros into S.

Lemma 5.3.5. *If Q is a zero padded extension of S and S is hashable, then Q is hashable.*

Proof. Let $\epsilon > 0$, and let $x, y \in \Delta^{K-1}$. Define $x_{f(\tau)} \in \mathbb{R}^n$ such that its i^{th} element is $x_{f^{-1}(i)}$. We the define $g(x; Q, \epsilon)$ as follows:

1. Sample $\alpha \in \{1, \dots, m\} \sim multi(x)$
2. If $\alpha \in f(\tau)$, return $\left(0, h(\frac{x_{f(\tau)}}{||x_{f(\tau)}||_1}; S, \frac{\epsilon}{2})\right)$, where $h \in \mathcal{H}(S, \frac{\epsilon}{2})$ as in Definition 5.3.1
   Else return $\beta \in \{1, \dots, N\}$ uniformly drawn, where $N = \lceil 2/\epsilon \rceil$.

We now show Q is hashable.

$$\begin{aligned}
&\Pr\left(g(x; Q, \epsilon) = g(y; Q, \epsilon)\right) \\
&\quad = \Pr\left(\alpha_x \in f(\tau) \wedge \alpha_y \in f(\tau) \wedge h\left(\frac{x_{f(\tau)}}{||x_{f(\tau)}||_1}; S, \frac{\epsilon}{2}\right) = h\left(\frac{y_{f(\tau)}}{||y_{f(\tau)}||_1}; S, \frac{\epsilon}{2}\right)\right) \\
&\qquad + \Pr\left(\beta_x = \beta_y\right) \\
&\quad = ||x_{f(\tau)}||_1 \cdot ||y_{f(\tau)}||_1 \cdot \left(\lambda_S \frac{x_{f(\tau)}^T}{||x_{f(\tau)}||_1} S \frac{y_{f(\tau)}}{||y_{f(\tau)}||_1} + \delta\right) \\
&\qquad + \frac{1}{N}(1 - ||x_{f(\tau)}||_1)(1 - ||y_{f(\tau)}||_1) \\
&\quad = \lambda_S \cdot x^T Q y + ||x_{f(\tau)}||_1 \cdot ||y_{f(\tau)}||_1 \cdot \delta + \frac{1}{N}(1 - ||x_{f(\tau)}||_1)(1 - ||y_{f(\tau)}||_1)
\end{aligned}$$

where $0 \leq \delta \leq \epsilon/2$ by the choice of h. Note that

$$||x_{f(\tau)}||_1 \cdot ||y_{f(\tau)}||_1 \cdot \delta + \frac{1}{N}(1 - ||x_{f(\tau)}||_1)(1 - ||y_{f(\tau)}||_1) \leq \epsilon/2 + \epsilon/2 = \epsilon$$

□

Lemma 5.3.6. *If S is hashable, then aS is hashable for any $a > 0$.*

Proof. This follows directly from Definition 5.3.1 (by using $\lambda_{aS} = \frac{1}{a}\lambda_S$).

□

Lemma 5.3.7. *If $Q = \sum_{l=1}^{L} S_l$ and S_l is hashable for $l = 1, \ldots, L$, then Q is hashable.*

Proof. Suppose the hash function for S_l is h_l and the scalar is λ_{S_l}, for $l = 1, \ldots, L$.

Let $z = \sum_{l=1}^{L} \frac{1}{\sqrt{\lambda_{S_l}}}$ and $\theta \in \mathbb{R}^L$ where $\theta_l = \frac{1}{z} \cdot \frac{1}{\sqrt{\lambda_{S_l}}}$.

We construct hash function $g(x; Q, \epsilon)$ as follows:

1. Sample $\alpha \in \{1, \ldots, L\} \sim multi(\theta)$.
2. return $(\alpha, h_\alpha(x; S_l, \epsilon/L))$.

Then

$$\begin{aligned}
&\Pr\left(g(x; Q, \epsilon) = g(y; Q, \epsilon)\right) \\
&= \sum_{l=1}^{L} \Pr\left(\alpha_x = \alpha_y = l \wedge h_l(x; S_l, \epsilon/L) = h_l(y; S_l, \epsilon/L)\right) \\
&= \sum_{l=1}^{L} \theta_l^2 (\lambda_{S_l} x^T S_l y + \delta_l) \\
&= \frac{1}{z^2} x^T Q y + \sum_{l=1}^{L} \theta_l^2 \delta_l
\end{aligned}$$

where $0 \leq \delta_l \leq \epsilon/L$. Note that $0 \leq \sum_l^L \theta_l^2 \delta_l \leq \epsilon$ and thus Q is hashable.

□

Lemma 5.3.8. *Let $T = G(V, E)$ be a rooted tree and define $\pi_{m,n}$ to be the lowest common ancestor between node m and n for any $m, n \in V$. Let $V_r \subseteq V$ be subtree rooted at r (i.e.,, the set of all nodes descending from node $r \in V$ including r itself). Let $\Omega_r \subseteq V_r$ be all the leaf nodes of r and let $K_r = |\Omega_r|$. Let $f_r : \Omega_r \to \{1, \ldots, K_r\}$ be a one-to-one correspondence of the leaf nodes of r to a set of integers. Let $\xi(\cdot) : V \to \mathbb{R}$ be any function defined on V. Let $S^{(r,\xi)} \in \mathbb{R}^{K_r \times K_r}$ be a similarity matrix induced by r and ξ, where $S^{(r,\xi)}_{ij} = \xi(\pi_{f_r^{-1}(i), f_r^{-1}(j)}), \forall i = 1, \ldots, K_r, j = 1, \ldots, K_r$.*

For any $r \in V$, if $\xi(\cdot)$ is non-negative and downward non-decreasing in the subtree of r, that is, $\xi(q) \geq 0$ for any $q \in V_r$ and $\xi(q) \geq \xi(p)$ for any $p, q \in V_r$ such that q is a child of p, then $S^{(r,\xi)}$ is hashable.

Proof. Let $r \in V$. Suppose $\xi(\cdot)$ is non-negative and downward non-decreasing in the subtree of r. We prove the claim by induction on the tree.

If r is a leaf node, then $S^{(r,\xi)}$ is a scalar and thus hashable.

Now we consider the case when r is an internal node. Let $\sigma(r)$ be the set of direct children of r. Our inductive hypothesis is that given any $c \in \sigma(r)$, the similarity matrix $S^{(c,\xi')}$ induced by c and any $\xi' : V_c \to \mathbb{R}$, which is non-negative and downward non-decreasing, is hashable.

For a given $c \in \sigma(r)$, let $f_r(\Omega_c)$ be the set of indices of the leaf nodes of c in $S^{(r,\xi)}$. The tree structure implies

$$\bigcup_{c \in \sigma(r)} f_r(\Omega_c) = \{1, \ldots, K_r\} \tag{5.1}$$

and

$$f_r(\Omega_c) \bigcap f_r(\Omega_d) = \emptyset, \text{for any } c, d \in \sigma(r) \text{ and } c \neq d \,. \tag{5.2}$$

That is, the columns and rows of $S^{(r,\xi)}$ can be partitioned by the direct children of r.

Also, if c and d are different direct children of r, then the lowest common ancestor between the descendant nodes of c and those of d must be r. Thus

$$S^{(r,\xi)}_{f_r(\Omega_c),f_r(\Omega_d)} = \xi(\pi_{\Omega_c,\Omega_d}) = \xi(r)\cdot\mathbf{1}, \text{for any } c,d\in\sigma(r) \text{ and } c\neq d\ . \tag{5.3}$$

where $\mathbf{1}$ is a matrix of all ones.

For a given $c\in\sigma(r)$, define $Q^{(c)}\in\mathbb{R}^{K_r\times K_r}$ such that

$$Q^{(c)}_{ij} = \begin{cases} S^{(r,\xi)}_{ij} - \xi(r) & \text{if } i,j\in f_r(\Omega_c) \\ 0 & \text{otherwise.} \end{cases}$$

It follows from (5.1), (5.2) and (5.3) that

$$S^{(r,\xi)} = \xi(r)\cdot\mathbf{1} + \sum_{c\in\sigma(r)} Q^{(c)} \tag{5.4}$$

Define $\xi'(\cdot) = \xi(\cdot) - \xi(r)$. Since the lowest common ancestor of the leaf nodes of r cannot be higher than r and ξ is downward non-decreasing, we conclude that $\xi'(d)\geq 0$ for any $d\in V_r$ and $\xi'(d)$ is downward non-decreasing.

By the inductive hypothesis, given any $c\in\sigma(r)$, the similarity matrix $S^{(c,\xi')}$ induced by c and ξ' is hashable.

Now we show that $Q^{(c)}$ is a zero padded extension of $S^{(c,\xi')}$.

Let $K_c = |\Omega_c|$ and f_c be the function that maps the nodes in Ω_c to indices of $S^{(c,\xi')}$. Recall that f_r maps nodes in Ω_r (including Ω_c) to indices in $S^{(r,\xi)}$.

Let $f:\{1,\ldots,K_c\}\to\{1,\ldots,K_r\}$, where $f = f_r\cdot f_c^{-1}$. Let $\tau = \{1,\ldots,K_c\}$. It follows that $f(\tau) = f_r(\Omega_c)$.

For any $i, j \in f(\tau)$, that is, $\forall i, j \in f_r(\Omega_c)$,

$$\begin{aligned}
Q_{ij}^{(c)} &= S_{ij}^{(r,\xi)} - \xi(r) \\
&= \xi(\pi_{f_r^{-1}(i), f_r^{-1}(j)}) - \xi(r) \text{(By definition of } f_r\text{)} \\
&= \xi'(\pi_{f_r^{-1}(i), f_r^{-1}(j)}) \text{(By definition of } \xi'\text{)} \\
&= S^{(c,\xi')}_{f_c \cdot f_r^{-1}(i), f_c \cdot f_r^{-1}(j)} \text{(By definition of } f_c\text{)} \\
&= S^{(c,\xi')}_{f^{-1}(i), f^{-1}(j)}
\end{aligned}$$

By Definition 5.3.4, $Q^{(c)}$ is a zero padded extension of $S^{(c,\xi')}$ and is therefore hashable by Lemma 5.3.5. It follows from Lemma 5.3.3, Lemma 5.3.6, Lemma 5.3.7 and from (5.4) that $S^{(r,\xi)}$ is hashable. $\square$

Note that a similarity matrix derived from a hierarchy, as in Lemma 5.3.8, is not necessarily diagonally dominant. For example, if a leaf node has many siblings, the sum of its similarities with its siblings can easily be more than its self similarity.

Definition 5.3.9. A matrix $S \in \mathbb{R}^{K \times K}$ is *generally hashable*, if there exists a $\lambda_S > 0$ and, for any $\epsilon > 0$, a distribution on a family $\mathcal{H}(S, \epsilon)$ of hash functions $h(\cdot; S, \epsilon)$ such that for any $x, y \in \tilde{\Delta}^{K-1}$,

$$0 \leq \Pr\left(h_1(x; S, \epsilon) = h_2(y; S, \epsilon)\right) \; - \; \lambda_S \cdot x^T S y \leq \epsilon$$

where h_1 and h_2 are drawn independently from $\mathcal{H}(S, \epsilon)$.

Lemma 5.3.10. Hashing for the general case. *Any hashable matrix $S \in \mathbb{R}^{K \times K}$ is generally hashable.*

Proof. For any $x, y \in \tilde{\Delta}^{K-1}$, let $\hat{x} = (x/K, 1 - \sum_i x_i/K) \in \mathbb{R}^{K+1}$ and $\hat{y} = (y/K, 1 -$

$\sum_i y_i/K) \in \mathbb{R}^{K+1}$. Observe that $\hat{x}$ and $\hat{y} \in \Delta^K$. Let

$$\hat{S} \in \mathbb{R}^{(K+1)\times(K+1)}, \hat{S} = \begin{pmatrix} S & 0 \\ 0 & 0 \end{pmatrix}$$

$\hat{S}$ is a zero padded extension of S and is therefore hashable by Lemma 5.3.5. That is, there exists a $\lambda_{\hat{S}}$ and for any $\epsilon > 0$, a distribution on a family of functions $\hat{\mathcal{H}}$ such that

$$0 \leq \Pr_{\hat{h}_1,\hat{h}_2 \in \hat{\mathcal{H}}}(\hat{h}_1(\hat{x}) = \hat{h}_2(\hat{y})) - \hat{x}^T \hat{S} \hat{y} \leq \epsilon.$$

Observe that $\hat{x}^T \hat{S} \hat{y} = x^T S y$. Therefore

$$0 \leq \Pr_{\hat{h}_1,\hat{h}_2 \in \hat{\mathcal{H}}}(\hat{h}_1(\hat{x}) = \hat{h}_2(\hat{y})) - x^T S y \leq \epsilon.$$

Let $h(z) = \hat{h}(\hat{z})$, for any $z \in \tilde{\Delta}^{K-1}$. Observe that $\Pr(h_1(x) = h_2(y)) = \Pr(\hat{h}_1(\hat{x}) = \hat{h}_2(\hat{y}))$. Therefore,

$$0 \leq \Pr(h_1(x) = h_2(y)) - x^T S y \leq \epsilon.$$

By Definition 5.3.9, S is generally hashable. □

5.4 Experiments

5.4.1 Datasets and Evaluation Criteria

We use Caltech256 [5] and ILSVRC2010 [46], a subset of ImageNet with 1,000 classes and 1.2 million images (See Section 4.5). [5] We use Caltech256 to compare with existing similarity learning algorithms and use ILSVRC2010 for large scale experiments. Both datasets assign one class label per image.

[5]we do not use the newly collected validation and test sets as they are too small for retrieval evaluation.

For both datasets, we split the data into training and test, use the training set to learn semantic models and use the test to evaluate retrieval performance. For retrieval we obtain the top k neighbors by brute force scan except in Section 5.4.5. Unless otherwise noted, all evaluation is done by using each of the test images to query against the rest of the test images and reporting the average.

We use a ranking based criteria for evaluation. Given a similarity function, $Sim(a_i, a_j) \in [0, \infty)$, between images a_i and a_j, it can be used assign a rank, $r_i^q \in \{1, \ldots, n\}$ to n images $\{a_i\}_{i=\{1,\ldots,n\}}$ in a dataset with respect to a query image q so that $r_i^q <= r_j^q$ iff $Sim(q, a_i) >= Sim(q, a_j)$. Let $Sim_g(x, y)$ be "ground truth" or desired values of the similarity function. We can evaluate a ranking of k data items with respect to a query q by a precision,

$$p(Sim, q, Sim_g, k) = \frac{\sum_{r_i^q=1}^{k} Sim_g(q, a_i)}{\max_o \sum_{i=1}^{k} Sim_g(q, a_{o_i})} \tag{5.5}$$

The numerator is the sum of ground truth similarities for the most similar k database items based on similarity s for a query q. The denominator is the sum of ground truth similarities for the best possible k database items. The complexity of the evaluation function allows it to represent the standard "precision at k" for category labels when $Sim_g(a_i, a_j) \in \{0, 1\}$ is 1 for a_i and a_j with the same category label and 0 otherwise, as well as more general scores when $Sim_g(a_i, a_j) \in [0, 1]$ is a more nuanced measure of similarity, for instance based on hierarchical relationships between semantic categories.

We define the ground truth similarity in a similar way the hierarchical cost is defined in ILSVRC2010 [46]. Let $h(\pi(i, j))$ be the height of the lowest common ancestor $\pi(i, j)$ between class i and class j on the category hierarchy. The height of a node is the length of the longest path to one of its leaf node (leaf nodes have height 0). Similarity between class i and class j is then defined as $1 - h(\pi(i, j))/h^*$, where h^* is the height of the root node(19 for ILSVRC2010). All classes have similarity 1 to itself. We can then define the ground truth similarity between image a with ground truth class c_a and image b with c_b as

$Sim_g(a, b) = 1 - h(\pi(c_a, c_b))/h^*$. Precision at top k as in Eqn. 5.5 is then the average class similarity between the query and top k returned images divided by the maximum possible similarity from the dataset (perfect would be 1). We refer to this criteria as *hierarchical precision*.

