

Learning Mid-Level Features from Object Hierarchy for Image Classification

by

Somayah A. Albaradei

A thesis submitted to
The Faculty of Graduate Studies of
The University of Manitoba
in partial fulfillment of the requirements
of the degree of

Master of Science

Department of Computer Science
The University of Manitoba
Winnipeg, Manitoba, Canada
August 2014

© Copyright 2014 by Somayah A. Albaradei

Thesis advisor

Yang Wang

Author

Somayah A. Albaradei

Learning Mid-Level Features from Object Hierarchy for Image Classification

Abstract

One of the most active research areas in computer vision is image classification. Although there have been many research efforts in this area, it remains a difficult problem, especially when the number of categories is large. Most of the previous work in image classification uses low-level image features. We believe low-level features ignore a lot of the semantic structures of the image classes. In this thesis, we go beyond simple low-level features and propose new approaches for constructing mid-level visual features for image classification. We represent an image using the outputs of a collection of binary classifiers. These binary classifiers are trained to differentiate pairs of object classes in an object hierarchy. Our feature representations implicitly capture the hierarchical structure in object classes. We show that our proposed approach outperforms other baseline methods in image classification.

Contents

Abstract	ii
Table of Contents	iii
List of Figures	iv
List of Tables	vii
Acknowledgments	viii
Dedication	ix
1 Introduction	1
2 Related work	7
2.1 Image classification/object recognition	7
2.2 Large scale image classification	8
2.3 Hierarchy	10
2.4 Mid-level features	11
3 Our approach	13
3.1 Approach A	14
3.1.1 Selecting concept pairs	15
3.1.2 Image representation	18
3.2 Approach B	19
3.2.1 Image representation using object hierachy	19
3.2.2 Prediction on unseen images	22
3.3 Building the tree-shaped hierarchy	25
4 Experiments	27
4.1 Datasets	27
4.2 Evaluation methodology	34
4.2.1 Experimental results of our approaches	36
4.2.2 Experimental results of building tree-shaped hierarchy	39
5 Conclusion	44

Bibliography**50**

List of Figures

1.1	Illustration of the tree-shaped hierarchy. Every leaf node represents an individual class label; such as, v_6, v_7, v_8 . The root node “living thing” is the union of all leaf nodes’ class labels. Every internal node represents a subset of leaf nodes’ class labels; such as, internal node $v_4 \in V$ represent the group $G_4 = v_6, v_7, v_8$	2
1.2	Intuitive example of our mid-level representation; (top) we have a subset of a hierarchy of object categories; (bottom) number of object pairs range from coarse pairs (e.g., “Canine” vs “Feline”), to fine-grained pairs: (e.g., “Shepard dog” vs “Dalmatian dog”). For the “Shepard dog” image, we use the output scores to form its mid-level features. .	3
1.3	The basic procedures for our classification system. Given a labeled dataset with K class labels, we first extract the low-level features from every image, then we build the tree-shaped hierarchy. Next, we exploit the semantic relations between object classes to construct the mid-level features, and then we learn a model to predict a class label for every image.	4
1.4	General overview of our two approaches. (bottom), we are given a tree-shaped hierarchy for object categories (the light blue line represent all possible comparison pairs between objects). We exploit this hierarchy to form the mid-level feature representation. In approach A, we represent the “Cat” image by differentiating it from all other objects in the tree-shaped hierarchy, where in approach B we only consider objects along its path.	6
2.1	Illustration of the standard BoW image representation.	8
2.2	An example of Cao et al. [6] image representation.	11

3.1	An overview of our approach A. Top: we are given a hierarchy of object categories. From this hierarchy, we construct L pairs of object categories and learn a binary classifier for each pair (details in Sec. 3.1.1). Bottom: for an image, we apply these L binary classifiers and treat the scores of these L binary classifiers as mid-level feature representation of this image. We then learn a non-linear classifier to recognize the image category corresponding to a leaf node in the hierarchy based on this mid-level image representation (details in Sec. 3.1.2).	15
3.2	Illustration of how to construct the scores at a single node (“animal”) in the hierarchy. We consider each pair between two children of this node, e.g. “cat” vs “dog”, “cat” vs “fox”, “dog” vs “fox”, etc. For each pair, we learn a binary classifier differentiating the two object categories. The outputs of these classifiers are the mid-level features constructed at this node.	16
3.3	Illustration of how to construct the mid-level feature from the entire hierarchy for an image. We apply the process illustrated in Fig. 3.2 to every internal node of the hierarchy. Each internal node will give a set of scores. The concatenation of scores across all internal nodes forms our mid-level feature representation.	17
3.4	An overview of approach B: (left) Given a tree-structured hierarchy, we construct a set of binary classifiers. Each dash blue line represents a classifier between two concepts in the hierarchy; (right) For a given image, we represent the image using a vector of mid-level features. The entries of this vector are the responses of the corresponding binary classifiers on this image.	20
3.5	For each training image, we form its mid-level representation by collecting scores from the set of binary classifiers L_i that associated to internal node V_i along its ground-truth path. Using these scores, we learn non-linear SVM to recognize to which class this image belongs to. (See Sec. 3.2.1 for more details).	21
3.6	To find a path for a testing image, a naïve approach traverses each of the K possible paths from the root to leaves in the search tree. For example, the naïve approach will explore all 8 paths in this hierarchy. The naïve approach gives reasonably good results (see Table. 4.2), but it is very computationally expensive since we need to repeatedly traverse paths in the hierarchy.	22
3.7	To find a path for a testing image, we prune many branches in the search tree. For example, our approach will explore 4 out of the 8 possible paths in this hierarchy. In contrast to the naïve approach (Fig. 3.6), our approach only chooses a subset of the children to visit at each internal node.	23

3.8	For each test image, we explore more than one path, and construct more than one mid-level representation. We feed these mid-level representations to the learned non-linear SVM (Sec. 3.2.1) which predicts the best path of the given image. Please refer to Sec. 3.2.2 for details.	24
3.9	An overview of building a tree-shaped hierarchy: (left) we recursively use k-medoid clustering to group class labels into k super-classes; (right) the resulting tree-shaped hierarchy.	25
4.1	Sample images of the ImageNet65 dataset.	28
4.2	Sub-set of the ImageNet hierarchy.	29
4.3	Sample images of the AwA dataset.	30
4.4	Sub-set of the AwA hierarchy.	31
4.5	Sample images of the CIFAR dataset.	32
4.6	Sub-set of the CIFAR hierarchy.	33
4.7	Sample images of the Yahoo Shoes dataset.	34
4.8	Sub-set of the Yahoo Shoes hierarchy.	35
4.9	Comparison of the average number of nodes visited by test images between approach B (Sec.3.2) and approach A (Sec(3.1) on the four datasets. Approach B visits significantly less nodes due to the pruning strategy.	37
4.10	Confusion matrices of approach A (Sec.3.1) on four datasets.	39
4.11	Confusion matrices of our approach B (Sec.3.2) on four datasets.	41
4.12	Some examples of our approach B predictions (Sec. 3.2) compared with single path method.	42
4.13	Some examples of our approaches predictions compared with baseline methods.	43

List of Tables

4.1	Results of overall accuracies for our approaches A and B (Sec. 3.1, and Sec. 3.2) compared with two baseline methods: raw features, and Cao et al. [6], on four datasets.	36
4.2	Overall accuracies for our approach B (Sec.3.2) compared with two baseline methods: single path method, and all paths method, on four datasets.	38
4.3	Comparison of the performance of our approaches using the predefined tree-shaped hierarchy (first two rows), and using the learned tree-shaped hierarchy (last two rows).	40

Acknowledgments

I would like to begin by thanking my adviser Prof. Wang Yang for his support and guidance during my Master years in University of Manitoba. I would also to thank my committee members. I also very thankful to my colleague Mrigank Rochan in Computer Vision Lab.

Also and foremost, my deepest thanks to my husband Dr. Asim Alsaedi for all the sacrifices that he did on behalf of me. A special thanks go beyond the seas to my parent as words can not express how grateful I am for all their supports.

Finally, I would like to thank King Abdullazia University, Saudi Arabia for their generous financial supports for me and my family.