5.4.2 Hierarchical Retrieval

We evaluate our similarity on hierarchical precision on ILSVRC2010 dataset. We split the ILSVRC2010 images 50%-50% as training and test. To learn the semantic attributes, we train binary linear SVMs using LIBLINEAR [86] on sparse 21K dimensional vectors formed by a three level SPM [15] on the published SIFT visual words from a 1,000 word code-book [46]. We use 2-fold cross validation to determine the parameter C. Probability calibration is done using Platt's scaling [121] during cross validation. In retrieval, we set the prior matrix S such that S_{ij}=1 $- h(\pi(i, j))/h^*$, matching the definition of hierarchical precision.

We compare our bilinear similarity with hierarchy encoded prior matrix (**B-Hie**) with various baselines: (1) **SPM**: ranking the images by intersection kernel on SPM histograms of visual words, representing low level feature based methods that do not use learning; (2) **Hard-Assign**: classifying the query image to the most likely class and ranking others by their probabilities of belonging to this class, equivalent to retrieval by annotation. (3) **Cosine-NoCal**: using cosine similarity of the raw outputs of semantic classifiers without probability calibration; (4)**Cosine-Flat**: using cosine similarity of the probabilities, same as Cosine-NoCal excpet with probability calibration; (5)**Cosine-Hie**: same as bilinear similarity with hierarchy encoded S except with L2 normalized probability vectors; (6)**B-Flat**: using bilinear similarity but without encoding the hierarchy in the prior matrix, i.e. with S set to identity.

We present the results in Figure 5.3 (left). Bilinear similarity with hierarchy encoded (B-Hie) achieves significantly better precision than all others. It also demonstrates that

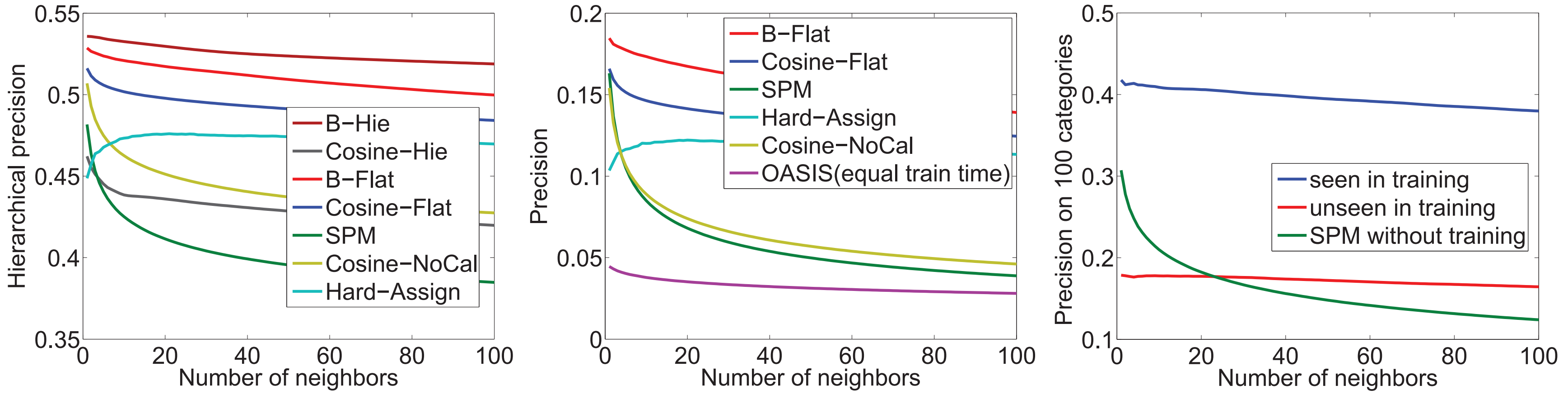


Figure 5.3: Precision vs. rank for similarity based retrieval on ILSVRC2010 images from 1,000 categories. Left: Hierarchical precision of our bilinear similarity with hierarchy encoded prior matrix (B-Hie) against others as described in Section 5.4.2. For all curves, stdev by swapping training and test is too small to show (≤ 0.002). Middle: Flat precision of our bilinear similarity with identity prior matrix (B-Flat) against others as described in Section 5.4.3. Stdev by swapping training and test is too small to show (≤ 0.001). Right: Flat precision on a subset 100 categories (Section 5.4.4). Using training data from the 100 categories (*"seen in training"*) performs the best, but training on 900 categories not including any of the 100 test categories (*"unseen in training"*) compares favorably to directly using SPM without any training.

all components of our similarity are essential: (1) learning semantic attributes is important as SPM (directly using low level features without learning) is quite effective for the first few nearest neighbors but for the rest of the curve performs significantly worse than those that use semantic classifiers; (2) probabilistic representation significantly improves over hard assignment (B-Flat versus Hard-Assign); (3) probability calibration is important as using raw classifier outputs (Cosine-NoCal) is significantly worse than the calibrated counterparts (Cosine-Flat & B-Flat) (4) cosine similarity performs significantly worse than bilinear similarity due to L2 normalization of probability vectors (Cosine-Flat versus B-Flat & Cosine-Hie versus B-Hie); (5) most importantly, encoding hierarchy into the prior matrix significantly improves precision, as quantitatively shown by B-Hie versus B-Flat and qualitatively by Figure 5.1, where images from nearby classes also rank higher.

5.4.3 Flat Retrieval

Our method is motivated by hierarchical retrieval. However, most existing work is optimized for a special case where the ground truth similarity between two images are 1 if they are from the same categories and 0 otherwise. The retrieval precision at top k as defined in Eqn. 5.5 is then percentage of the images from the same class of the query image. We refer to this criteria as *flat precision*.

To effectively compare with existing work we adapt our bilinear similarity to this special case by simply setting the prior matrix to identity in retrieval.

We experimented with our method on Caltech256 [5] in the same setting as in evaluating OASIS [105]. We use linear SVMs as the semantic classifiers, trained on the same features published by the authors of [105], parameter C determined by by 5-fold cross validation. We report results from 5 random splits of training and testing data (40 training images and 25 test images per class), as in [105]. Figure 5.4 shows our method is on par with OASIS and significantly outperforms all other algorithms. Moreover, our scheme is more efficient. For 50 classes, a single run of OASIS takes 96 seconds to converge [105],

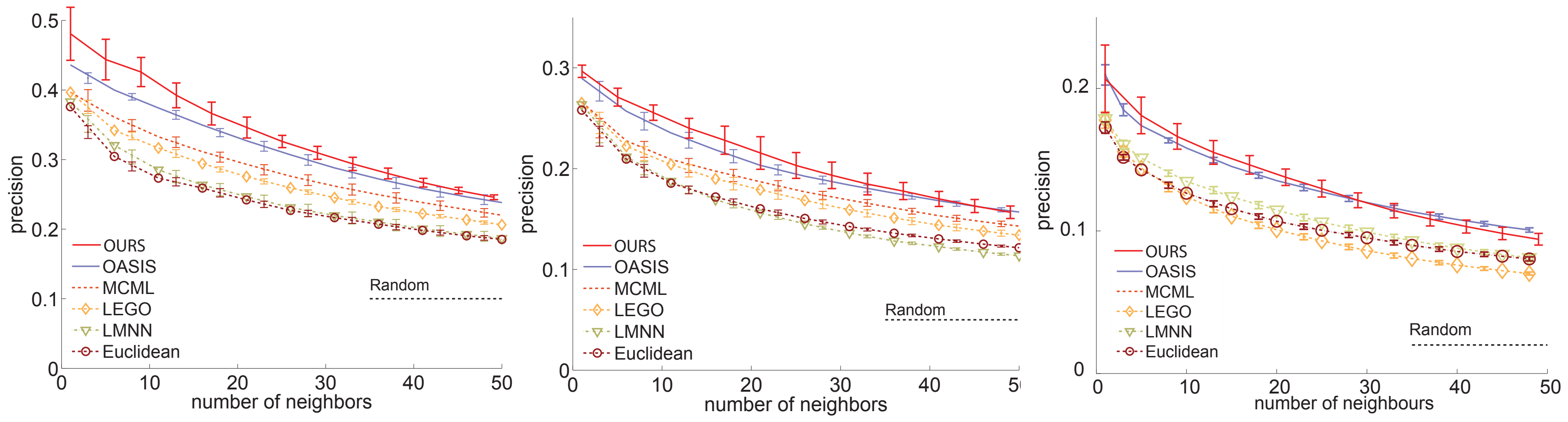


Figure 5.4: Comparison with (dis-)similarity learning methods including OASIS [105], MCML [107], LEGO [124] and LMNN [106] on 10 classes (left), 20 classes (middle) and 50 classes (right) from Caltech256. All curves except ours are from [105].

while learning all 50 linear SVMs and sigmoid functions for probability estimation (including 5 trials of parameter C and 5 fold cross validation for each trial) take 12 seconds in total, on a single CPU.

Next we compare our method with OASIS on the much larger ILSVRC2010 data and present results in Figure 5.3 (middle). We run multiple instances of OASIS with aggressiveness parameters $C = \{0.001, 0.01, \ldots, 1, 000\}$ and report the best after 14 days of training (our method takes 14 days in total on one CPU). We also include a subset of the baselines from Figure 5.3(left) excluding those optimized for hierarchy. OASIS seems slower to converge with the larger dataset and higher feature dimensionality (21K versus 1K for Caltech256) as it is still worse than SPM. Note that OASIS is inherently sequential while our method can be easily parallelized by training each semantic classifier independently.

5.4.4 Cross-Category Generalization

A potential advantage of using semantic attributes is the ability to generalize to new categories, as demonstrated in [112, 120] in a classification setting. We evaluate how our method can generalize to unseen categories in a retrieval setting. Only 900 of the 1,000 ILSVRC2010 semantic attributes are used to build the semantic representation and retrieval is *only* evaluated for categories for which no images are seen during training ("unseen in training" curve in Fig 5.3 right). While performance is lower than then when example images from those categories are seen in training ("seen in training" curve in Figure 5.3 right) it is still better than the raw feature comparison baseline for much of the retrieval list. Note that, as in all other experiments, training and test image sets are disjoint.

5.4.5 Indexing Efficiency

We evaluate the effectiveness of our bilinear LSH by measuring retrieval precision versus scanning cost. Scanning cost is the percentage of the data points needed to be scanned for top k retrieval, the dominating computation for retrieval. Brute force linear scan would re-

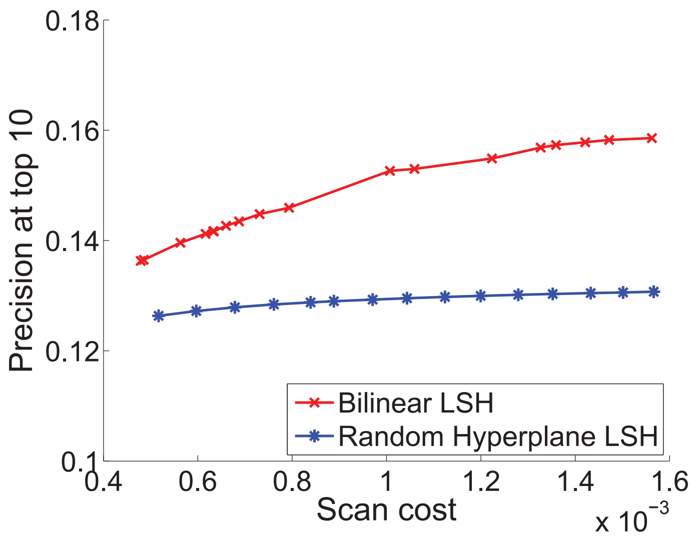


Figure 5.5: Precision at top 10 vs scanning cost for bilinear LSH and random hyperplane LSH as described in Section 5.4.5

sult 1.0 scan cost and maximum precision. One can adjust the number of hash tables and the number of hash function concatenations to trade off between precision and scanning cost. We compare our hashing technique with the widely used random hyperplane LSH [123] on retrieval. Random hyperplane LSH approximates the cosine similarity $\frac{x^T y}{\|x\|_2\|y\|_2}$ by repeatedly selecting a random hyperplane and project vector x to one bit depending on which side x is on. To compare fairly with random hyperplane LSH, we set our prior similarity matrix S to identity and use flat precision to evaluate. We set the length of hash code for both methods to be the same (20 bits). In Figure 5.5, we vary the number of hash tables to produce the precision versus scanning cost curves. Figure 5.5 shows that bilinear LSH achieves a precision of 0.15 for top 10 images, very close to the precision (0.17) of linear scan, while examining only 0.1% of the database points. This is significantly better than random hyperplane LSH.

5.5 Summary

We have presented an approach that can exploit prior knowledge of a semantic hierarchy for similar image retrieval, and is scalable to very large retrieval problems. Experiments show that adding hierarchical knowledge significantly increases retrieval performance. In addition we show that a handicapped version of our system – without prior hierarchical information – can start with the same training information as the state of the art for similarity function learning (OASIS) and learn a more accurate similarity function with less total computation, and much less "wall clock time" due to significantly better inherent parallelism. We note that our technique should be seen as a complement to previous work on similarity learning as it is most useful when some explicit labels are available at training time (a common scenario). Our final contribution is a hashing scheme for bilinear similarity on probability distributions that is shown to provide efficient (sublinear) retrieval in our setting, and may be useful in a wide range of applications. Using our technique, one query against a database of 600K images takes only 3 milliseconds with an accuracy close to 90% of linear scan and the computational cost 0.001 times that of linear scan. This completes an end-to-end system for very large scale hierarchical retrieval that has inherent parallelization, linear time training, sublinear time retrieval, and better accuracy and scalability than the state of the art.

Chapter 6

Multi-level Classification with Accuracy Guarantees

6.1 Introduction

This chapter further exploits the semantic relations between categories. We start by revisiting the ultimate goal of recognition: can computers recognize all object classes while almost never making mistakes? This is a challenging task even to a knowledgeable human! [1]

So far this has turned out to be elusive—the recent state of the art performance on 10K-way classification is only 16.7% [125], albeit significantly improved since our benchmarking study as described in Chapter 3. Of course there *is* a way to *always be right*, as we can just report everything as an "entity," which is, however, not very informative. This chapter shows how to achieve something sensible between the two extremes of inaccurate choices forced among a large number of categories and the uninformative option of declaring that everything is an "entity."

One key to success is to observe that object categories form a semantic hierarchy, con-

[1]The materials of this chapter are also presented in [56].

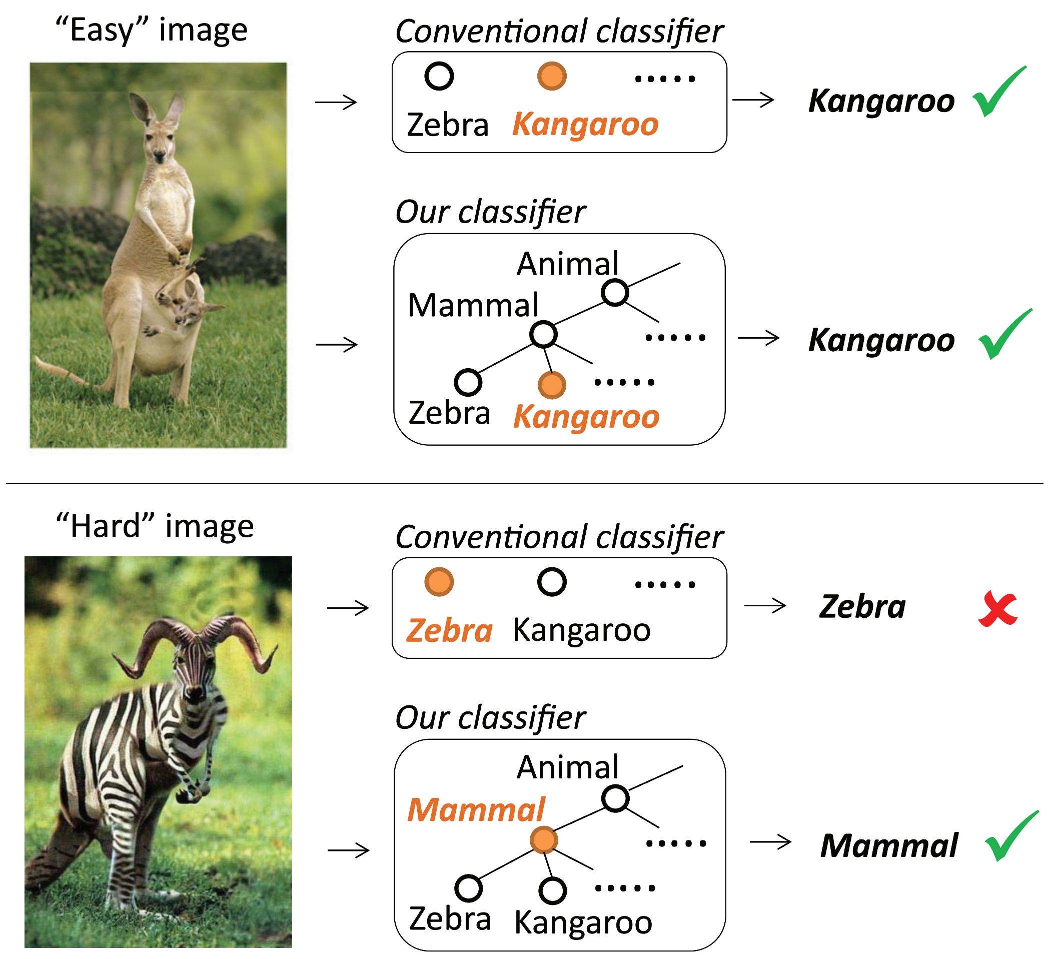


Figure 6.1: Conventional classifier versus our approach.

sisting of many levels of abstraction. For example, a kangaroo is also a mammal, an animal, and a living thing. The classifier should predict "mammal" instead if it is uncertain of the specific species. Meanwhile, the classifier should try to be as specific as possible. Consider the bottom image in Figure 6.1, where it would have been correct to report animal, but choosing mammal provides more information without being wrong. A sensible classifier thus "hedges its bets" as necessary, maintaining high accuracy while making its best effort for specificity.

Our goal is *to create a classification system that maximizes information gain while maintaining a fixed, arbitrarily small error rate.* We measure information gain in the standard information theoretical sense, i.e., the decrease in uncertainty from our prior distribution to the posterior over the classes. For example, our prior can be uniform among the tens of thousands of leaf nodes in a hierarchy. A classification output of "mammal," though maintaining uncertainty about the specific species, provides information by ruling out many other possibilities. Note that our algorithmic approach can also handle alternate, application-specific measures instead of information gain.

Results on datasets ranging from 65 to 10K classes show that not only is our proposed algorithm effective at training classifiers that optimize information gain while maintaining high accuracy, but that the resulting classifications are informative. This is a step toward making automatic object classification more widely useful by making explicit the *trade-off between accuracy and specificity*. This trade-off can be relevant in many visual tasks with high level semantic labels, e.g., detection, scene understanding, segmentation, describing images with sentences, etc. Our focus here, multiclass image classification, serves as a building block. To our knowledge this is the first time that *optimizing* the accuracy-specificity trade-off has been addressed in large scale visual recognition.

In this chapter, we make the following contributions: (1) introducing the problem of classification in a hierarchy subject to an accuracy bound while maximizing information gain (or other measures), (2) proposing the Dual Accuracy Reward Trade-off Search (DARTS) algorithm and proving a non-trivial result that, under practical conditions, it converges to an optimal trade-off, and (3) validating our algorithm with experiments on 65 to 10K classes, showing large improvements over baseline approaches.

6.2 Related Work

Our problem is related to cost-sensitive classification and hierarchical classification [126, 127, 52, 53, 128, 26, 27, 29, 24, 31, 23, 28, 25, 22, 129, 130, 131]. The key differences are: (1) conventional multiclass or cost-sensitive techniques do not consider overlapping classes in a hierarchy; (2) previous work on hierarchical classification has not addressed

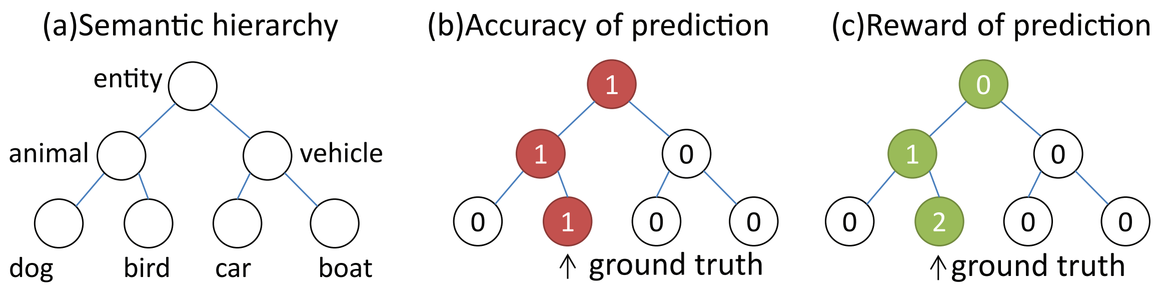


Figure 6.2: Illustration of the formulation with a simple hierarchy. The numbers correspond to the accuracy (middle) and the reward (information gain) of a prediction (right) given the ground truth.

the issue of automatically selecting the appropriate level of abstraction to *optimize* the accuracy-specificity trade-off.

Also related is classification with reject options, which grants binary classifiers an option to abstain, for a particular cost [132, 133]. In the multiclass case [134, 135, 136], also termed "class selective rejection," the classifier can output an *arbitrary* set of classes. Our problem fits in this framework, with the admissible sets restricted to internal nodes of the hierarchy. To our knowledge we are the first to connect class selective rejection with hierarchical visual classification. Our primal dual framework follows [135], but our results on the optimality of DARTS are new (Section 6.4.2).

Our work is also inspired by an emerging trend in computer vision studying large scale visual recognition [98, 36, 83, 30, 137, 52, 125, 20, 138], as well as fine-grained categorization [104, 139, 140, 141, 142]. Our technique scales up easily and we demonstrate its effectiveness on large scale datasets.

6.3 Formulation

We describe the visual world with a semantic hierarchy $H = (V, E)$, a directed acyclic graph (DAG) with a unique root $\hat{v} \in V$, each node $v \in V$ representing a semantic class (Figure 6.2a). The leaf nodes $\mathcal{Y} \subset V$ are *mutually exclusive* classes. The internal nodes are unions of leaf nodes determined by the hierarchy, e.g., in Figure 6.2a, "animal" is a combination of “dog” and “bird,” while “entity” is a combination of everything under “animal” and “vehicle.”

Given the hierarchy, it is then correct to label an image at either its ground truth leaf node or any of its ancestors (Figure 6.2b), e.g., a dog is also an animal and an entity. Let X be an image represented in some feature space and Y its ground truth leaf label, X and Y drawn from a joint distribution on $\mathcal{X} \times \mathcal{Y}$. A classifier $f : \mathcal{X} \to V$ labels an image $x \in \mathcal{X}$ as a node $v \in V$, either a leaf or an internal node. The accuracy $\Phi(f)$ of the classifier f is then

$$\Phi(f) = \mathbb{E}\left[(f(X) \in \pi(Y)\right],\ ^{2} \tag{6.1}$$

where $\pi(Y)$ is the set of all possible correct predictions, i.e., the ground truth leaf node and its ancestors. Note that without the internal nodes, $\Phi(f)$ reduces to the conventional flat multiclass accuracy. In this work, we use “accuracy” in the hierarchical sense unless stated otherwise.

The conventional goal of classification is maximizing accuracy. In our case, however, always predicting the root node ensures 100% accuracy, yielding an uninformative solution. We clearly prefer an answer of “dog” over “entity,” whenever they are both correct. We encode this preference as a reward $r_v \geq 0$ for each node $v \in V$. One natural reward is information gain, the decrease in uncertainty (entropy) from the prior distribution to the posterior over the leaf classes. Assuming a uniform prior, it is easy to verify that a prediction at node v decreases the entropy by

$$r_v = \log_2 |\mathcal{Y}| - \log_2 \sum_{y \in \mathcal{Y}} [v \in \pi(y)]. \tag{6.2}$$

The information gain is zero at the root node and maximized at a leaf node. Note that we use information gain in experiments but our algorithm and analysis can accommodate an *arbitrary* non-negative reward. Given the reward of each node, the reward $R(f)$ for a

[2] “$[P]$” is the Iverson bracket, i.e., 1 if P is true and 0 otherwise.

classifier f is

$$R(f) = \mathbb{E}\left(r_{f(X)}[f(X) \in \pi(Y)]\right), \tag{6.3}$$

i.e., r_v for a correct prediction at node v, and 0 for a wrong one(Figure 6.2c). In the case of information gain, the reward of a classifier is the average amount of correct information it gives. Our goal then is to maximize the reward given an accuracy guarantee $0 < 1-\epsilon \leq 1$, i.e., the following optimization problem (OP1).

$$\begin{aligned} \underset{f}{\text{maximize}} \quad & R(f) \\ \text{subject to} \quad & \Phi(f) \geq 1-\epsilon. \end{aligned} \tag{OP1}$$

Note that OP1 is always feasible because there exists a trivial solution that only predicts the root node.

6.4 The DARTS Algorithm

In this section we present the Dual Accuracy Reward Trade-off Search (DARTS) algorithm to solve OP1 and prove its optimality under practical conditions.

DARTS is a primal dual algorithm based on "the generalized Lagrange multiplier method" [143]. In our case, the dual variable controls the trade-off between reward and accuracy. We first write the Lagrange function

$$L(f, \lambda) = R(f) + \lambda(\Phi(f) - 1 + \epsilon), \tag{6.4}$$

with the dual variable $\lambda \geq 0$. Given a λ, we obtain a classifier f_λ that maximizes the Lagrange function, a weighted sum of reward and accuracy controlled by λ. It can be shown that the accuracy of the classifier $\Phi(f_\lambda)$ is non-decreasing and the reward $R(f_\lambda)$ non-increasing with respect to λ [143]. Moreover, if a $\lambda^\dagger \geq 0$ exists such that $\Phi(f_{\lambda^\dagger}) = 1-\epsilon$, i.e., the classifier $f_{\lambda^\dagger}$ has an accuracy of exactly $1-\epsilon$, then $f_{\lambda^\dagger}$ is optimal for OP1 [143].

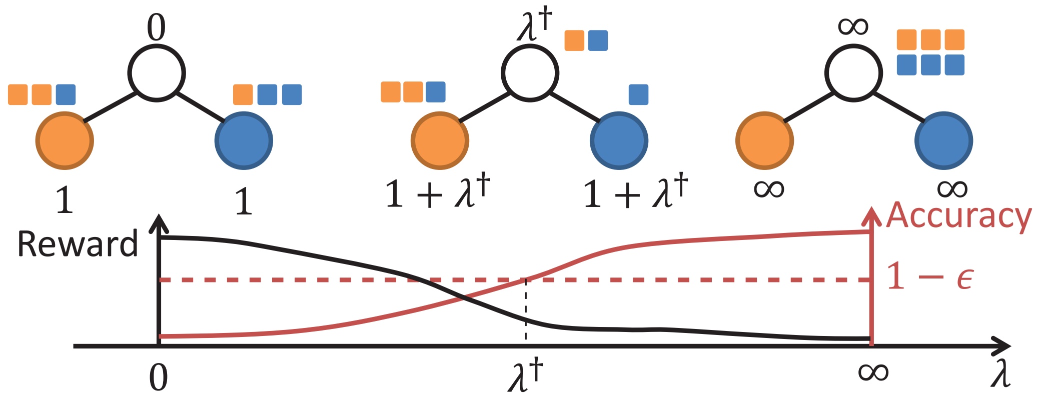


Figure 6.3: Bottom: The general properties of the reward and accuracy of f_λ, a classifier that maximizes the Lagrange function, with respect to the dual variable λ. An optimal solution to OP1 is $f_{\lambda^\dagger}$, where the accuracy is exactly the minimum guaranteed, provided $\lambda^\dagger$ exists. Top: The solid squares represent image examples; their color indicates the ground truth. The numbers next to the nodes are the transformed rewards $r_v + \lambda$ in the Lagrange function. As λ increases, the classifier f_λ predicts more examples to the root node. Eventually every example goes to the root node unless some other node already has posterior probability 1.

These properties, illustrated in Figure 6.3 (bottom), lead to a binary search algorithm to find such a $\lambda^\dagger$. At each step the algorithm seeks a classifier that maximizes the Lagrange function. It converges to an optimal solution provided such a $\lambda^\dagger$ exists.

To apply this framework, however, we must address two challenges:(1) finding the classifier that maximizes the Lagrange function and (2) establishing conditions under which $\lambda^\dagger$ exists and thus the binary search converges to an optimal solution. The latter is particularly non-trivial as counterexamples exist, e.g., the red curve in Figure 6.3 can be discontinuous and as a result the dashed line can fail to meet it. We will give a concrete example in Section 6.4.2.1.

6.4.1 Maximizing the Lagrange Function

DARTS maximizes the Lagrange function by using posterior probabilities. Using Eqn. 6.3, Eqn. 6.1, and Eqn. 6.4 yields

$$L(f,\lambda) = \mathbb{E}\ (r_{f(X)} + \lambda)[f(X) \in \pi(Y)] + \lambda(\epsilon - 1), \tag{6.5}$$

i.e., maximizing the Lagrange function is simply maximizing a transformed reward $r_v + \lambda, \forall v \in V$. This can be achieved by estimating posterior probabilities and predicting the node with the maximum expected reward, breaking ties arbitrarily. Let f_λ be such a classifier given a λ, then

$$f_\lambda(x) = \operatorname*{argmax}_{v \in V} (r_v + \lambda) p_{Y|X}(v|x), \tag{6.6}$$

where $p_{Y|X}(v|x) = \Pr(v \in \pi(Y)|X = x)$. This can be easily proven by rewriting Eqn. 6.5 using iterated expectations, conditioning on X first.

Let's examine f_λ. When $\lambda = 0$, f_0 simply maximizes the original reward. As λ tends to infinity, the transformed reward $r_v + \infty$ becomes equal on all nodes. The root node has maximum probability and therefore the best expected reward. Thus every example is predicted to the root node, unless some other node already has probability 1. Either way, all predictions are accurate with $\lambda = \infty$.

To obtain the posterior probabilities, we learn conventional one-vs-all classifiers on the leaf nodes (e.g., SVMs), obtain probability estimates (e.g., via Platt scaling [121]), and sum them to get internal node probabilities.

We summarize DARTS in Algorithm 3. It first obtains posterior probabilities for all nodes and exits if f_0, the classifier that maximizes the original reward only, is already at least $1 - \epsilon$ accurate (step 1–4). Otherwise it does a binary search to find a $\lambda^\dagger > 0$ such that the classifier that maximizes the transformed ward $r_v + \lambda^\dagger$ is exactly $1 - \epsilon$ accurate. The upper bound of the binary search interval, i.e., $\bar{\lambda}$, is set such that $\lambda^\dagger \leq \bar{\lambda}$ is guaranteed (proof in Section 6.4.1.1). DARTS runs for no more than T iterations or until $\Phi(f_\lambda)$ is within a small number $\tilde{\epsilon}$ from $1 - \epsilon$.

To obtain the classifier f_λ given a new λ (step 6), it suffices to have the posterior probabilities on the leaf nodes. Thus we only need to learn 1-vs-all classifiers on the leaf nodes *once*, i.e., DARTS essentially converts a "base" flat classifier with probability estimates to a hierarchical one with the optimal accuracy-specificity trade-off.