To my family

Chapter 1

Introduction

Image classification is a core research area in computer vision. It is a task of assigning a label to a given image. It is a very challenging problem for computers due to the dramatic changes in size, position, illumination, and viewpoint of objects. The basic idea in image classification is to learn a model for classifying new images from a set of labeled training images. To perform the learning, images have to be transformed into representations that are understandable by machine, and then a model (called a “classifier”) has to be trained to predict the labels for these representations.

Most approaches in image classification use low-level feature representation, which can be extracted directly from images (e.g. color, texture, shape). For example, the most widely used approach is based on local features, or bag-of-words (BoW) representation. In this approach, local descriptors (e.g. SIFT [20]) are extracted from images. Then, the local descriptors from the training images are clustered to form the “visual words”. Each image is then represented by a BoW representation by counting the frequency of visual words in the image. Finally, a predictive model

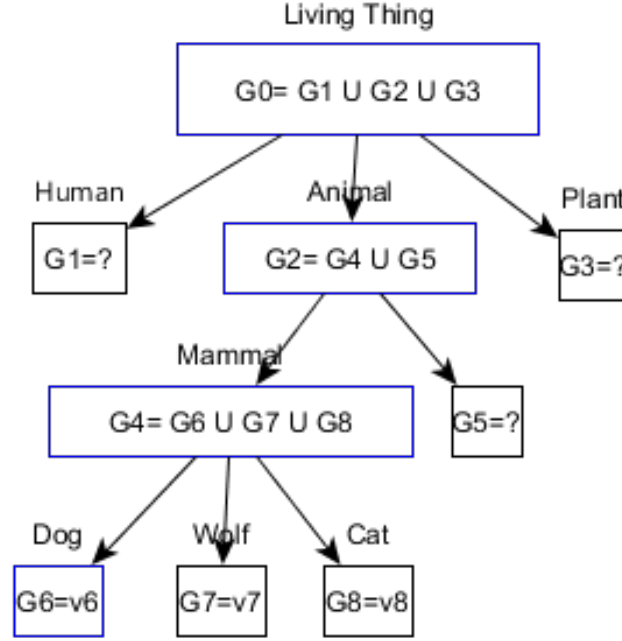


Figure 1.1: Illustration of the tree-shaped hierarchy. Every leaf node represents an individual class label; such as, v_6, v_7, v_8 . The root node “living thing” is the union of all leaf nodes’ class labels. Every internal node represents a subset of leaf nodes’ class labels; such as, internal node $v_4 \in V$ represent the group $G_4 = v_6, v_7, v_8$.

is learned based on this BoW representation of images.

Although BoW models have been very popular, the visual words used in these models usually have no explicit semantic meanings. So they fail to offer sufficient discriminative power. This is commonly known as the *semantic gap* [18] in visual recognition.

To address this issue, we introduce new semantically meaningful mid-level feature representations for visual recognition. In this thesis, we use the term “mid-level features” to mean features that cannot be extracted from images alone. Instead we need some kind of semantic knowledge to learn those features. We consider the

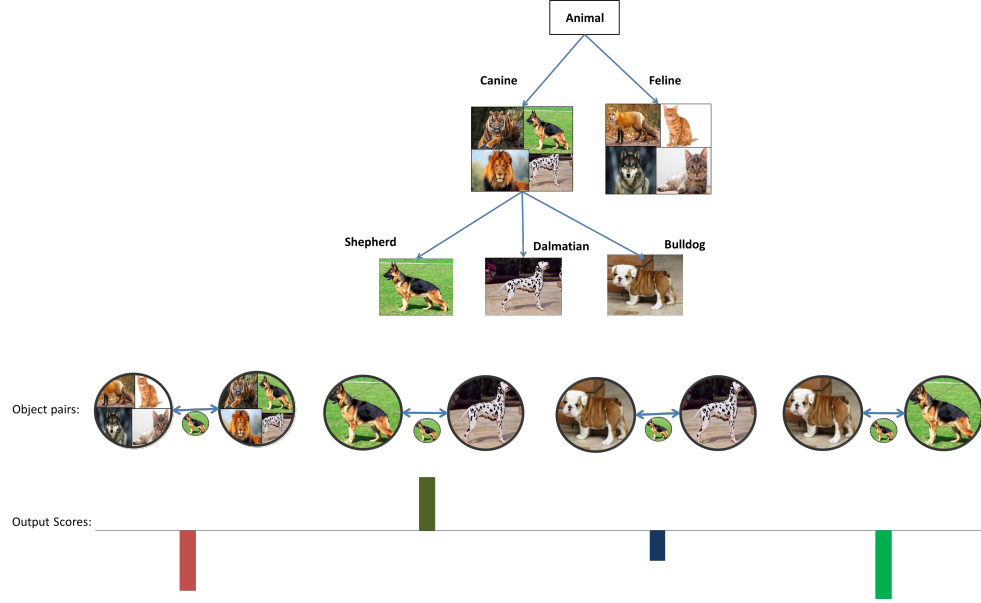


Figure 1.2: Intuitive example of our mid-level representation; (top) we have a subset of a hierarchy of object categories; (bottom) number of object pairs range from coarse pairs (e.g., “Canine” vs “Feline”), to fine-grained pairs: (e.g., “Shepard dog” vs “Dalmatian dog”). For the “Shepard dog” image, we use the output scores to form its mid-level features.

hierarchical structure of object categories as a resource for the semantic knowledge. We can define the hierarchical structure (see Fig. 1.1) as a tree $T = (V, E)$ with nodes V and edges E . The leaf nodes in the tree represent individual class labels, and each internal node $v \in V$ is associated with group of class labels l_v corresponding to all leaf nodes in the subtree rooted at v .

Object categories naturally have a hierarchical structure (i.e. taxonomy) with different levels of abstraction. For example, a path in the object hierarchy could be “living thing \rightarrow animal \rightarrow mammal \rightarrow dog”. In computer vision, object hierarchies have been used to organize images [19], provide fast run-time algorithms [4], enable novel applications [8]. However, there is little work on using object hierarchy to construct semantic features.

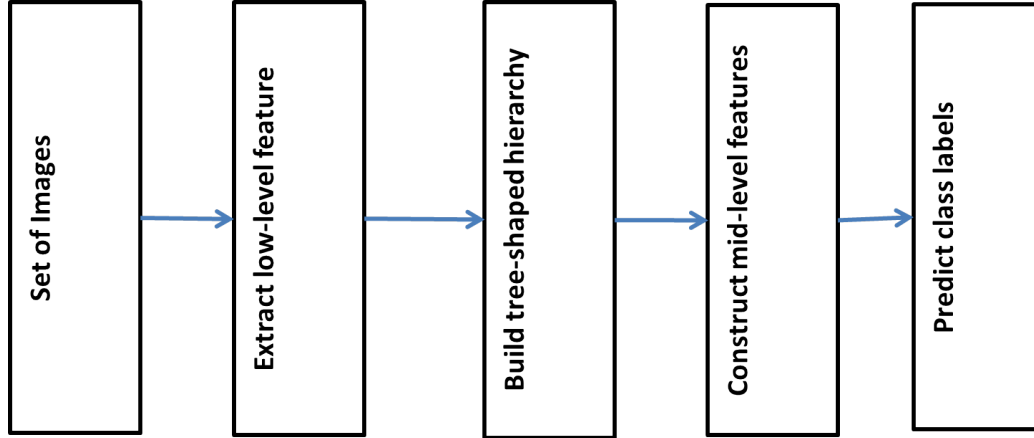


Figure 1.3: The basic procedures for our classification system. Given a labeled dataset with K class labels, we first extract the low-level features from every image, then we build the tree-shaped hierarchy. Next, we exploit the semantic relations between object classes to construct the mid-level features, and then we learn a model to predict a class label for every image.

In this thesis, we propose new approaches for constructing mid-level feature representations by exploiting the hierarchical structure of object categories. Our mid-level representation is based on a collection of output scores from a set of binary classifiers, which differentiate between pairs of object categories in the hierarchy. To give an intuitive example, suppose we have a sub-set of a hierarchy as in Fig. 1.2. We first learn a set of binary classifiers to differentiate between every pair of object categories which range from coarse object categories (e.g. “Canine” vs “Feline”) to fine-grained object categories (e.g. “Shepard dog” vs “Dalmatian dog”). Then, to represent the “Shepard dog” image with mid-level representation, we collect the output scores from these binary classifiers. The output scores are then used to form the mid-level features representation for the “Shepard dog” image.