Finally we remark that DARTS is not sensitive to non-exact maximization of the La-

Algorithm 3 DARTS

```
1. Obtain $p_{Y|X}(y|x), y \in \mathcal{Y}$.
2. $p_{Y|X}(v|x) \leftarrow \sum_{y \in \mathcal{Y}} [v \in \pi(y)] p_{Y|X}(y|x), \forall v \in V$.
3. $f_0 \leftarrow \arg\max_{v \in V} r_v p_{Y|X}(v|x)$.
4. If $\Phi(f_0) \geq 1 - \epsilon$, return $f_0$.
5. $\bar{\lambda} \leftarrow (r_{max}(1-\epsilon) - r_{\hat{v}})/\epsilon$, where $r_{max} = \max_{v \in V} r_v$.
6. Do binary search for a $\lambda \in (0, \bar{\lambda}]$ until $0 \leq \Phi(f_\lambda) - 1 + \epsilon \leq \tilde{\epsilon}$ for a maximum of $T$
   iterations. Return $f_\lambda$.
```

grange function, e.g., inaccurate probability estimates, as the error will not be amplified [143]: if a solution f_λ is within $\delta > 0$ from the maximizing the Lagrange function, then with the accuracy guarantee set to that of f_λ, f_λ is within δ from maximizing the reward.

6.4.1.1 Proof about $\bar{\lambda}$ in DARTS

In this section we show that $\bar{\lambda}$ as defined in Line 5 of DARTS can be an upper-bound of the binary search interval in DARTS. To that end, we first prove a lemma stating that given a $1 - \epsilon$, if the transformed root reward $r_{\hat{v}} + \lambda$ is large enough, then the classifier f_λ that maximizes the Lagrange function $L(f, \lambda)$ is at least $1 - \epsilon$ accurate.

Let $r_{max} = \max_{v \in V} r_v$ and we assume that $r_{max} > 0$ because otherwise all rewards are zero and OP1 is not meaningful.

Lemma 6.4.1. *For any $\lambda \geq 0$, if $(r_{\hat{v}} + \lambda)/(r_{max} + \lambda) \geq 1 - \epsilon$, where $\hat{v}$ is the root node, then $\Phi(f_\lambda) \geq 1 - \epsilon$.*

Proof. Assume to the contrary that $\Phi(f_\lambda) < 1 - \epsilon$. Then

$$\begin{aligned}
L(f_\lambda, \lambda) =& \mathbb{E}\,(r_{f_\lambda(X)} + \lambda)[f_\lambda(X) \in \pi(Y)] + \lambda(\epsilon - 1) \\
\leq& \mathbb{E}\,(r_{max} + \lambda)[f_\lambda(X) \in \pi(Y)] + \lambda(\epsilon - 1) \\
=& (r_{max} + \lambda)\Phi(f_\lambda) + \lambda(\epsilon - 1) \\
<& (r_{max} + \lambda)(1 - \epsilon) + \lambda(\epsilon - 1) \\
\leq& r_{\hat{v}} + \lambda + \lambda(\epsilon - 1) = L(\hat{f}, \lambda),
\end{aligned}$$

where $\hat{f}$ is the trivial solution that maps all examples to the root node. This contradicts that f_λ maximizes $L(f, \lambda)$. □

Now we are ready to prove the claim about $\bar{\lambda}$.

Lemma 6.4.2. *Let* $\bar{\lambda} = (r_{max}(1 - \epsilon) - r_{\hat{v}})/\epsilon$*, where* $\hat{v}$ *is the root node and* $r_{max} = \max_{v \in V} r_v$*. If* $\Phi(f_0) < 1 - \epsilon$*, then* $\bar{\lambda} > 0$ *and* $\Phi(f_{\bar{\lambda}}) \geq 1 - \epsilon$*.*

Proof. Note that $\bar{\lambda} > 0$ because otherwise Lemma 6.4.1 implies that $\Phi(f_0) \geq 1 - \epsilon$. It is easy to verify that $(r_{\hat{v}} + \bar{\lambda})/(r_{max} + \bar{\lambda}) \geq 1 - \epsilon$. Lemma 6.4.1 then implies that $\Phi(f_{\bar{\lambda}}) \geq 1 - \epsilon$. □

6.4.2 Optimality of DARTS

Now we prove that under practical conditions, roughly when the posterior probabilities are continuously distributed, DARTS converges to an optimal solution.

The key is to investigate when the dual variable $\lambda^\dagger$ exists, i.e., when the monotonic red curve in Figure 6.3 can meet the dashed line. This is only of concern when $\Phi(f_0) < 1 - \epsilon$, i.e., the start of the red curve is below the dashed line, because otherwise we have satisfied the accuracy guarantee already. With $\Phi(f_0) < 1 - \epsilon$, $\lambda^\dagger$ may not exist in two cases: (1) when the end of the curve is below the dashed line, i.e., $\Phi(f_\infty) < 1 - \epsilon$, or (2) when the curve is discontinuous. Our main theoretical results state that under normal conditions, these two cases cannot happen and then $\lambda_\dagger$ must exist.

Case(1) cannot happen because we can show that $\bar{\lambda} > 0$ and $\Phi(f_{\bar{\lambda}}) \geq 1 - \epsilon$, where $\bar{\lambda}$ is defined in line 5 of DARTS (proof in Section 6.4.1.1).

Case(2) is harder as the curve can indeed be discontinuous (see Section 6.4.2.1 for a concrete case). However, we show that case(2) cannot occur if the posterior probabilities are continuously distributed except possibly at 0 or 1, a condition normally satisfied in practice. For example, consider a hierarchy of two leaf nodes a and b. The posterior probability $p_{Y|X}(a|X)$, as a function of X, is also a random variable. The condition implies that the distribution of $p_{Y|X}(a|X)$ does not concentrate on any single real number other than 0 and 1, i.e., practically, the posterior probability estimates are sufficiently diverse.

Formally, let $\Delta = \{q \in \mathbb{R}^{|\mathcal{Y}|-1} : q \succeq 0, \|q\|_1 \leq 1\}$ be the set of possible posterior probabilities over the $|\mathcal{Y}| - 1$ leaf nodes. Note that for $|\mathcal{Y}|$ leaf nodes there are only $|\mathcal{Y}| - 1$ degrees of freedom. Let $\Delta^\ddagger = \{q \in \Delta : \|q\|_\infty = 1 \vee q = 0\}$ be the set of posterior probabilities at the vertices of Δ, where one of the leaf nodes takes probability 1. Let $\vec{p}_{Y|X} : \mathcal{X} \to \Delta$ be a Borel measurable function that maps an example x to its posterior probabilities on leaf nodes. Let $Q = \vec{p}_{Y|X}(X)$ be the posterior probabilities on leaf nodes for the random variable X. As a function of X, Q is also a random variable. Our main result is the following theorem.

Theorem 6.4.3. *If $\Pr(Q \in \Delta^\ddagger) = 1$, or Q has a probability density function with respect to the Lebesgue measure on $\mathbb{R}^{|\mathcal{Y}|-1}$ conditioned on $Q \notin \Delta^\ddagger$, then strong duality holds for OP1 and DARTS converges to an optimal solution of OP1.*

Proof. We only need to show the continuity of $\Phi(f_\lambda)$ with respect to $\lambda \geq 0$.

We first have

$$\begin{aligned}\Phi(f_\lambda) =& p^\ddagger \mathbb{E}_{X,Y|Q\in\Delta^\ddagger}[f_\lambda(X) \in \pi(Y)] \\ &+ (1 - p^\ddagger)\mathbb{E}_{X,Y|Q\notin\Delta^\ddagger}[f_\lambda(X) \in \pi(Y)],\end{aligned} \tag{6.7}$$

where $p^\ddagger = \Pr(Q \in \Delta^\ddagger)$.

Consider the first expectation in Eqn. 6.7. Let $y^\ddagger$ be the leaf node such that $Q_{y^\ddagger} = 1$, i.e., $Y = y^\ddagger$ with probability 1. Then $p_{_{Y|X}}(v|X) = 1$ for any $v \in \pi(y^\ddagger)$ and $p_{_{Y|X}}(v|X) = 0$ otherwise. Therefore $f_\lambda(X) = \operatorname{argmax}_{v\in V} r_v p_{_{Y|X}}(v|X) = \operatorname{argmax}_{v\in\pi(y^\ddagger)} r_v$. Thus $f_\lambda(X) \in \pi(Y)$ with probability 1, i.e., the first expectation is the constant 1.

Therefore we only need to show the continuity of the second expectation with respect to λ and we do this in the rest of the proof. For simplicity of notation, we drop $Q \notin \Delta^\ddagger$ hereafter and simply write the second expectation in Eqn. 6.7 as $\mathbb{E}_{X,Y}[f_\lambda(X) \in \pi(Y)]$.

Define $\tilde{f}_\lambda : \Delta \to V$ as

$$\tilde{f}_\lambda(q) = \operatorname{argmax}_{v\in V}(r_v + \lambda)q_v,$$

breaking ties the same way as in Equation 6.6. Then it follows that $\forall x \in \mathcal{X}$, $f_\lambda(x) = \tilde{f}_\lambda(\vec{p}_{_{Y|X}}(x))$. We also define

$$\Gamma_v(\lambda) = \{q \in \Delta : (r_v + \lambda)q_v > (r_{v'} + \lambda)q_{v'}, \forall v' \neq v\}.$$

to be the open polyhedron in Δ such that $\tilde{f}_\lambda(q) = v, \forall q \in \Gamma_v(\lambda)$, i.e., the set of posterior probabilities that lead to a prediction at node v given λ. Also let

$$\begin{aligned}\bar{\Gamma}(\lambda) =& \{q \in \Delta : \exists v', v'', v' \neq v'', \forall u \neq v', u \neq v'',\\ &(r_{v'} + \lambda)q_{v'} = (r_{v''} + \lambda)q_{v''} \geq (r_u + \lambda)q_u\},\end{aligned}$$

i.e., the set of probabilities that lie on a decision boundary. It is then a simple exercise to check that $\bar{\Gamma}(\lambda)$ and $\Gamma_v(\lambda), \forall v \in V$ partition Δ.

Let $p_Q(q)$ be the (conditional) density function of Q given $Q \notin \Delta^{\ddagger}$, then

$$\begin{aligned}
&\mathbb{E}_{X,Y}[f_\lambda(X) \in \pi(Y)] \\
&= \mathbb{E}_X \mathbb{E}_{Y|X}[f_\lambda(X) \in \pi(Y)] \\
&= \mathbb{E}_Q \mathbb{E}_{X|Q} \mathbb{E}_{Y|X,Q}[f_\lambda(X) \in \pi(Y)] \\
&= \mathbb{E}_Q \mathbb{E}_{X|Q} \sum_{y \in \mathcal{Y}} [f_\lambda(X) \in \pi(y)] p_{Y|X}(y|X) \\
&= \mathbb{E}_Q \mathbb{E}_{X|Q} \sum_{y \in \mathcal{Y}} [\tilde{f}_\lambda(Q) \in \pi(y)] Q_y \\
&= \mathbb{E}_Q \sum_{y \in \mathcal{Y}} [\tilde{f}_\lambda(Q) \in \pi(y)] Q_y \\
&= \int_\Delta \sum_{y \in \mathcal{Y}} [\tilde{f}_\lambda(q) \in \pi(y)] q_y \, p_Q(q) \, \mathrm{d}q \\
&= \left(\int_{\bar{\Gamma}(\lambda)} + \sum_{v \in V} \int_{\Gamma_v(\lambda)} \right) \sum_{y \in \mathcal{Y}} [\tilde{f}_\lambda(q) \in \pi(y)] q_y \, p_Q(q) \, \mathrm{d}q \\
&= \sum_{v \in V} \int_{\Gamma_v(\lambda)} q_v p_Q(q) \, \mathrm{d}q \\
&= \sum_{v \in V} \int [q \in \Gamma_v(\lambda)] q_v p_Q(q) \, \mathrm{d}q.
\end{aligned}$$

Note that the first two equalities are by iterated expectations. Also we can drop $\int_{\bar{\Gamma}(\lambda)}$ at the second to last step because $\bar{\Gamma}(\lambda)$ has dimensions less than $|\mathcal{Y}| - 1$ and therefore has zero measure.

Let $\phi_v(\lambda, q) = [q \in \Gamma_v(\lambda)] q_v p_Q(q)$. To prove continuity, it suffices to show that for each v, $\int \phi_v(\lambda, q) \, \mathrm{d}q$ is continuous with respect to λ, i.e., for sequences $\{\lambda_n\}$, if $\lim_{n\to\infty} \lambda_n = \lambda$, then $\lim_{n\to\infty} \int \phi_v(\lambda_n, q) \, \mathrm{d}q = \int \phi_v(\lambda, q) \, \mathrm{d}q$. This is directly implied by Lebesgue's dominated convergence theorem if we can show (1) $\lim_{n\to\infty} \phi_v(\lambda_n, q) = \phi_v(\lambda, q)$ almost everywhere and (2) for all n and every q, $|\phi_v(\lambda_n, q)| \le \psi(q)$ for some integrable ψ.

Note that condition(2) is trivial as $\phi_v(\lambda_n, q) \le p_Q(q)$ and thus we only need to check condition(1). First note that condition(1) trivially holds for any $q \notin \Delta$. For $q \in \Delta$, there are three possibilities: (i) $q \in \Gamma_v(\lambda)$, (ii) $q \in \Gamma_u(\lambda)$ for some $u \neq v$, or (iii) $q \in \bar{\Gamma}(\lambda)$. We only need to check (i) and (ii) because $\bar{\Gamma}(\lambda)$ has zero measure.

For (i), let $\gamma_v(\lambda, q) = (r_v + \lambda)q_v - \max_{v' \neq v}(r_{v'} + \lambda)q_{v'}$. For any $q \in \Gamma_v(\lambda)$, as $n \to \infty$, $\gamma_v(\lambda_n, q) \to \gamma_v(\lambda, q) > 0$. Therefore there exists n' such that $\forall n > n', \gamma_v(\lambda_n, q) > 0$ and thus $\forall n > n', [q \in \Gamma_v(\lambda_n)] = 1$. Therefore $[q \in \Gamma_v(\lambda_n)] \to 1 = [q \in \Gamma_v(\lambda)]$.

For (ii), since $q \in \Gamma_u(\lambda)$ for some $u \neq v$, as $n \to \infty$, $\gamma_v(\lambda_n, q) \to \gamma_v(\lambda, q) < 0$ and therefore $[q \in \Gamma_v(\lambda_n)] \to 0 = [q \in \Gamma_v(\lambda)]$.

□

Note that our condition differs from the one given in [135] for strong duality in a general class selective rejection framework, i.e., a continuous density function $p_{X|Y}(x|y) = \Pr(X = x|Y = y)$ exists for each $y \in \mathcal{Y}$. First, neither condition implies the other. Second, theirs guarantees strong duality but not the optimality of a dual algorithm using only posterior probabilities to maximize the Lagrange function, as the maximizer may not be unique. We elaborate using a concrete example in Section 6.4.2.2.

In practice, one can estimate whether the condition holds by checking the dual variable λ and the classifier f_λ DARTS returns. If $\lambda = 0$ or the accuracy of f_λ is close to $1 - \epsilon$, then the solution is near optimal. Even if the condition fails to hold, i.e., $\lambda > 0$ and the accuracy of f_λ is $1 - \epsilon' \neq 1 - \epsilon$, then although f_λ is sub-optimal for the $1 - \epsilon$ accuracy guarantee, it is nonetheless optimal for a guarantee of $1 - \epsilon'$.

In the next two sections (Section 6.4.2.1 and Section 6.4.2.2), we give an example of discontinuous $\Phi(f_\lambda)$, further shedding light on the condition given in Theorem 6.4.3, and discuss in full detail the difference between our condition and the condition established in [135]. These two sections are theoretically important but can be skipped without affecting the understanding of the rest of the chapter.

6.4.2.1 An example of discontinuous $\Phi(f_\lambda)$

We assume the simplest hierarchy: two leaf nodes a and b, plus a root node c. Let the rewards $r_a = r_b = 1$ and $r_c = 0$. Let $p(a|x) = \Pr(Y = a|X = x)$ be the posterior probability of node a, which completely determines the posterior distribution.

Assuming that ties are broken alphabetically, using Equation 6.6, it is easy to verify the following.