Both of our approaches exploit the semantic hierarchy to define a collection of object pairs to extract the mid-level features. The main difference between them

lies in the way of unifying these mid-level features. In the first approach (approach A), we construct the mid-level representation for an image by concatenating all output scores from all binary classifiers which differentiate between all object pairs in the hierarchy. In the second approach (approach B), we limit our attention to construct the mid-level representation by concatenating the output scores from binary classifiers of object pairs that produce meaningful discriminative scores. Meaningful discriminative scores can be acquired by considering only object pairs along the path of the given image. An overview of our two approaches is illustrated in Fig.1.4. Both of our approaches require a tree-shaped hierarchy of object classes. In many cases, this hierarchy is given, e.g. based on the WordNet hierarchy. However, to propose a complete classification system, we proposed a method for constructing the hierarchy from data when the hierarchy is not readily available. The basic procedures for our classification system are shown in Fig.1.3.

Our work is motivated by the observation that visual features might have different discriminative power at various levels in the hierarchy. For example, certain features might be useful for differentiating high-level abstract categories (e.g. “animal” vs “plant”), while others are useful for more fine-grained object categories (e.g. Shepard dog vs Eskimo dog). By constructing object pairs at different levels of the hierarchy, we are able to learn a diverse set of discriminative mid-level features.

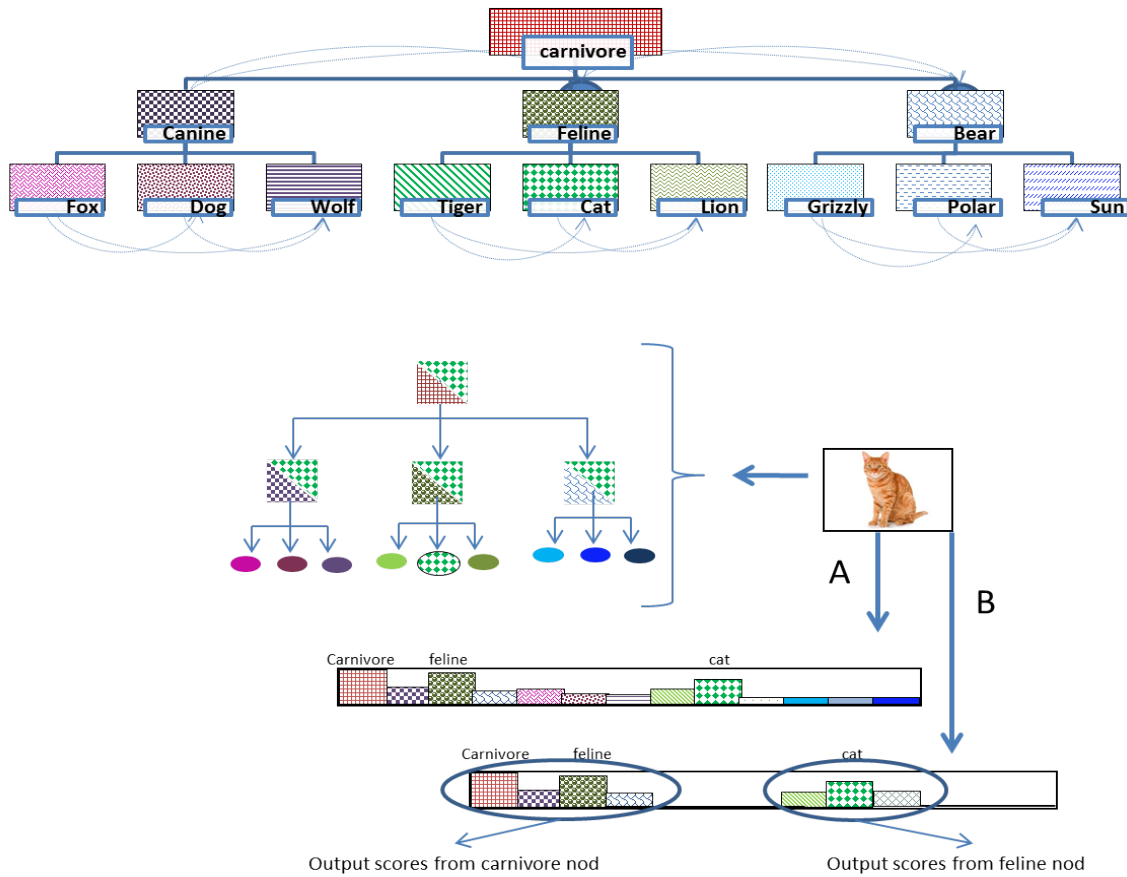


Figure 1.4: General overview of our two approaches. (bottom), we are given a tree-shaped hierarchy for object categories (the light blue line represent all possible comparison pairs between objects). We exploit this hierarchy to form the mid-level feature representation. In approach A, we represent the “Cat” image by differentiating it from all other objects in the tree-shaped hierarchy, where in approach B we only consider objects along its path.

Chapter 2

Related work

Image classification is a classic problem in computer vision. A comprehensive review of the image classification literature is beyond the scope of this thesis. Here we only go over the works most related to our proposed method.

2.1 Image classification/object recognition

The most popular approach for image classification is the bag-of-words method(BoW). It involves extracting local descriptors from interest points found in an image (e.g. SIFT descriptors [20]), quantizing the descriptors into visual words by clustering the descriptors, and counting the occurrences of each visual word to construct the histogram of word frequencies for each image (see Fig.2.1). The histogram representation is then used as the feature vector of the whole image and fed to a classifier. The BoW representation ignores the spatial layout of local descriptors.

Spatial Pyramid Matching (SPM) was proposed by Lazebnik et al [17] to overcome

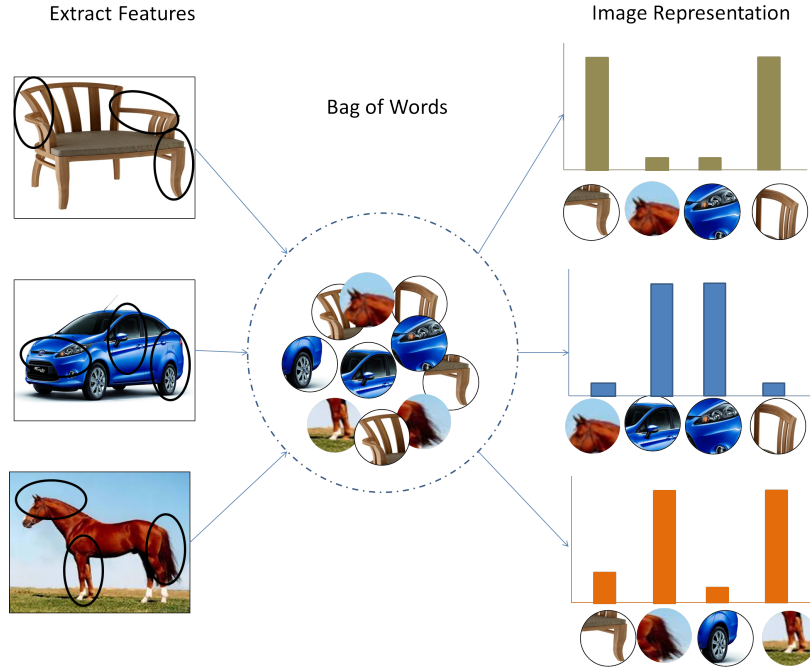


Figure 2.1: Illustration of the standard BoW image representation.

the limitation of BoW model by capturing the spatial information of local descriptors. SPM divides an image into several regions at different spatial resolutions and represents each region using BoW. The whole image is represented by the concatenation of BoW representations for all subdivided regions. SPM is considered as one of the most effective image classification techniques.

2.2 Large scale image classification

Datasets are crucial for the development of image classification systems. In the computer vision community, a lot of efforts have been made in collecting and annotating large-scale image datasets. One representative example is the ImageNet dataset [9], which organized according to the WordNet hierarchy, and consists of more than

5247 object categories (also called “synset” in ImageNet) and more than 3.2 million images. Images in each synset are obtained by querying the web. The results are human verified. WordNet [21] is one of the most popular semantic networks for English language. It groups words into sets of synonyms and define different semantic relations between them. For example, two connected nodes in the hierarchy indicate the “is-a” relationship between them. For example, a “dog” is a “mammal”, an “animal”, and a “living thing”.

Deng et al. [8] conducted the first experimental study of image categorization on large scale ImageNet dataset. They showed that as the number of categories increases, accuracies of standard classifiers decrease. Also, they showed that classification based on hierarchical cost is significantly more informative when working on large scale dataset. For example, classifying a dog image as “cat” should incur a lower cost than classifying it as “helicopter”, since “cat” is closer to “dog” in the object hierarchy than “helicopter”.

Lin et al. [19] were motivated by the fact that existing algorithms consume a significant amount of time when dealing with large dataset of about 1.2 million images. Thus, they worked to enhance the efficiency of large scale image classification by developing a fast feature extraction and fast classifier training method. For feature extraction, they achieved efficient performance by allowing data to be stored in a distributed way using Hadoop (an open-source software framework that supports data-intensive distributed application). For classification, they achieved fast SVM training using parallel averaging stochastic gradient descent (ASGD) where classifiers can be learned in parallel and the training data are shared through careful memory

sharing.

2.3 Hierarchy

There are many approaches in image classification that focus on proposing faster and memory efficient classifiers for large dataset. In particular, Bengio et al. [4] proposed the label tree model to reduce the complexity of classifying larger scale dataset. The label tree consists of internal nodes and leaf nodes, where each leaf node corresponds to a single class label, and each internal node is associated with a linear classifier. The test image traverses the tree from the root to a leaf. At each internal node, the linear classifier associated with that node determines which child to visit. The label tree method is efficient because the run-time in testing is logarithmic to the number of object categories.

Deng et al. [11] developed an improved method by learning the hierarchy jointly with the classification model. Gao et al. [12] further improved the method by allowing overlapping object classes at different child nodes.

Object hierarchy has also been used to improve image retrieval [7] and to provide accuracy-specificity trade-offs in large scale recognition [10], where a testing image can be recognized at various levels in the tree with different confidence. For example if the main object in the testing image is a smart car, the system might predict it as a car with 90% confidence, a vehicle with 95% confidence, etc.

To achieve better trade-off between efficiency and accuracy with the object hierarchy, Sun et al. [25] proposed to use the branch-and-bound technique for efficient classification. The test image in the standard tree-based algorithm traverses only

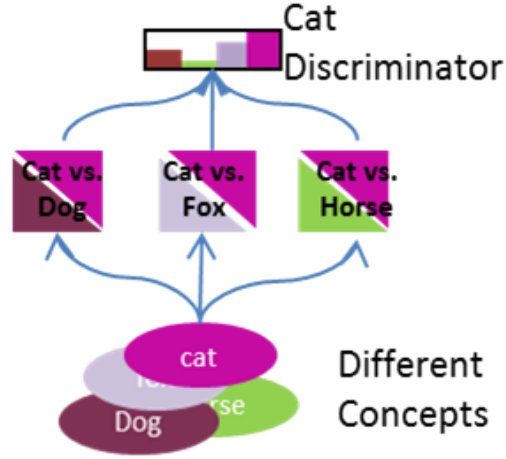


Figure 2.2: An example of Cao et al. [6] image representation.

single path in the tree from the root to a leaf based on the linear classifiers associated with every internal node. This implies that if a mistake occurs at one node, it will be propagated to a leaf node and cannot be recovered. Sun et al. [25] overcame this limitation by exploring more than one path in the tree-shaped hierarchy and typically finish in sublinear time.

2.4 Mid-level features

Most work in image classification use low-level visual features. These features do not have much high-level semantic information about the object categories. In order to address this limitation of low-level features, researchers have been developing mid-level features that encode semantic information about the objects. Two representative examples are Cao et al. [6] and Li et al. [18]. Cao et al. [6] proposed a learning-by-focusing method. Given a dataset with many classes, this approach learns a set

of one-vs-one classifiers between every pair of object classes. The responses of these classifiers are then used as features for recognition. For example, to build a cat classifier (Fig.2.2), they consider number of concept pair; such as, “cat” vs “dog”, “cat” vs “horse”, “dog” vs “horse”, etc. For each pair, they learn a binary classifier differentiating the two object categories. The outputs of these classifiers are the discriminator for the cat object. Such mid-level representation had enhanced the performance of image classification systems. Li et al. [18] proposed a representation called “Object Bank” for scene recognition. This representation first learns a large collection of object detectors. For a given image, these detectors are applied and their responses are used as mid-level features for recognition. This image representation has been shown to be effective for high-level vision tasks, such as scene classification.

An extension of object bank, called *action bank* [23], is proposed to represent complex activities in videos. Torresani et al. [26] developed a similar representation called *classme* for object classification. Classme first learns classifiers for a set of basis classes. Any new object category is then represented as combinations of these basis classes.

Chapter 3

Our approach

Most image classification methods use low-level features, which do not capture much semantic information about the object categories. Cao et al. [6] and Li et al. [18] use mid-level features, but do not consider the valuable semantic knowledge on object hierarchy. We propose two approaches to show how a taxonomy could be exploited to help object category recognition.

Given an input image belonging to a leaf node of a hierarchy, we first represent the image as a vector of standard low-level features (e.g., color, texture, etc). We then apply a large number of binary classifiers on this low-level feature vector. Each classifier will output a score. In approach A (Sec.3.1) , we concatenate the scores of all binary classifiers from all the internal nodes in the hierarchy to form a mid-level feature representation of the image. In approach B (Sec.3.2), we propose an alternative way to form a mid-level feature representation of the image by concatenating the scores of binary classifiers that are associated with the most “relevant” internal nodes in the hierarchy. Our approaches require a tree-shaped hierarchy of object classes.

In many cases, this hierarchy is given, e.g. based on the WordNet hierarchy [9]. In Sec.3.3, we describe how to construct the hierarchy from data when the hierarchy is not readily available.

3.1 Approach A

An overview of our approach A, which has been published as [2], is illustrated in Fig. 3.1. Given an input image, we first represent the image as a vector of standard low-level features (e.g. color, texture, etc). We then apply a large number of binary classifiers on this low-level feature vector. Each classifier will output a score. We concatenate the scores of all binary classifiers to form a mid-level feature representation of this image. We then apply a non-linear classifier on this mid-level representation to predict the class label.

Our image representation is constructed from the responses (i.e. scores) of many binary classifiers. Similar to Cao et al. [6], we learn each classifier to differentiate one pair of semantically exclusive concepts ¹ (one-vs-one). An alternative is to learn a classifier that differentiates between one concept from all other concepts (one-vs-all). As demonstrated by Cao et al. [6], the former (one-vs-one) is preferable, since it is easier for the learning algorithm to find the discriminative features that distinguish two concepts. The latter (one-vs-all) is arguably more challenging for the learning algorithm, since it has to learn features that distinguish a concept from a diverse set of other concepts. In our work, we choose the one-vs-one strategy.

¹In this thesis, “concept” and “object class” are used interchangeably.

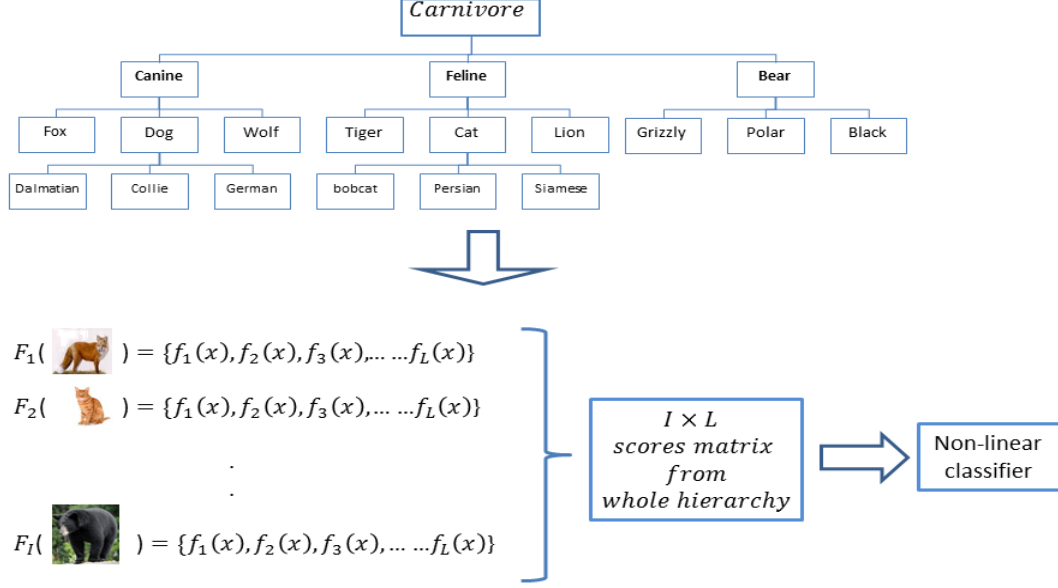


Figure 3.1: An overview of our approach A. Top: we are given a hierarchy of object categories. From this hierarchy, we construct L pairs of object categories and learn a binary classifier for each pair (details in Sec. 3.1.1). Bottom: for an image, we apply these L binary classifiers and treat the scores of these L binary classifiers as mid-level feature representation of this image. We then learn a non-linear classifier to recognize the image category corresponding to a leaf node in the hierarchy based on this mid-level image representation (details in Sec. 3.1.2).