For $0 \le \lambda \le 1$,

$$f_\lambda(x) = \begin{cases} a, & p(a|x) \ge 0.5 \\ b, & p(a|x) < 0.5 \end{cases} . \tag{6.8}$$

For $\lambda > 1$,

$$f_\lambda(x) = \begin{cases} a, & p(a|x) \ge \frac{\lambda}{\lambda+1} \\ c, & \frac{1}{\lambda+1} < p(a|x) < \frac{\lambda}{\lambda+1} \\ b, & p(a|x) \le \frac{1}{\lambda+1} \end{cases} . \tag{6.9}$$

Suppose $p(a|x)$ only takes two discrete values, $p_1 = 0.6$ and $p_2 = 0.4$. Let $\mu(p)$ be the percentage of examples such that $p(a|x) = p$. Here let $\mu(p_1) = \mu(p_2) = 0.5$, that is, half of examples have a posterior probability of 0.6, the other half 0.4. Then it is simple to confirm that for $0 \le \lambda < 1.5$, $\Phi(f_\lambda) = 0.6$, i.e., all examples are predicted to the leaf nodes. For $\lambda \ge 1.5$, $\Phi(f_\lambda) = 1$, i.e., all examples are predicted to the root node. Therefore $\Phi(f_\lambda)$ is discontinuous at $\lambda = 1.5$ with a gap between 0.6 and 1. Thus for $1 - \epsilon \in (0.6, 1)$, there exists no $\lambda^\dagger$ such that $\Phi(f_{\lambda^\dagger}) = 1 - \epsilon$.

Note that in this example, the distribution of $p(a|x)$ concentrates on 0.4 and 0.6, violating our optimality condition stated in Theorem 6.4.3.

6.4.2.2 Our condition versus the one in [135]

In this section we discuss the difference between our condition in Theorem 6.4.3 and the one in [135], and show that their condition is insufficient to establish the optimality of DARTS by giving an example where their condition is satisfied but DARTS cannot con-

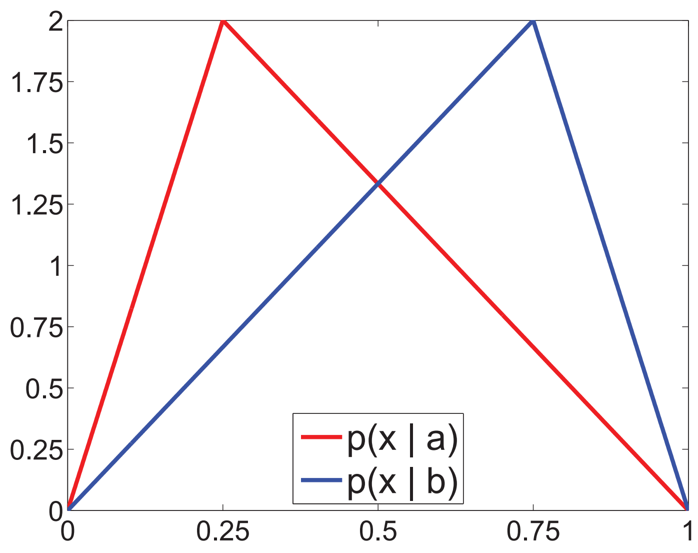


Figure 6.4: Probability density functions $p(x|a)$ and $p(x|b)$ for the example in Section 6.4.2.2.

verge to an optimal solution for certain $1-\epsilon$.

Consider the same hierarchy as in Section 6.4.2.1, i.e., a hierarchy of two leaf nodes a and b, plus a root node c. Let X be a one dimensional feature, specifically a real number on $[0,1]$. Let $p(x|a)$ be the density function of X given $Y=a$ and we set

$$p(x|a) = \begin{cases} 8x, & x \in [0, \frac{1}{4}) \\ \frac{8}{3}(1-x), & x \in [\frac{1}{4}, 1] \end{cases} .$$

Let $p(x|b)$ be the density function of X given $Y=b$ and we set

$$p(x|b) = \begin{cases} \frac{8}{3}x, & x \in [0, \frac{3}{4}) \\ 8(1-x), & x \in [\frac{3}{4}, 1] \end{cases} .$$

Then $p(x|a)$ and $p(x|b)$ are both continuous and thus satisfy the condition in [135]. We plot $p(x|a)$ and $p(x|b)$ in Figure 6.4.

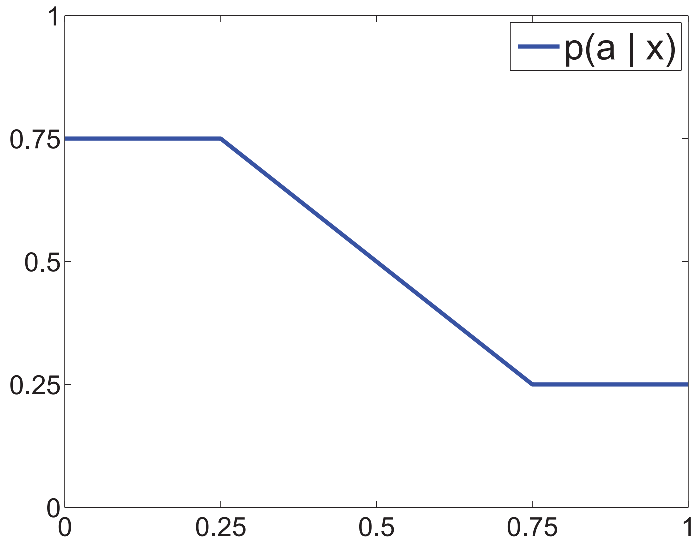


Figure 6.5: Posterior probability $p(a|x)$ for the example in Section 6.4.2.2.

Further assume that $\Pr(Y = a) = \Pr(Y = b) = \frac{1}{2}$. By Bayes' law,

$$p(a|x) = \begin{cases} 3/4, & x \in [0, 1/4] \\ 1 - x, & x \in (1/4, 3/4) \\ 1/4, & x \in [3/4, 1] \end{cases} .$$

We plot $p(a|x)$ in Figure 6.5.

Now consider f_λ. Eqn. 6.9 implies that if $\lambda > 3$, $\Phi(f_\lambda) = 1$, i.e., every example is predicted to the root node. If $\lambda = 3$, all examples x in $[0, 1/4]$ are predicted to node a with $3/4$ of them being correct and all examples x in $[3/4, 1]$ are predicted to node b with $3/4$ of them being correct. The rest, i.e., $x \in (1/4, 3/4)$ is predicted to the root node c with all of them being correct. Therefore with $\lambda = 3$, $\Phi(f_\lambda) = 3/4 \times 1/2 + 1 \times 1/2 = 7/8$.

Thus $\Phi(f_\lambda)$ is discontinuous at $\lambda = 3$ and for $1 - \epsilon \in (7/8, 1)$ there exists no $\lambda^\dagger$ such that $\Phi(f_{\lambda^\dagger}) = 1 - \epsilon$.

Now we can show that DARTS fails to converge to an optimal solution. Let's set $1-\epsilon = 15/16$. The binary search in DARTS returns $\lambda' = 3 + \delta$ where $\delta > 0$ is a small number. We then have $\Phi(f_{\lambda'}) = 1$ and $R(f_{\lambda'}) = 0$ since all examples are predicted to the root node.

Consider a classifier g:

$$g(x) = \begin{cases} a, & x \in [0, 1/16] \\ c, & x \in (1/16, 1] \end{cases}.$$

Then $\Phi(g) \geq 15/16$ and $R(g) > 0$. Thus g is a better solution to OP1 than $f_{\lambda'}$.

The condition in [135] guarantees strong duality, i.e., an optimal solution to OP1 maximizes the Lagrange function that has a dual variable that minimizes the dual function. Thus under strong duality, if we can find all maximizers of the Lagrange function for any given dual variable, we then have a dual algorithm guaranteed to converge to an optimal solution. However, the maximizer is not necessarily unique and it can be impractical to find all of them. In fact, there can be infinitely many of them. DARTS finds a maximizer of the Lagrange function by only using posterior probabilities. This classifier is not necessarily optimal without certain conditions, as the example shows.

This example also shows that their condition does not imply ours, because otherwise by Theorem 6.4.3 DARTS would converge to an optimal solution under their condition. On the other hand, our condition does not imply theirs either, because $p(x|b)$ and $p(x|a)$ do not need to be continuous to satisfy our condition.

6.5 Experiments

6.5.1 Datasets.

We use three datasets ranging from 65 classes to 10K classes, ILSVRC65, ILSVRC1K, and ImageNet10K, all subsets of ImageNet. ILSVRC1K is the same as ILSVRC2010 [46] in Section 4.5 but named differently to make the number of categories clear. Table 6.1 lists the detailed statistics. We follow the train/val/test split in [46] for ILSRVC1K. ImageNet10K is the same as the one used in Section 3.3.1 except two differences: (1) we exclude images from the internal nodes because we require that all images have ground truth at leaf nodes; (2) we include all ancestors of the leaf nodes, which gives 263 additional internal

Dataset	Tr	Val	Ts	# Leaf	# Int	H
ILSVRC65	100	50	150	57	8	3
ILSVRC1K	1,261	50	150	1,000	676	17
ImageNet10K	428	214	213	7,404	3,043	19

Table 6.1: Dataset statistics: average number of images per class for training (Tr), validation (Val) and test (Ts), number of leaf and internal nodes, and height of the hierarchy (H).

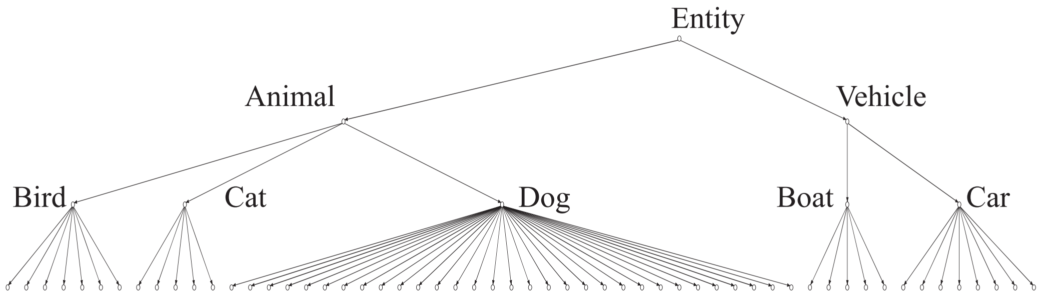


Figure 6.6: The tree structure of ILSVRC65.

nodes not used in Section 3.3.1. For ImageNet10K we use a 50-25-25 train/val/test split. ILSVRC65 is a subset of ILSVRC1K consisting of the leaf nodes of 5 “basic” categories (“dog,” ”cat,” etc. in Figure 6.6), with a simplified hierarchy and a down-sampled training size. The smaller scale allows thorough exploration of parameter space and comparison with baselines.

6.5.2 Implementation.

We represent all images using the LLC [16] features from densely sampled SIFT over a 16K codebook (10K for ILSVRC65) and a 2 level spatial pyramid (1×1 and 3×3). We train one-vs-all linear SVMs, convert the outputs of each SVM to probabilities via Platt scaling [121], and then L1 normalize them to get multiclass posterior probability estimates [126].

In implementing DARTS, we obtain f_λ using the training set but estimate $\Phi(f_\lambda)$, the expected accuracy of f_λ, using the validation set (step 6). This reduces overfitting. To ensure with high confidence that the true expected accuracy satisfies the guarantee, we compute the $.95$ confidence interval of the estimated $\Phi(f_\lambda)$ and stop the binary search when the lower bound is close enough to $1 - \epsilon$.

We also implement TREE-DARTS, a variant of DARTS that obtains posterior probabilities differently. It learns one-versus-all classifiers for each internal node to estimate the *conditional* posterior probabilities of the child nodes. It obtains the posterior probability of a node by multiplying all conditional posterior probabilities on its path from the root.

We compare DARTS with five baselines, LEAF-GT, TREE-GT, MAX-REW, MAX-EXP, MAX-CONF.

LEAF-GT is a naive extension of binary classification with a reject option. It takes the posterior probabilities on leaf nodes and predicts the most likely leaf node, if the largest probability is not below a fixed global threshold. Otherwise it predicts the root node. LEAF-GT becomes a flat classifier with threshold 0 and the trivial classifier that only predicts the root node with any threshold above 1.

TREE-GT takes the same *conditional* posterior probabilities in TREE-DARTS but moves an example from the root to a leaf, at each step following the branch with the highest conditional posterior probability. It stays at an internal node if the highest probability is below a fixed global threshold. This represents the decision tree model used in [31].

MAX-REW predicts the node with the best reward among those with probabilities greater than or equal to a threshold. Intuitively, it predicts the most specific node among the confident ones. MAX-EXP is similar to MAX-REW, except that it predicts the node with the best *expected* reward, i.e., its posterior probability times its reward.

MAX-CONF learns a binary, one-vs-all classifier for each node, including all internal nodes except the root node. Given a test image, it predicts the node with the most confident classifier. Despite being intuitive, this baseline is fundamentally flawed. First, assuming accurate confidences, the confidence of a node should never be more than that of its parent, i.e., we can never be more confident that something is a dog than that it is an animal. Thus in theory only the immediate children of the root node get predicted. Second, it is unclear how to satisfy an arbitrary accuracy guarantee—given the classifiers, the accuracy is fixed.

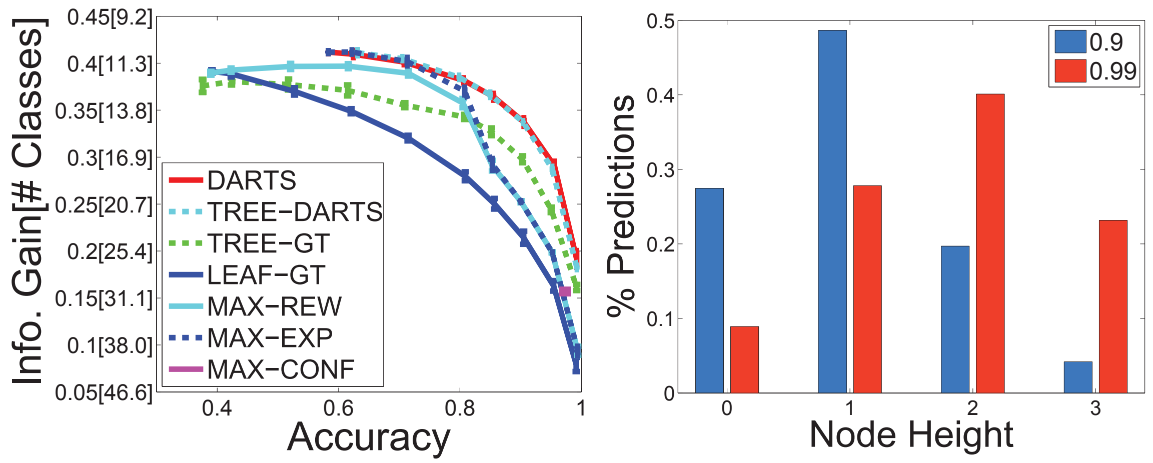


Figure 6.7: ILSVRC65 results. Left: Reward (normalized information gain, with 1 as the maximum possible) versus accuracy. The numbers in brackets on the Y axis indicate the equivalent of number of uncertain classes. The error bars are the standard deviation from 5 training sets, each with 100 images per class randomly sampled from a set of about 1,500 per class. Right: The distribution of predictions of DARTS with $.9$ and $.99$ accuracy guarantees.

For all threshold-based baselines, a higher threshold leads to higher accuracy and typically less reward in our experiments. Thus, to satisfy a particular accuracy guarantee, we find the best threshold by binary search.

We test all approaches on ILVRC65 but exclude TREE-DARTS, TREE-GT, MAX-CONF on ILSVRC1K and ImageNet10K, because both TREE-DARTS and TREE-GT require significant extension with a non-tree DAG—the child nodes overlap and there can be multiple paths from the root, possibly creating inconsistent probabilities—and because MAX-CONF is fundamentally unusable. We use information gain as reward and normalize it by the maximum possible (i.e., that of leaf nodes) such that the information gain of a flat classifier equals its accuracy.

6.5.3 Results on ILSVRC65

Figure 6.7 presents the reward-vs-accuracy curves. We set the accuracy guarantee $1 - \epsilon$ to $\{0, .1, .2, \dots, .8, .85, .9, .95, .99\}$ and plot the reward and the *actual* accuracy achieved on the test set. Note that all methods are able to satisfy an arbitrary accuracy guarantee, except

MAX-CONF that has a fixed accuracy.