3.1.1 Selecting concept pairs

Suppose we have a set of K concepts $\mathcal{C} = \{C_1, C_2, \dots, C_K\}$. These concepts correspond to the leaf nodes in the object hierarchy. The method in [6] constructs $\frac{K(K-1)}{2}$ concept pairs by considering every possible combination of two concepts. For each concept pair (e.g. “bicycle” vs “bus”), it then learns a linear SVM classifier that differentiates these two concepts.

We propose a new way of constructing concept pairs by exploiting the semantic

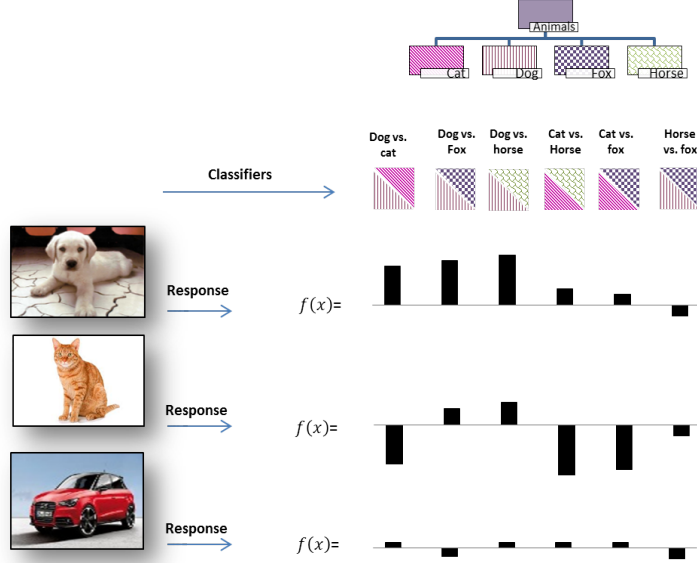


Figure 3.2: Illustration of how to construct the scores at a single node (“animal”) in the hierarchy. We consider each pair between two children of this node, e.g. “cat” vs “dog”, “cat” vs “fox”, “dog” vs “fox”, etc. For each pair, we learn a binary classifier differentiating the two object categories. The outputs of these classifiers are the mid-level features constructed at this node.

hierarchical structure of object categories. Let us consider a non-leaf node V in the hierarchy, e.g. V might correspond to the concept “animal”. Suppose the node V has t child nodes. The child nodes of “animal” might correspond to concepts such as “dog”, “cat”, “horse”, etc. For the node V , we construct $\frac{t(t-1)}{2}$ concept pairs by choosing all pairs of concepts from the child nodes of V . For example, the concept pairs for “animal” will include “dog” vs “cat”, “dog” vs “horse”, “cat” vs “horse”, etc.

We repeat the process for all internal nodes in the hierarchy. In the end, we

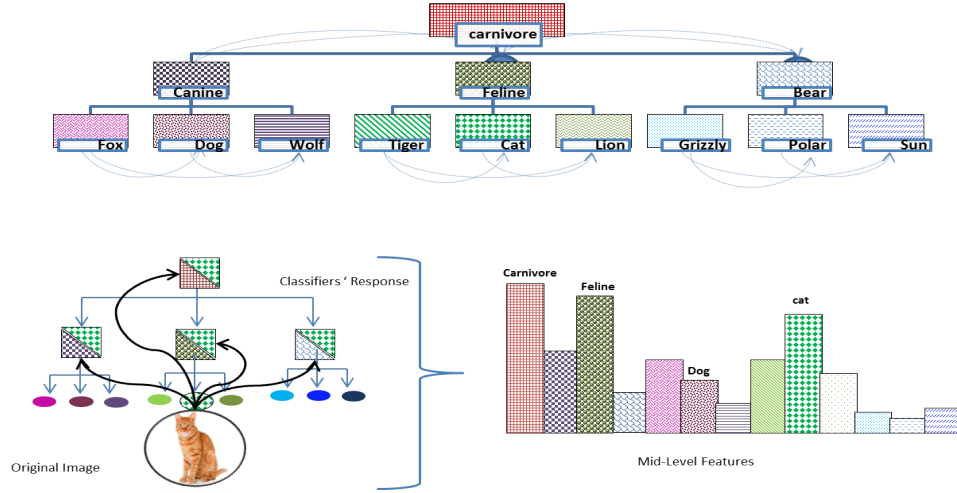


Figure 3.3: Illustration of how to construct the mid-level feature from the entire hierarchy for an image. We apply the process illustrated in Fig. 3.2 to every internal node of the hierarchy. Each internal node will give a set of scores. The concatenation of scores across all internal nodes forms our mid-level feature representation.

will get a collection of concept pairs. Some of the concept pairs will correspond to abstract concepts, such as “animal” vs “plant”. Others will correspond to fine-grained concepts, such as “Shepard dog” vs “Eskimo dog”.

We then use standard techniques to learn a classifier to differentiate the two concepts in each concept pair. We first extract some standard low-level image features from the images, such as color histogram, SIFT histogram [20], GIST [22], LBP [1], etc. We will describe in detail the low-level features we used on each dataset in the experiment later (Sec. 4.1). For a concept (e.g. “dog”) in the hierarchy, its training examples are those labeled as descendants of this concept. In other words,

all the images that are labeled as specific species of dogs in the training data will be considered as “dog” images. We then learn a binary linear SVM classifier for each concept pair. We represent the classifier for the i -th concept pair as f_i . Given a new input image \mathbf{x} , this classifier will return a score $f_i(\mathbf{x})$. We can interpret this score as the confidence of differentiating \mathbf{x} between the positive/negative class in the i -th concept pair. If f_i corresponds to the “dog” vs “cat” concept pair with the “dog” being the positive class, we would expect $f_i(\mathbf{x})$ to be high if \mathbf{x} is an image of a dog. Similarly, we would expect a low score if it is an image of a cat, and a score close to zero if it is neither a dog or a cat image. Figs. 3.2 and Fig.3.3 illustrate the whole process.

3.1.2 Image representation

The method in Sec. 3.1.1 gives us a collection of L binary classifiers f_1, f_2, \dots, f_L . We can interpret these binary classifiers as defining a L -dimensional semantic concept space. Due to the way we construct these binary classifiers, this concept space encodes information about the hierarchical structure of object categories.

For an image \mathbf{x} , we encode this image as a L -dimensional vector $F(\mathbf{x})$ by concatenating the scores of these binary classifiers applied on \mathbf{x} , i.e. $F(\mathbf{x}) = [f_1(\mathbf{x}), f_1(\mathbf{x}), \dots, f_L(\mathbf{x})]$. We treat $F(\mathbf{x})$ as the mid-level feature representation of the image \mathbf{x} . Using this L -dimensional feature representation on training data, we then learn a K -class non-linear SVM classifier to predict the class label of any given image.

During testing, we are given an unseen image \mathbf{x} . Similarly, we encode \mathbf{x} using the L binary classifiers as $F(\mathbf{x}) = [f_1(\mathbf{x}), f_1(\mathbf{x}), \dots, f_L(\mathbf{x})]$. We then apply the learned

K -class non-linear SVM to predict the class label of this new image.

3.2 Approach B

In this section we propose an alternative approach which better exploits the hierarchical structure. The approach A (Sec. 3.1) has to evaluate all the binary linear classifiers on an image. But intuitively, only a subset of these classifiers will produce meaningful scores on a given image. As an example, let us consider an image of “shepard dog”. If a concept pair (e.g. “apple” vs “banana”) is irrelevant to “shepard dog”, the score of the corresponding linear classifier will likely to be close to 0. This suggests that we only need to evaluate a subset of the binary classifiers and approximate the scores of the remaining ones with 0.

An overview of our approach B is shown in Fig. 3.4. For a given image, we first extract standard low-level visual features (e.g. color, texture, shape, etc). We then apply a large collection of binary classifiers on the low-level features. The responses of the most “relevant” binary classifiers are used to construct a mid-level image representation. The “relevant” binary classifiers correspond to the concept pairs of every internal node V along the path of the image.

3.2.1 Image representation using object hierarchy

In this approach, we use the same process in approach A (Sec.3.1.1) to learn the collection of binary classifiers, since it allows exploiting the hierarchical structure of object classes.

For a new image, we construct its representation using the output scores of these

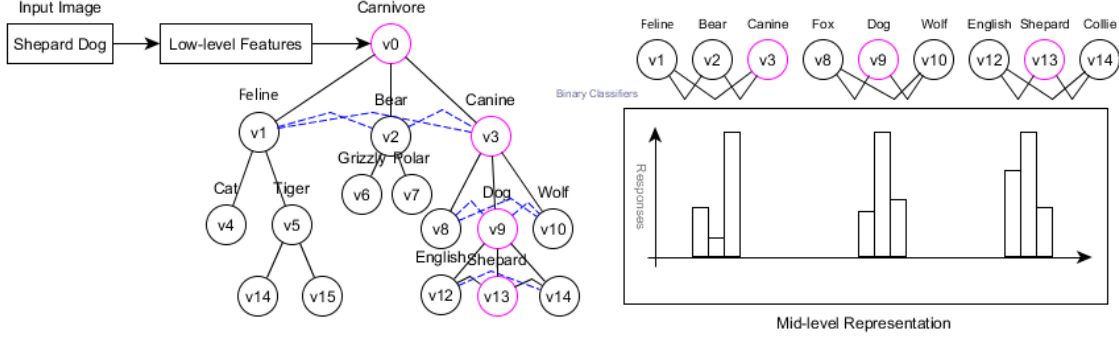


Figure 3.4: An overview of approach B: (left) Given a tree-structured hierarchy, we construct a set of binary classifiers. Each dash blue line represents a classifier between two concepts in the hierarchy; (right) For a given image, we represent the image using a vector of mid-level features. The entries of this vector are the responses of the corresponding binary classifiers on this image.

linear classifiers. However, in contrast to approach A (Sec. 3.1.2), we propose to construct the image representation by only considering the most relevant concept pairs. We first describe how to construct the image representation for training images, for which we know the ground-truth labels. In Sec. 3.2.2, we will explain how to handle test images for which the ground-truth labels are unknown.

Given a training image, since we know its ground-truth label, we can find the path (from the root to a leaf) of its object class in the hierarchy. For each internal node V along the path, we consider the concept pairs between each pair of its children to be “relevant”. Our intuition is that these concept pairs are most likely to provide discriminative information for this path. For example, let us consider a training image of “collie dog” in the hierarchy in Fig. 3.5. We first find the ground-truth path (from the root to a leaf node) corresponding to “collie dog”. In this case, the path is “ $v_0 \rightarrow v_1 \rightarrow v_5 \rightarrow v_{13} \rightarrow v_{21}$ ”. The relevant concept pairs at v_0 are $(v_1, v_2), (v_1, v_3), (v_2, v_3)$. Similarly, the relevant concept pairs at v_1 are $(v_4, v_5), (v_4, v_6), (v_6, v_5)$.

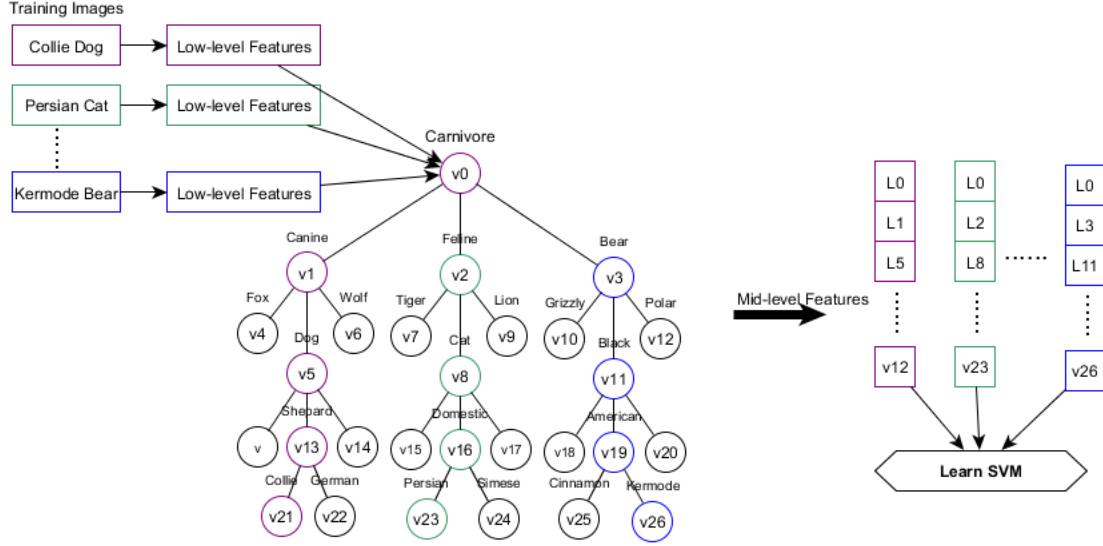


Figure 3.5: For each training image, we form its mid-level representation by collecting scores from the set of binary classifiers L_i that associated to internal node V_i along its ground-truth path. Using these scores, we learn non-linear SVM to recognize to which class this image belongs to. (See Sec. 3.2.1 for more details).

The final image representation is a vector of SVM scores. An entry of this vector is nonzero only when its corresponding concept pair is “relevant”. Note that the number of relevant concept pairs can be different for different object classes. But the length of the image representation is identical for all classes. If the total number of linear SVMs (from Sec. 3.1.1) is M , the length of this vector is M . This vector is also sparse, since a lot of the entries correspond to irrelevant concept pairs and will be set to zero.

In the end, each training image is represented as a M -dimensional sparse vector. We then learn a non-linear multi-class SVM to classify the image into one of the K classes. Given the image representation, we can also use this non-linear classifier to obtain the score of predicting each of the K classes.

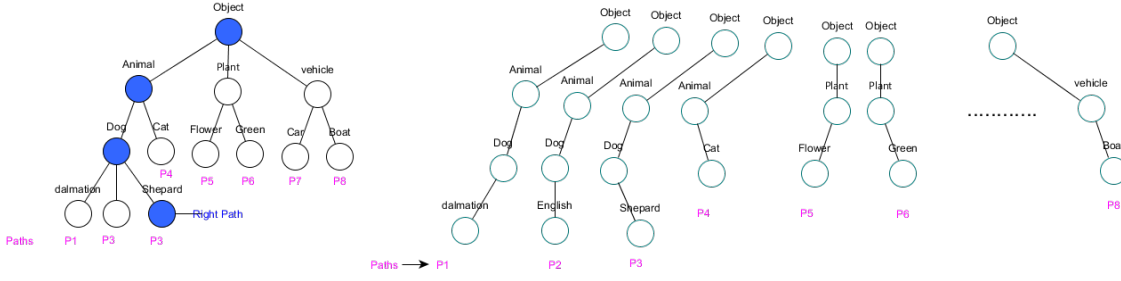


Figure 3.6: To find a path for a testing image, a naïve approach traverses each of the K possible paths from the root to leaves in the search tree. For example, the naïve approach will explore all 8 paths in this hierarchy. The naïve approach gives reasonably good results (see Table. 4.2), but it is very computationally expensive since we need to repeatedly traverse paths in the hierarchy.