First observe that the LEAF-GT curve starts with an accuracy and information gain both at $.391$, where the global threshold is too low to reject any example, making LEAF-GT equivalent to a flat classifier. The normalized information gain here equals the flat accuracy. In contrast, the DARTS curve starts with an accuracy of $.583$, achieved by maximizing the reward with a low, inactive accuracy guarantee. This is much higher than the flat accuracy $.391$ because the rewards on internal nodes already attract some uncertain examples that would otherwise be predicted to leaf nodes. Moreover, DARTS gives more correct information than the flat classifier ($.412$ versus $.391$); at this point our classifier is better than a flat classifier in terms of *both* accuracy and information gain. As we increase the accuracy guarantee, specificity is traded off for better accuracy and the information gain drops.

To interpret the information gain, we provide the equivalent number of uncertain leaf classes in Figure 6.7 (left). For example, at $.9$ accuracy, on average DARTS gives the same amount of correct information as a classifier that always correctly predicts an internal node with 14.57 leaf nodes.

Figure 6.7 (left) shows that both versions of DARTS significantly beat the baselines, validating our analysis on the optimality of DARTS. Interestingly both versions perform equally well, suggesting that DARTS is not sensitive to the particular means of estimating posterior probabilities.

Figure 6.7 (right) plots the distribution of predictions over different semantic levels for DARTS. As the accuracy guarantee increases, the distribution shifts toward the root node. At $.9$ accuracy, our classifier predicts leaf nodes 27% of the time and one of the 5 basic classes 49% of the time. Given that the flat accuracy is only $.391$, this is a useful trade-off with a high accuracy and a good amount of information.

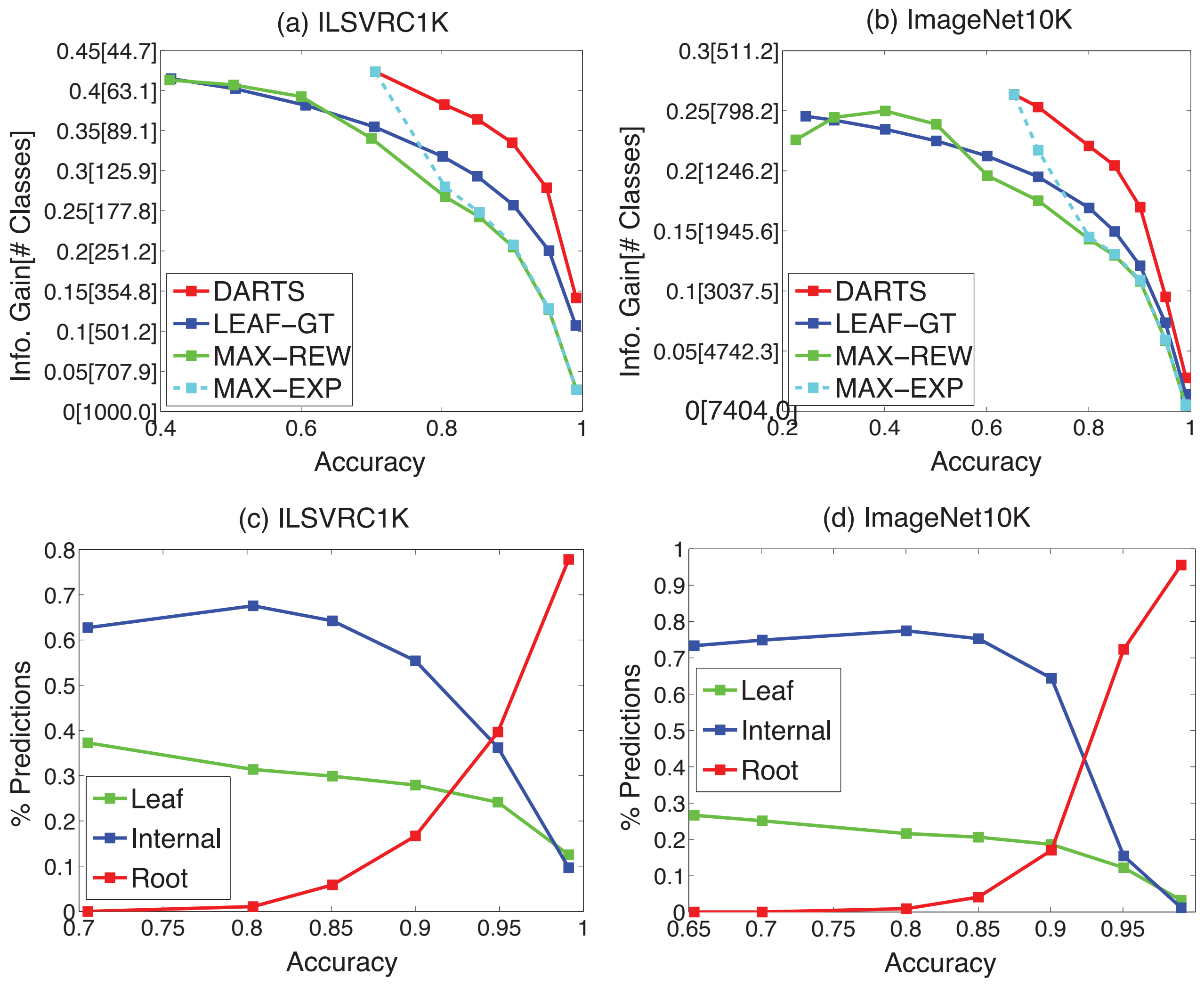


Figure 6.8: Large scale results: reward-vs-accuracy curves and distributions of predictions. Here the “internal nodes” exclude the root.

6.5.4 Results on ILSVRC1K and ImageNet10K

Figure 6.8a and Figure 6.8b present the reward-vs-accuracy curves for ILSVRC1K and ImageNet10K. On both datasets, DARTS achieves large improvements over the baselines. Also, at the start of the DARTS curve on ILSVRC1K (i.e., with an inactive accuracy guarantee), DARTS beats the flat classifier (the start of the LEAF-GT curve) on both information gain ($.423$ versus $.415$) *and* accuracy ($.705$ versus $.415$).

Figure 6.8c and Figure 6.8d show how the distribution of predictions changes with accuracy for DARTS. As accuracy increases, more examples are predicted to non-root internal nodes instead of leaf nodes. Eventually almost all examples move to the root node. On ILSVRC1K at $.9$ accuracy, 28% of the examples are predicted to leaf nodes, 55% to non-root internal nodes, and only 17% to the root node (i.e., the classifier declares "entity"). On ImageNet10K, the corresponding numbers are 19%, 64%, and 17%. Given the difficulty of problem, this is encouraging.

Figure 6.10 visually compares the confusion matrices of a flat classifier and our classifier, showing that our classifier significantly reduces the confusion among leaf nodes. Figure 6.9 shows examples of "hard" images for which the flat classifier makes mistakes whereas our classifier remains accurate.

We remark that in all of our experiments, DARTS either returns $\lambda = 0$ or is able to get sufficiently close to the accuracy guarantee in the binary search, as shown by all trade-off curves. This validates our analysis that, under practical conditions, DARTS converges to an optimal solution.

6.5.5 Zero-shot Recognition.

Another advantage of our classifier over a flat one is the ability of zero-shot recognition: classifying images from an unseen class *whose name is also unknown*. The flat classifier completely fails with 0 accuracy and 0 information gain. Our classifier, however, can predict internal nodes to "hedge its bets." Figure 6.12 shows the performance of our classifier

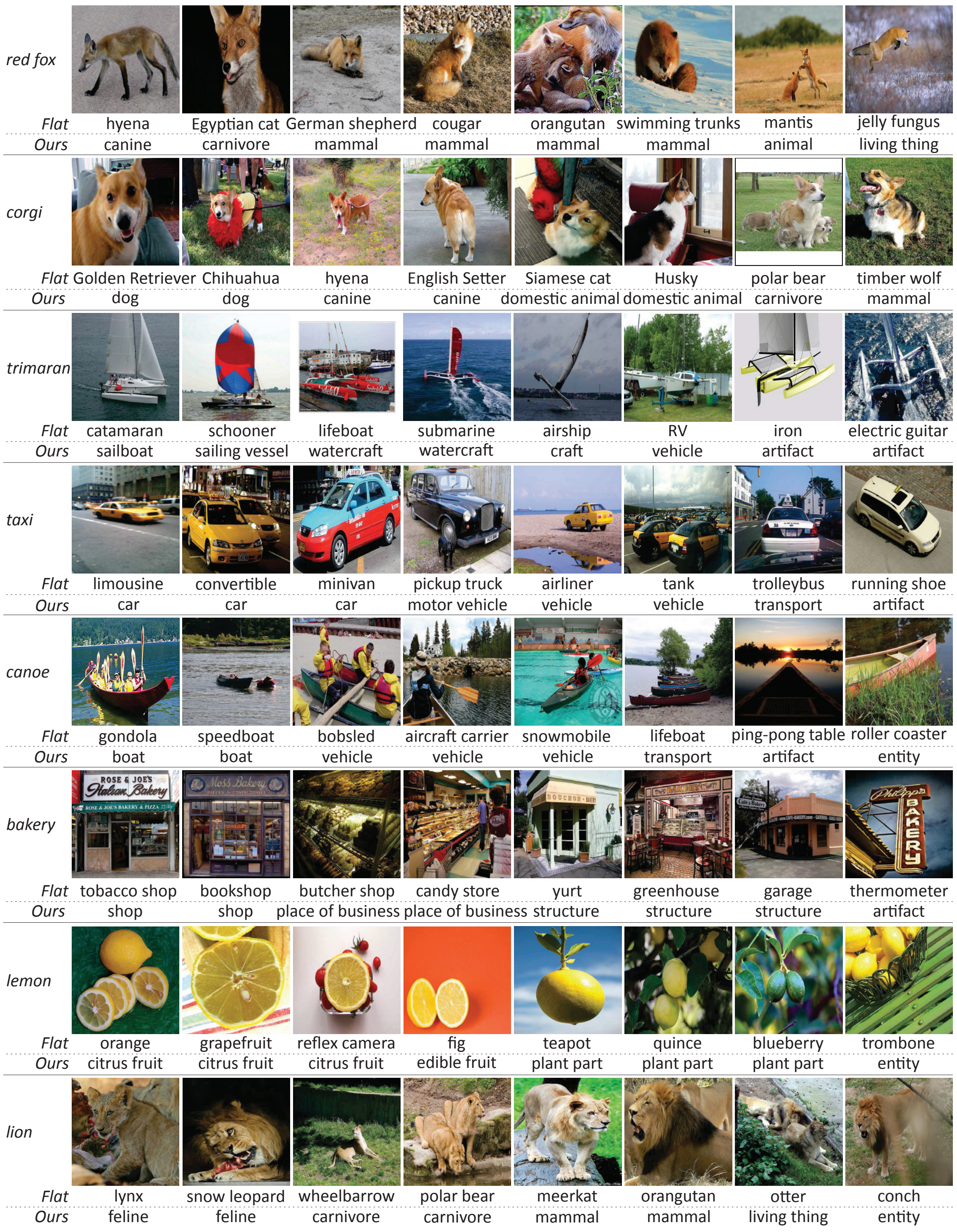

red fox
Flat hyena Egyptian cat German shepherd cougar orangutan swimming trunks mantis jelly fungus
Ours canine carnivore mammal mammal mammal mammal animal living thing
corgi
Flat Golden Retriever Chihuahua hyena English Setter Siamese cat Husky polar bear timber wolf
Ours dog dog canine canine domestic animal domestic animal carnivore mammal
trimaran
Flat catamaran schooner lifeboat submarine airship RV iron electric guitar
Ours sailboat sailing vessel watercraft watercraft craft vehicle artifact artifact
taxi
Flat limousine convertible minivan pickup truck airliner tank trolleybus running shoe
Ours car car car motor vehicle vehicle vehicle transport artifact
canoe
Flat gondola speedboat bobsled aircraft carrier snowmobile lifeboat ping-pong table roller coaster
Ours boat boat vehicle vehicle vehicle transport artifact entity
bakery
Flat tobacco shop bookshop butcher shop candy store yurt greenhouse garage thermometer
Ours shop shop place of business place of business structure structure structure artifact
lemon
Flat orange grapefruit reflex camera fig teapot quince blueberry trombone
Ours citrus fruit citrus fruit citrus fruit edible fruit plant part plant part plant part entity
lion
Flat lynx snow leopard wheelbarrow polar bear meerkat orangutan otter conch
Ours feline feline carnivore carnivore mammal mammal living thing entity

Figure 6.9: "Hard" test images in ILSVRC1K and the predictions made by a flat classifier and our classifier with a .8 accuracy guarantee. The flat classifier makes mistakes whereas ours stays accurate by "hedging its bets."

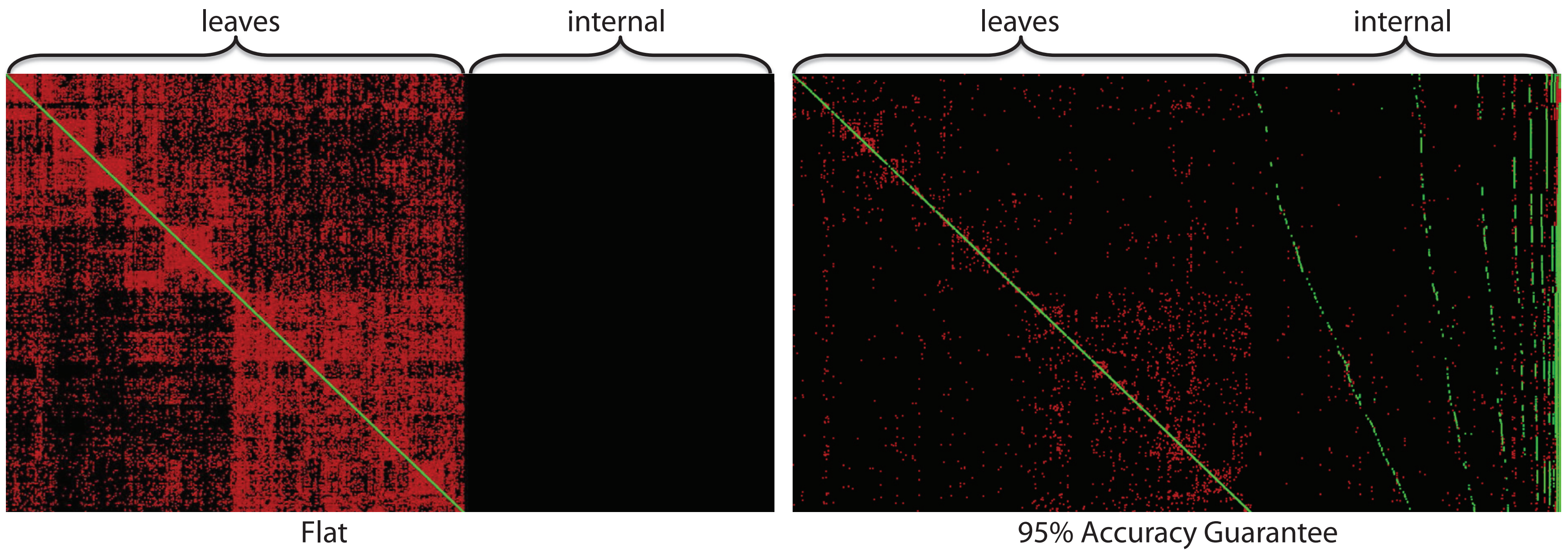


Figure 6.10: Comparison of confusion matrices on ILSVRC1K classes between a flat classifier and our classifier with a $.95$ accuracy guarantee. The rows represent leaf nodes; the columns are ordered from leaf to root by node height and then by the DFS order of the hierarchy. The matrices are downsampled; each pixel represents the max confusion between 4×4 entries. Correct predictions are colored green and incorrect ones red.

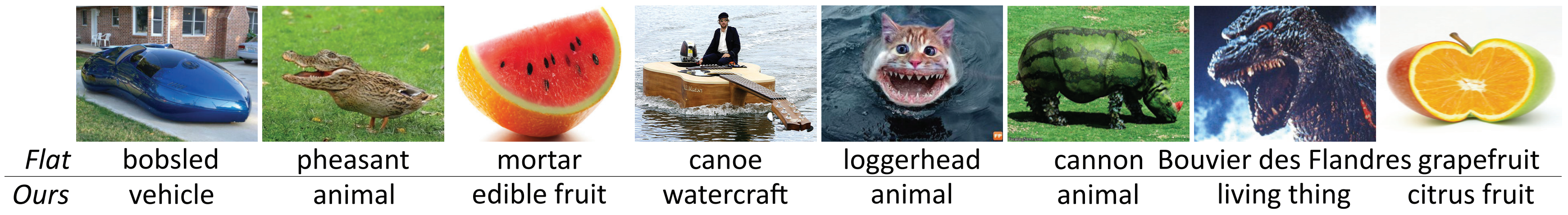


Figure 6.11: Predictions of "unusual" images by the flat classifier versus ours with a .7 accuracy guarantee, both trained on ILSVRC1K.

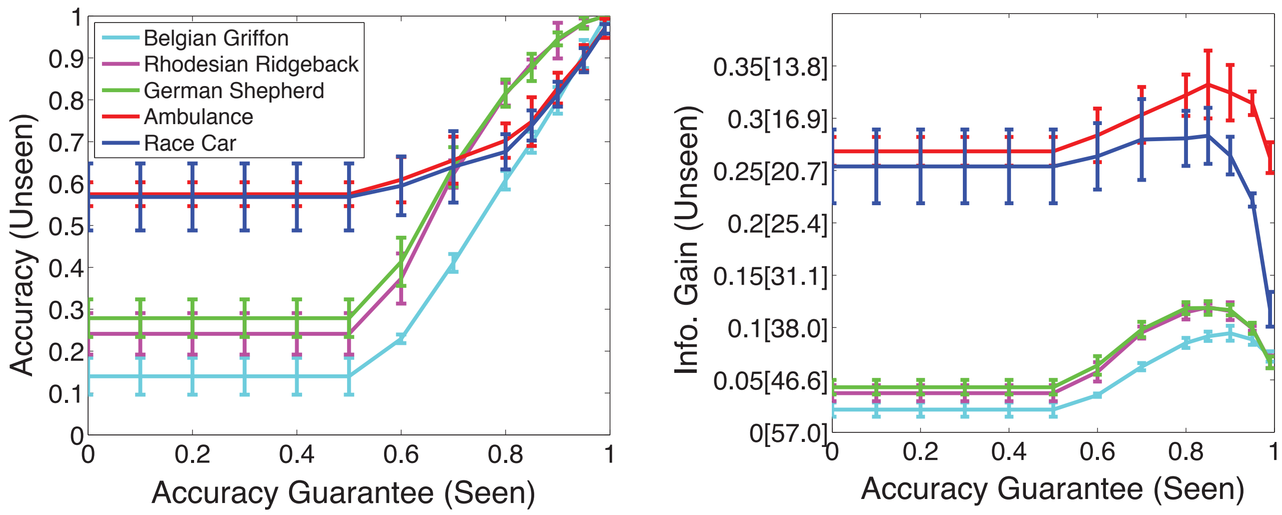


Figure 6.12: Zero-shot recognition: accuracy and information gain of 5 randomly chosen unseen classes versus accuracy guarantees on seen classes on ILSVRC65.

on 5 randomly chosen classes of ILSVRC65, taken out of the training set *and* the hierarchy. Our classifier is able to predict the correct internal nodes a significant amount of the time and with non-trivial information gain. Our final experiment is recognizing "unusual objects," objects that defy categorization at the subordinate levels. Figure 6.11 compares the predictions of a flat classifier versus our classifier, both trained on ILSVRC1K. We observe that the flat classifier is confused whereas our classifier stays sensible.

6.6 Summary

We have introduced the problem of optimizing accuracy-specificity trade-offs in large scale recognition. We have also presented the DARTS algorithm and its theoretical analysis, and demonstrated its effectiveness on large scale datasets with as many as 10K classes. Our algorithm makes it possible to guarantee an arbitrary accuracy while giving informative answers. On 10K classes, it can guarantee a 90% accuracy and at the same time give non-trivial answers 83% of the time. This is in deep contrast with the accuracy of only 24.5% for a flat classifier restricted to leaf nodes. [3] This result is significant because it shows that at large scale, even a low flat accuracy can still be made useful through our algorithm. This is an encouraging step toward a practical large scale recognition system.

[3]The flat accuracy in this chapter is much higher than that in Chapter 3 (around 8% for ImageNet7K, or the leaf nodes of ImageNet10K) due to improvements in feature representations.

Chapter 7

Conclusion and Future Work

7.1 Contributions

This dissertation has contributed to large scale visual recognition in two major aspects. One is the collection of large scale data. In particular, we have constructed ImageNet through crowdsourcing (Chapter 2) and used it to benchmark previous state of the art recognition algorithms (Chapter 3). The other is large scale learning. Specifically, we have studied three problems in large scale learning: efficient classification (Chapter 4), hierarchy-aware retrieval (chapter 5), and multi-level classification with accuracy guarantees (Chapter 6).

7.1.1 Constructing ImageNet

Data is the foundation of machine learning. We have constructed ImageNet, the first large scale labeled dataset with 22 thousand categories and 14 million human-verified images (Chapter 2). The images are mapped to the semantic structure of WordNet, a standard linguistic database. The large scale of ImageNet makes it well suited as a training and benchmarking resource for large scale recognition. Its rich semantic structure opens up opportunities for us to advance vision related research by understanding and exploiting the semantic relations between the categories.

7.1.2 Benchmarking with ImageNet

The construction of ImageNet enables us to evaluate previous state of the art algorithms at a scale never tested before. Through a study of image classification on 10,184 categories, we find that (1) computational issues become crucial in algorithm design; (2) the accuracy of the previous state of the art on 10,184 classes is quite low, only 6.4% (3) conventional wisdom from a couple of hundred image categories regarding the relative performance of different classifiers does not necessarily hold when the number of categories increases; (4) there is a surprisingly strong relationship between the structure of WordNet and the difficulty of visual categorization. These findings point to promising future research directions.

7.1.3 Efficient Large Scale Classification

The thrust of our contribution on large scale learning is exploiting the relations between categories. There are two ways that categories are related with each other—visually and semantically. In Chapter 3 we have shown that these these two types of relations are highly correlated and both of them give rise to a hierarchical structure. Exploiting the visual relations, we have improved the state of the art learning algorithm for efficient large scale classification (Chapter 4). Experiments on 10,184 classes and 9 million images demonstrate that our algorithm achieves better accuracy and test-time efficiency yet is 31 times faster in learning compared to the previous state of the art by Bengio et al [26].

7.1.4 Hierarchy-Aware Large Scale Retrieval

In addition to exploiting visual relations among categories, we also take advantage of the semantic relations. In Chapter 5, we studied large scale image retrieval in the context of a large number of classes, a problem closely related to classification. The core novel contribution is an approach that can exploit prior knowledge of a semantic hierarchy. We show significant improvements over OASIS [105], the previous state of the art for similarity learning. An additional contribution is a novel hashing scheme (for bilinear similarity on vectors of probabilities, optionally taking into account hierarchy) that is able to reduce the computational cost of retrieval by 1,000 times while achieving close to 90% of the accuracy of brute force.

7.1.5 Multi-level Classification with Accuracy Guarantees

Further exploiting the semantic relations, we have proposed in Chapter 6 an "infallible" classifier that tries to be as specific as possible yet almost never makes mistakes, based on the observation that a classifier can choose the level of specificity to guarantee an arbitrary accuracy. This leads to the novel problem of optimizing the accuracy-specificity trade-off on a semantic hierarchy in large scale recognition. We have proposed an algorithm that is provably optimal under practical conditions. Experiments on 10K categories have demonstrated that our approach can produce a classifier that has 90% accuracy and still give informative answers 83% of the time. This holds promise toward a real, practical large scale recognition system.

7.2 Future Work

Large scale visual recognition remains a challenging and active research area. The following directions appear particularly promising based on the findings presented in this dissertation.

7.2.1 Image Representation

Powerful and efficient image representations are necessary for recognizing tens of thousands of categories. This issue remains to be explored, as most current representations do not perform as well for fine-grained classes as for "sparser" classes (Chapter 3). The key question is what the next step is. One approach would be *learning oriented*, that is, seeking better generic representations as well as learning algorithms such that the discriminative features of fine-grained classes can be captured with a unified mechanism. Another approach would be *knowledge oriented*. The idea is that a lot of domain-specific knowledge is necessary to tell the fine-grained classes apart. Thus we should encode more prior knowledge into the representation. Both approaches have merits and the ultimate solution could well be a combination of the two.

7.2.2 Crowdsourcing

Deeper annotations such as bounding boxes, object contours, and part correspondences help recognition algorithms tremendously by providing strong supervision in learning. However, to crowd-source these tasks on sites such as Amazon Mechanical Turk, where workers are by and large financially motivated, the cost can be prohibitively high — these tasks are much more time-consuming than the verification tasks in constructing ImageNet. For this reason there has been no dataset of fully parsed images at a scale comparable to ImageNet. A promising direction is to explore low or zero cost approaches to crowdsourcing, in particular, online games [144]. The research questions would include how to design the games and how to control quality. The goal is to have complex tasks done during enjoyable game play, with high quality, and of a large scale. In addition, a closely related question is how to develop active learning techniques [145] to optimize the selection of images and the granularity of annotation.

7.2.3 Systems

Visual data in general tend to be high dimensional and dense. This, coupled with a large number of images and categories, easily generates an overwhelming amount of data. In this case both data and computation must be distributed, which poses challenges on both machine learning and systems. On the machine learning front, one challenge is how to develop distributed visual learning algorithms that can scale to hundreds, even thousands of computing nodes in the cloud, especially for algorithms that are not already "embarrassingly parallel." On the systems front, the challenge is how to develop new programming models that facilitate visual learning on a highly distributed infrastructure.

7.2.4 Large Scale Image Parsing

Current work in large scale visual recognition mostly focuses on classification tasks, without specifying the location, pose or spatial support of the object. This brings forward the even more challenging problem of full image parsing with a large number of categories, that is, labeling every single pixel in an image and forming a hierarchical representation that relates the pixels, parts, objects, and scenes with each other. This calls for approaches that can handle tens of thousands of categories in terms of not only their semantic relations ("is a," "part of," "instance of," etc.) but also their spatial interactions ("next to," "above," "behind," etc.). Research in this direction will eventually lead to a system that can answer any queries about any image, with an accuracy and specificity unmatched even by a very knowledgeable human.

Bibliography

[1] Felleman, D.J., Van Essen, D.C.: Distributed hierarchical processing in the primate cerebral cortex. Cerebral Cortex **1** (1991) 1–47

[2] Biederman, I.: Recognition by components: A theory of human image understanding. PsychR **94** (1987) 115–147

[3] Sigurbjörnsson, B., van Zwol, R.: Flickr tag recommendation based on collective knowledge. In: WWW. (2008) 327–336

[4] Fei-Fei, L., Fergus, R., Perona, P.: One-shot learning of object categories. PAMI **28** (2006) 594–611

[5] Griffin, G., Holub, A., Perona, P.: Caltech-256 object category dataset. Technical Report 7694, Caltech (2007)

[6] Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. (http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html)

[7] Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2008 (VOC2008) Results. (http://www.pascal-network.org/challenges/VOC/voc2008/workshop/index.html)

[8] Halevy, A., Norvig, P., Pereira, F.: The unreasonable effectiveness of data. IEEE Intelligent Systems **24** (2009) 8–12

[9] Rowley, H., Baluja, S., Kanade, T.: Neural network-based face detection. Pattern Analysis and Machine Intelligence, IEEE Transactions on **20** (1998) 23 –38

[10] Viola, P., Jones, M.J.: Robust real-time face detection. Int. J. Comput. Vision **57** (2004) 137–154

[11] (http://blog.flickr.net/en/2011/08/04/6000000000/)

[12] (http://blog.facebook.com/blog.php?post=206178097130)

[13] Lowe, D.: Distinctive image features from scale-invariant keypoints. IJCV **60** (2004) 91–110

[14] Grauman, K., Darrell, T.: The pyramid match kernel: Discriminative classification with sets of image features. In: ICCV. (2005)

[15] Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: CVPR06. (2006)

[16] Wang, J., Yang, J., Yu, K., Lv, F., Huang, T., Gong, Y.: Locality-constrained linear coding for image classification. In: CVPR. (2010)

[17] Dance, C., Willamowski, J., Fan, L., Bray, C., Csurka, G.: Visual categorization with bags of keypoints. In: ECCV International Workshop on Statistical Learning in Computer Vision. (2004)

[18] Zhang, H., Berg, A.C., Maire, M., Malik, J.: SVM-KNN: Discriminative Nearest Neighbor Classification for Visual Category Recognition. In: CVPR06. (2006)

[19] Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR05. (2005) 886–893

[20] Perronnin, F., Sánchez, J., Mensink, T.: Improving the fisher kernel for large-scale image classification. In: ECCV. (2010)

[21] Zhou, X., Yu, K., Zhang, T., Huang, T.S.: Image classification using super-vector coding of local image descriptors. In: Proceedings of the 11th European conference on Computer vision: Part V. ECCV'10, Berlin, Heidelberg, Springer-Verlag (2010) 141–154

[22] Nister, D., Stewenius, H.: Scalable recognition with a vocabulary tree. In: CVPR. (2006)

[23] Marszalek, M., Schmid, C.: Semantic hierarchies for visual object recognition. In: CVPR. (2007)

[24] Zweig, A., Weinshall, D.: Exploiting object hierarchy: Combining models from different category levels. In: ICCV. (2007)

[25] Griffin, G., Perona, P.: Learning and using taxonomies for fast visual categorization. CVPR08 (2008)

[26] Bengio, S., Weston, J., Grangier, D.: Label embedding trees for large multi-class tasks. In: NIPS. (2010)

[27] Gao, T., Koller, D.: Discriminative learning of relaxed hierarchy for large-scale visual recognition. In: ICCV. (2011)

[28] Marszalek, M., Schmid, C.: Constructing category hierarchies for visual recognition. In: ECCV. (2008)

[29] Amit, Y., Fink, M., Srebro, N.: Uncovering shared structures in multiclass classification. In: ICML. (2007)

[30] Fergus, R., Bernal, H., Weiss, Y., Torralba, A.: Semantic label sharing for learning with many categories. (In: ECCV)

[31] Vailaya, A., Figueiredo, M.A.T., Jain, A.K., Zhang, H.: Content-based hierarchical classification of vacation images. In: ICMCS, Vol. 1. (1999) 518–523

[32] Everingham, M., Zisserman, A., Williams, C.K.I., Van Gool, L.: The PASCAL Visual Object Classes Challenge 2006 (VOC2006) Results. (http://www.pascal-network.org/challenges/VOC/voc2006/results.pdf)

[33] Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2009 (VOC2009) Results. (http://www.pascal-network.org/challenges/VOC/voc2009/workshop/index.html)

[34] Russell, B., Torralba, A., Murphy, K., Freeman, W.: Labelme: A database and web-based tool for image annotation. IJCV **77** (2008) 157–173

[35] Yao, B., Yang, X., Zhu, S.: Introduction to a large-scale general purpose ground truth database: Methodology, annotation tool and benchmarks. In: EMMCVPR07. (2007) 169–183

[36] Torralba, A., Fergus, R., Freeman, W.: 80 million tiny images: A large data set for nonparametric object and scene recognition. PAMI **30** (2008) 1958–1970

[37] Fellbaum, C.: WordNet: An Electronic Lexical Database. Bradford Books (1998)

[38] Bottou, L., Bousquet, O.: The tradeoffs of large scale learning. Advances in neural information processing systems **20** (2008) 161–168

[39] Dean, J., Ghemawat, S.: Mapreduce: simplified data processing on large clusters. Commun. ACM **51** (2008) 107–113

[40] Chu, C.T., Kim, S.K., Lin, Y.A., Yu, Y., Bradski, G., Ng, A.Y., Olukotun, K.: Map-reduce for machine learning on multicore. In Schölkopf, B., Platt, J., Hoffman, T., eds.: Advances in Neural Information Processing Systems 19. MIT Press, Cambridge, MA (2007) 281–288

[41] Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Foundations and Trends in Machine Learning **3** (2011) 1–122

[42] Beygelzimer, A., Langford, J., Ravikumar, P.: Multiclass classification with filter trees. Preprint, June (2007)

[43] Beygelzimer, A., Langford, J., Lifshits, Y., Sorkin, G.B., Strehl, A.L.: Conditional probability tree estimation analysis and algorithms. Computing Research Repository (2009)

[44] Deng, J., Dong, W., Socher, R., Li, L., Li, K., Fei-Fei, L.: ImageNet: A large-scale hierarchical image database. In: CVPR. (2009)

[45] (http://www.mturk.com)

[46] (`http://www.image-net.org/challenges/LSVRC/2010/`)

[47] (`http://www.image-net.org/challenges/LSVRC/2011/`)

[48] Rohrbach, M., Stark, M., Schiele, B.: Evaluating knowledge transfer and zero-shot learning in a large-scale setting. In: CVPR. (2011) (code available at http://www.d2.mpi-inf.mpg.de/large-scale-kt).