3.2.2 Prediction on unseen images

For an unseen image during testing, we cannot directly construct the image representation in Sec. 3.2.1 since we do not know its ground-truth path in the hierarchy. A naïve approach is to traverse each of the K possible paths from the root to leaves. For the i -th path, we can construct the image representation using the method in Sec. 3.2.1. We then use the K -class non-linear classifier to obtain a score of predicting the i -th class. After traversing all paths, we will have the score for each of the K classes. In the experiments (Sec. 4.2.1), we will show that this naïve approach gives reasonably good results. But the limitation of this approach is that it is very computationally expensive, since we need to repeatedly traverse paths in the hierarchy. The naïve approach (Fig.3.6) requires traversing the hierarchy starting from the root. At each internal node, it recursively visits all of the children of this node in a depth-first search manner. This procedure is repeated recursively until all nodes in the hierarchy are processed. In contrast, our approach (Fig.3.7) only chooses a subset of the children to visit at each internal node. In other words, we effectively prune

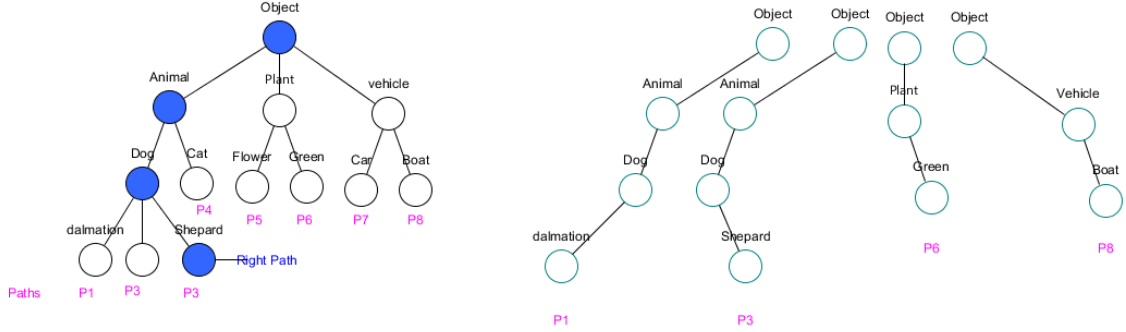


Figure 3.7: To find a path for a testing image, we prune many branches in the search tree. For example, our approach will explore 4 out of the 8 possible paths in this hierarchy. In contrast to the naïve approach (Fig. 3.6), our approach only chooses a subset of the children to visit at each internal node.

many branches in the search tree.

Let x be an unseen image, v be an internal node with n children $\{c_1, c_2, \dots, c_n\}$. Sec. 3.1.1 gives us a binary linear SVM classifier between each pair c_i and c_j ($i, j \in \{1, 2, \dots, n\}$). Suppose we consider c_i to be the positive class and c_j to be the negative class. We use $f_{ij}(x)$ to denote the score of this classifier on the image x . Note that $f_{ij}(x) = -f_{ji}(x)$ for any i and j . Using these binary classifiers, we first define the score of picking c_i as a child to visit: $h_i = \sum_{j: j \in \{1, 2, \dots, n\}, j \neq i} f_{ij}(x)$. We will visit the child c_i only when h_i is greater than a certain threshold T . In our experiment, we choose the threshold as the median of these scores, i.e. $T = \text{median}_{i \in \{1, 2, \dots, n\}} h_i$. The same procedure is iteratively applied to all child nodes. In the end, we would have traversed a subset of the K possible paths in the hierarchy. We use the K -class non-linear SVM (see Sec. 3.2.1) to obtain a final score for each traversed path. The path with the maximum score will give us the final prediction. For example (see Fig. 3.8),

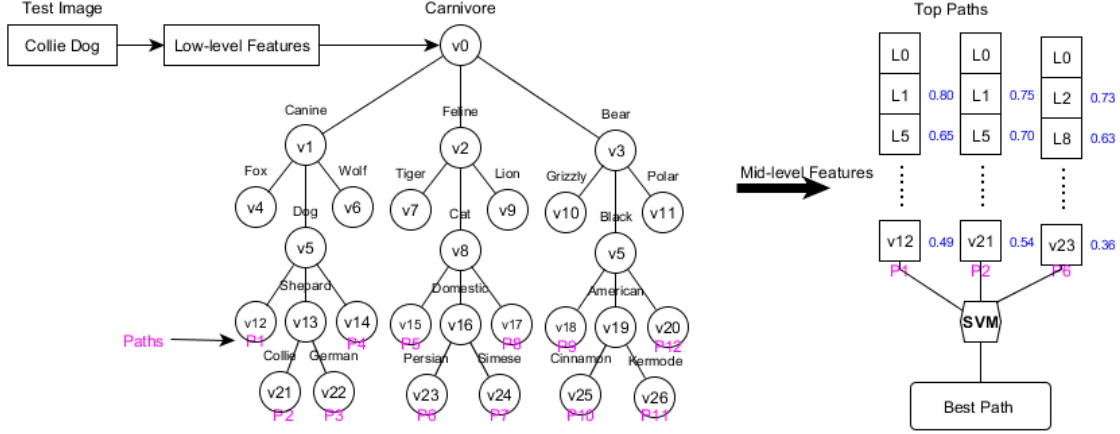


Figure 3.8: For each test image, we explore more than one path, and construct more than one mid-level representation. We feed these mid-level representations to the learned non-linear SVM (Sec. 3.2.1) which predicts the best path of the given image. Please refer to Sec. 3.2.2 for details.

suppose we have a test image “Collie dog”. Starting from root node $v_0 = \text{“carnivore”}$, we collect the output scores L_0 from the corresponding binary classifiers (“Canine” vs “Feline”, “Canine” vs “Bear”, “Feline” vs “Bear”). Suppose the scores of visiting “Canine” and “Feline” are greater than the threshold T . We will then prune “Bear” and only visit “Canine” and “Feline” at the next level in the hierarchy. We repeat this process. At each visited internal node v_i , we collected scores L_i from its binary classifiers, and pick some children to visit until leaf nodes are reached. At the end, we will have explored more than one path in the hierarchy. Each path will result in a mid-level representation. Then, we feed this representation to the learned non-linear SVM (Sec. 3.2.1) which gives a score for predicting the class k . After traversing several paths, we will pick the class with the best score from the non-linear SVM as the best predicted class label for the given image.

3.3 Building the tree-shaped hierarchy

Our methods are based on a tree-shaped hierarchy. In many applications, this hierarchy of object categories is given, e.g. based on the WordNet hierarchy [9]. In this section, we describe how to construct the hierarchy from data when it is not readily available.

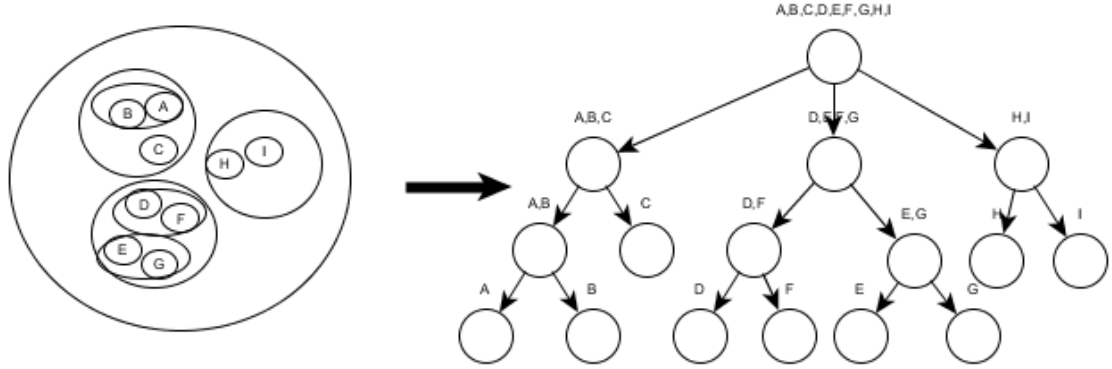


Figure 3.9: An overview of building a tree-shaped hierarchy: (left) we recursively use k -medoid clustering to group class labels into k super-classes; (right) the resulting tree-shaped hierarchy.

Given a labeled dataset with K class labels, we first learn a regular multi-class SVM with K classes based on the low-level image features. Using a separate validation set, we can get a confusion matrix $C \in R^{K \times K}$ for this multi-class SVM. The confusion matrix C_{ij} counts the number of images from class i which are misclassified as belonging to class j .