[49] Rastegari, M., Fang, C., Torresani, L.: Scalable object-class retrieval with approximate and top-k ranking. In: ICCV. (2011) 2659–2666

[50] Leong, C.W., Mihalcea, R.: Measuring the semantic relatedness between words and images. In: Proceedings of the Ninth International Conference on Computational Semantics. IWCS '11, Stroudsburg, PA, USA, Association for Computational Linguistics (2011) 185–194

[51] Ma, X., Fellbaum, C., Cook, P.R.: A multimodal vocabulary for augmentative and alternative communication from sound/image label datasets. In: Proceedings of the

NAACL HLT 2010 Workshop on Speech and Language Processing for Assistive Technologies. SLPAT '10, Stroudsburg, PA, USA, Association for Computational Linguistics (2010) 62–70

[52] Deng, J., Berg, A.C., Li, K., Fei-Fei, L.: What does classifying more than 10, 000 image categories tell us? In: ECCV. (2010)

[53] Deng, J., Satheesh, S., Berg, A.C., Fei-Fei, L.: Fast and balanced: Efficient label tree learning for large scale object recognition. In: NIPS. (2011)

[54] Deng, J., Berg, A.C., Fei-Fei, L.: Hierarchical semantic indexing for large scale image retrieval. In: CVPR. (2011)

[55] Andoni, A., Indyk, P.: Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In: FOCS06. (2006) 459–468

[56] Deng, J., Krause, J., Berg, A.C., Fei-Fei, L.: Hedging your bets: Optimizing accuracy-specificity trade-offs in large-scale visual recognition. In: CVPR. (2012)

[57] (http://www.emc.com/collateral/demos/microsites/emc-digital-universe 2011/index.htm)

[58] von Ahn, L., Dabbish, L.: Labeling images with a computer game. In: CHI04. (2004) 319–326

[59] Phillips, P., Wechsler, H., Huang, J., Rauss, P.: The feret database and evaluation procedure for face-recognition algorithms. IVC **16** (1998) 295–306

[60] Huang, G., Ramesh, M., Berg, T., Learned Miller, E.: Labeled faces in the wild: A database for studying face recognition in unconstrained environments. In: UMass. (2007)

[61] Fink, M., Ullman, S.: From aardvark to zorro: A benchmark for mammal image classification. IJCV **77** (2008) 143–156

[62] Shotton, J., Winn, J., Rother, C., Criminisi, A.: Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In: ECCV06. (2006) I: 1–15

[63] Rosch, E., Lloyd, B.: Principles of categorization. In: Cognition and categorization. (1978) 27–48

[64] (`http://www.hunch.net/~jl/`)

[65] Fergus, R., Fei-Fei, L., Perona, P., Zisserman, A.: Learning object categories from google's image search. In: ICCV05. (2005) II: 1816–1823

[66] : The Spanish WordNet. (`http://www.lsi.upc.edu/~nlp`)

[67] : The Chinese WordNet. (`http://bow.sinica.edu.tw`)

[68] Artale, A., Magnini, B., C., S.: Wordnet for italian and its use for lexical discrimination. In: AI*IA97. (1997) 16–19

[69] Vossen, P., Hofmann, K., de Rijke, M., Tjong Kim Sang, E., Deschacht, K.: The Cornetto database: Architecture and user-scenarios. In: Proceedings DIR 2007. (2007) 89–96

[70] Sorokin, A., Forsyth, D.: Utility data annotation with amazon mechanical turk. In: InterNet08. (2008) 1–8

[71] Varma, M., Ray, D.: Learning the discriminative power-invariance trade-off. In: ICCV. (2007)

[72] Felzenszwalb, P., Mcallester, D., Ramanan, D.: A discriminatively trained, multiscale, deformable part model. In: CVPR08. (2008)

[73] Tuytelaars, T., Mikolajczyk, K.: Local Invariant Feature Detectors: A Survey. Foundations and Trends in Computer Graphics and Vision **3** (2008) 177–820

[74] Fei-Fei, L., Fergus, R., Torralba, A.: Recognizing and learning object categories. CVPR Short Course (2007)

[75] Fei-Fei, L., Fergus, R., Torralba, A.: Recognizing and learning object categories. ICCV Short Course (2009)

[76] Rosch, E., Mervis, C.B., Gray, W.D., Johnson, D.M., Braem, P.B.: Basic objects in natural categories. Cognitive Psychology **8** (1976) 382–439

[77] Everingham, M., Zisserman, A., Williams, C.K.I., van Gool, L., et al.: The 2005 pascal visual object classes challenge. In: Pascal Challenges Workshop. Volume 3944 of LNAI., Southampton, UK, Springer, Springer (2006) 117–176

[78] Chum, O., Zisserman, A.: An exemplar model for learning object classes. In: CVPR07. (2007) 1–8

[79] Vedaldi, A., Gulshan, V., Varma, M., Zisserman, A.: Multiple kernels for object detection. In: ICCV. (2009)

[80] Gehler, P.V., Nowozin, S.: On feature combination for multiclass object classification. In: ICCV. (2009)

[81] Rifkin, R., Klautau, A.: In defense of one-vs-all classification. JMLR **5** (2004) 101–141

[82] Maji, S., Berg, A.C.: Max-margin additive models for detection. In: ICCV. (2009)

[83] Fergus, R., Weiss, Y., Torralba, A.: Semi-supervised learning in gigantic image collections. In: NIPS. (2009)

[84] Wang, C., Yan, S., Zhang, H.J.: Large scale natural image classification by sparsity exploration. ICASP (2009)

[85] Oliva, A., Torralba, A.: Modeling the shape of the scene: A holistic representation of the spatial envelope. IJCV **42** (2001) 145–175

[86] Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.: LIBLINEAR: A library for large linear classification. JMLR **9** (2008) 1871–1874

[87] Crammer, K., Singer, Y., Cristianini, N., Shawe-Taylor, J., Williamson, B.: On the algorithmic implementation of multiclass kernel-based vector machines. JMLR **2** (2001)

[88] Boiman, O., Shechtman, E., Irani, M.: In defense of nearest-neighbor based image classification. In: CVPR08. (2008)

[89] Thorpe, S., Fize, D., Marlot, C.: Speed of processing in the human visual system. Nature **381** (1996) 520–522

[90] Martinez-Munoz, G., Larios, N., Mortensen, E., Zhang, W., Yamamuro, A., Paasch, R., Payet, N., Lytle, D., Shapiro, L., Todorovic, S., Moldenke, A., Dietterich, T.: Dictionary-free categorization of very similar objects via stacked evidence trees. CVPR (2009)

[91] Nilsback, M.E., Zisserman, A.: A visual vocabulary for flower classification. In: CVPR06. (2006) 1447–1454

[92] Ferencz, A., Learned-Miller, E.G., Malik, J.: Building a classification cascade for visual identification from one example. In: ICCV05. (2005) 286–293

[93] Vedaldi, A., Fulkerson, B.: VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org (2008)

[94] Lin, H.T., Lin, C.J., Weng, R.C.: A note on platt's probabilistic outputs for support vector machines. Mach. Learn. **68** (2007) 267–276

[95] Wang, J., Yang, J., Yu, K., Lv, F., Huang, T., Gong, Y.: Locality-constrained linear coding for image classification. (2010)

[96] Zhou, X., Yu, K., Zhang, T., Huang, T.: Image classification using super-vector coding of local image descriptors. Computer Vision–ECCV 2010 (2010) 141–154

[97] Yu, K., Zhang, T.: Improved local coordinate coding using local tangents. ICML09 (2010)

[98] Lin, Y., Lv, F., Zhu, S., Yang, M., Cour, T., Yu, K., Cao, L., Huang, T.: Large-scale image classification: Fast feature extraction and svm training. In: CVPR. (2011)

[99] Weiss, D., Sapp, B., Taskar, B.: Sidestepping intractable inference with structured ensemble cascades. In: NIPS. Volume 1281. (2010) 1282–1284

[100] Bertsimas, D., Tsitsiklis, J.: Introduction to linear optimization. (1997)

[101] Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2010 (VOC2010) Results. (http://www.pascal-network.org/challenges/VOC/voc2010/workshop/index.html)

[102] Zinkevich, M., Weimer, M., Smola, A., Li, L.: Parallelized stochastic gradient descent. In Lafferty, J., Williams, C.K.I., Shawe-Taylor, J., Zemel, R., Culotta, A., eds.: Advances in Neural Information Processing Systems 23. (2010) 2595–2603

[103] Crammer, K., Singer, Y.: On the algorithmic implementation of multiclass kernel-based vector machines. The Journal of Machine Learning Research **2** (2002) 265–292

[104] Branson, S., Wah, C., Schroff, F., Babenko, B., Welinder, P., Perona, P., Belongie, S.: Visual recognition with humans in the loop. In: ECCV. (2010)

[105] Chechik, G., Shalit, U., Bengio, S., Sonnenburg, S., Franc, V., Yom-tov, E., Sebag, M.: Large scale online learning of image similarity through
. JMLR (2010)

[106] Weinberger, K.Q., Blitzer, J., Saul, L.K.: Distance metric learning for large margin nearest neighbor classification. In: NIPS, MIT Press (2006)

[107] Globerson, A., Roweis, S.: Metric learning by collapsing classes. NIPS **18** (2006) 451–458

[108] Frome, A., Singer, Y., Sha, F., Malik, J.: Learning globally-consistent local distance functions for shape-based image retrieval and classification. In: ICCV. (2007) 1–8

[109] Kumar, N., Belhumeur, P.N., Nayar, S.K.: FaceTracer: A Search Engine for Large Collections of Images with Faces. In: ECCV. (2008) 340–353

[110] Kumar, N., Berg, A.C., Belhumeur, P.N., Nayar, S.K.: Attribute and Simile Classifiers for Face Verification. (In: ICCV09)

[111] Farhadi, A., Endres, I., Hoiem, D., Forsyth, D.: Describing objects by their attributes. In: CVPR. (2009)

[112] Lampert, C.H., Nickisch, H., Harmeling, S.: Learning to detect unseen object classes by between-class attribute transfer. In: CVPR. (2009)

[113] Torresani, L., Szummer, M., Fitzgibbon, A.: Efficient object category recognition using classemes. (In: ECCV)

[114] Aytar, Y., Shah, M., Luo, J.: Utilizing semantic word similarity measures for video retrieval. In: CVPR. (2008)

[115] Hauptmann, A.G., Christel, M.G., Yan, R.: Video retrieval based on semantic concepts. Proceedings of the IEEE **96** (2008) 602–622

[116] Dong, W., Charikar, M., Li, K.: Asymmetric distance estimation with sketches for similarity search in high-dimensional spaces. In: SIGIR '08. (2008)

[117] Kulis, B., Jain, P., Grauman, K.: Fast similarity search for learned metrics. PAMI **31** (2009) 2143–2157

[118] Salakhutdinov, R., Hinton, G.: Semantic hashing. Int. J. Approx. Reasoning **50** (2009) 969–978

[119] Torralba, A., Fergus, R., Weiss, Y.: Small codes and large image databases for recognition. (In: CVPR)

[120] Rohrbach, M., Stark, M., Szarvas, G., Gurevych, I., Schiele, B.: What Helps Where – And Why? Semantic Relatedness for Knowledge Transfer. In: CVPR. (2010)

[121] Platt, J.: Probabilistic outputs for support vector machines and comparison to regularize likelihood methods. In: Advances in Large Margin Classifiers. (2000) 61–74

[122] Provost, F., Domingos, P.: Well-trained pets: Improving probability estimation trees (2000)

[123] Charikar, M.S.: Similarity estimation techniques from rounding algorithms. In: STOC '02. (2002)

[124] Jain, P., Kulis, B., Dhillon, I.S., Grauman, K.: Online metric learning and fast similarity search. (In: NIPS08)

[125] Sánchez, J., Perronnin, F.: High-dimensional signature compression for large-scale image classification. In: CVPR. (2011)

[126] Zadrozny, B., Elkan, C.: Transforming classifier scores into accurate multiclass probability estimates. In: KDD. (2002)

[127] Masnadi-shirazi, H., Vasconcelos, N.: Risk minimization, probability elicitation, and cost-sensitive svms. In: ICML. (2010)

[128] Cesa-Bianchi, N., Gentile, C., Zaniboni, L.: Incremental algorithms for hierarchical classification. JMLR **7** (2006) 31–54

[129] Zhao, B., Fei-Fei, L., Xing, E.: Large-scale category structure aware image categorization. In: NIPS. (2011)

[130] Dekel, O., Keshet, J., Singer, Y.: Large margin hierarchical classification. In: ICML. (2004)

[131] Hwang, S.J., Grauman, K., Sha, F.: Learning a tree of metrics with disjoint visual features. In: NIPS. (2011)

[132] Yuan, M., Wegkamp, M.: Classification methods with reject option based on convex risk minimization. JMLR **11** (2010) 111–130

[133] El-Yaniv, R., Wiener, Y.: On the foundations of noise-free selective classification. JMLR **11** (2010) 1605–1641

[134] Ha, T.: The optimum class-selective rejection rule. PAMI **19** (1997) 608 –615

[135] Grall-Maes, E., Beauseroy, P.: Optimal decision rule with class-selective rejection and performance constraints. PAMI **31** (2009) 2073 –2082

[136] del Coz, J.J., Bahamonde, A.: Learning nondeterministic classifiers. JMLR **10** (2009) 2273–2293

[137] Xiao, J., Hays, J., Ehinger, K., Oliva, A., Torralba, A.: Sun database: Large-scale scene recognition from abbey to zoo. In: CVPR. (2010)

[138] Perronnin, F., Sánchez, J., Liu, Y.: Large-scale image categorization with explicit data embedding. In: CVPR. (2010)

[139] Farrell, R., Oza, O., Zhang, N., Morariu, V., Darrell, T., Davis, L.: Birdlets: Subordinate categorization using volumetric primitives and pose-normalized appearance. In: ICCV. (2011)

[140] Hillel, A., Weinshall, D.: Subordinate class recognition using relational object models. In: NIPS. (2007)

[141] Martinez-Munoz, G., Zhang, W., Payet, N., Todorovic, S., Larios, N., Yamamuro, A., Lytle, D., Moldenke, A., Mortensen, E., Paasch, R., Shapiro, L., Dietterich, T.: Dictionary-free categorization of very similar objects via stacked evidence trees. In: CVPR. (2009)

[142] Wah, C., Branson, S., Perona, P., Belongie, S.: Multiclass recognition and part localization with humans in the loop. In: ICCV. (2011)

[143] Everett, H.: Generalized lagrange multiplier method for solving problems of optimum allocation of resources. Operations Research **11** (1963) 399–417

[144] Cooper, S., Khatib, F., Treuille, A., Barbero, J., Lee, J., Beenen, M., Leaver-Fay, A., Baker, D., Popovi, Z., Players, F.: Predicting protein structures with a multiplayer online game. Nature **466** (2010) 756–760

[145] Collins, B., Deng, J., Li, K., Fei-Fei, L.: Towards scalable dataset construction: An active learning approach. In: ECCV. (2008)