Based on the confusion matrix between classes, we group the class labels so that classes in a group are similar and classes in different groups are dissimilar. A standard k -medoids clustering algorithm [13] is applied to group the class labels into k super-classes. We set $(k \approx \sqrt[3]{k/2} \approx k/2)$, where k is the number of class labels [14]. This

process is then recursively applied to group the super-classes, until all class labels are group together in a single cluster. The result of this hierarchical clustering will give us a tree-shaped hierarchy (see Fig. 3.9).

Chapter 4

Experiments

Our experiments demonstrate the performance of our proposed approaches on several image classification datasets.

4.1 Datasets

We evaluate our proposed approaches on four publicly available datasets. On each dataset, we randomly choose 90% of the examples for training and use the remaining ones for testing. For simplicity, we use the pre-computed low-level features that come with some of the datasets. But it is important to note that our proposed approaches can be used together with any low-level features.

ImageNet65:

ImageNet [9] is a large scale dataset 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. The categories have been organized into a semantic

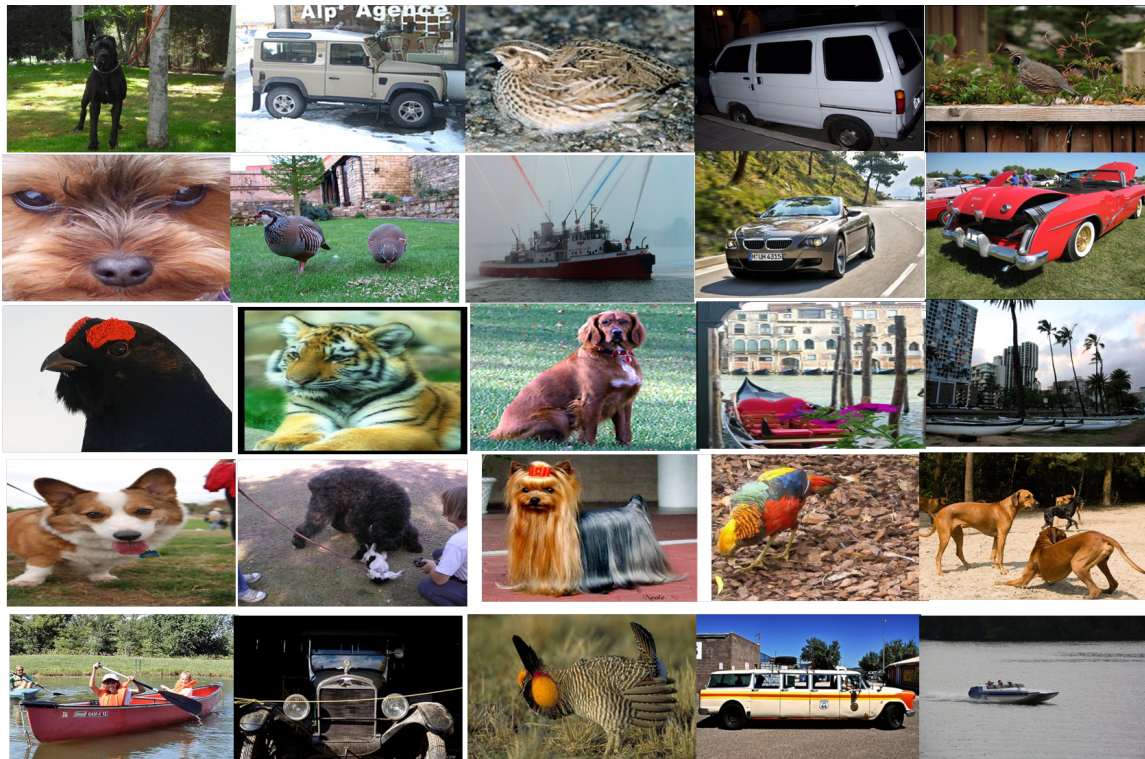


Figure 4.1: Sample images of the ImageNet65 dataset.

hierarchy based on the “is a” relation provided by WordNet .

ImageNet65 dataset is a subset of the ImageNet [9]. It contains 39600 images of 65 leaf nodes from the “plant” “animal” and “vehicle” subtrees in ImageNet. Some sample images from ImageNet65 are shown in Fig.4.1, and a subset of its hierarchy is shown in Fig.4.2.

Animal-with-Attributes (AwA) [16]:

This dataset consists of 30475 images of 50 different animal classes. This dataset comes with pre-extracted features which include six different feature types: RGB color histograms, SIFT [20], rgSIFT [27], PHOG [5], SURF [3] and local self-similarity histograms [24]. Some sample images from AwA are shown in Fig.4.3, and a subset

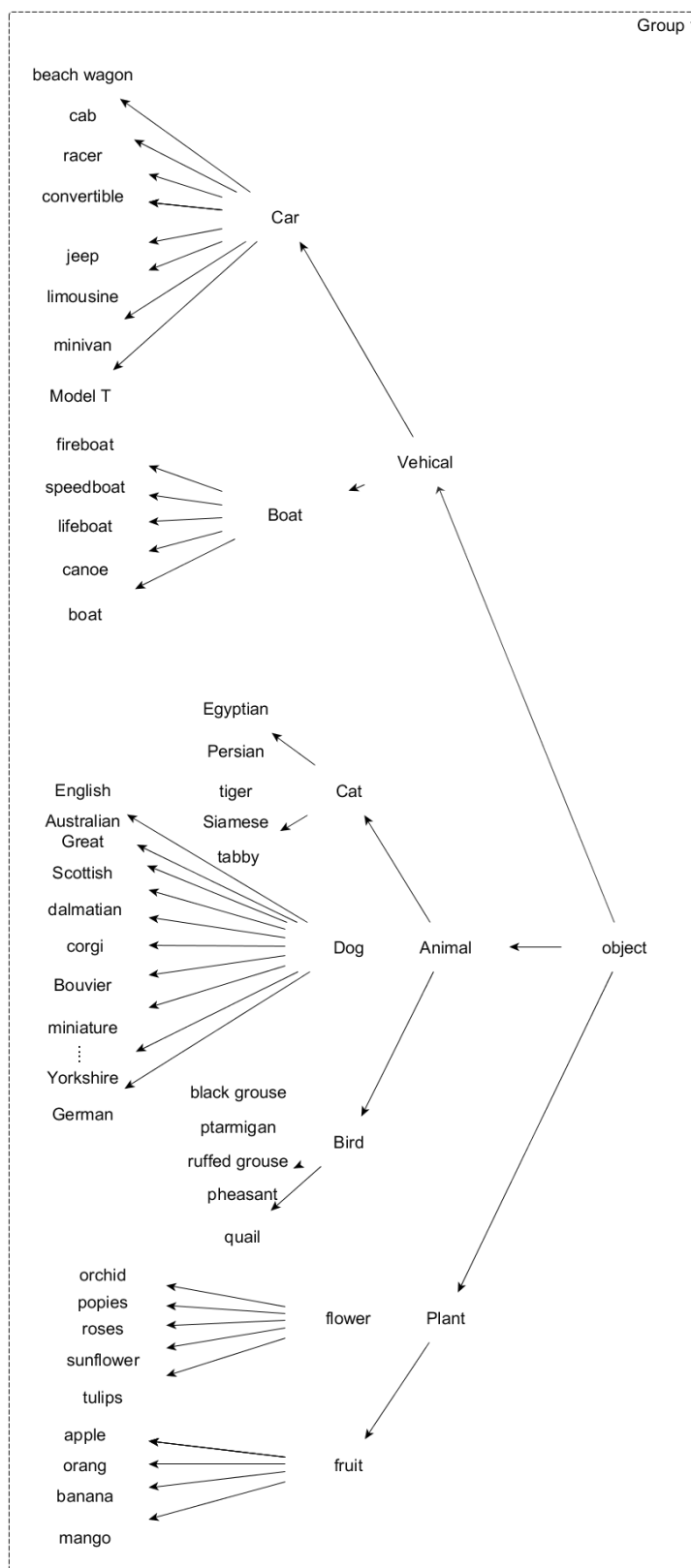


Figure 4.2: Sub-set of the ImageNet hierarchy.



Figure 4.3: Sample images of the AWA dataset.

of its hierarchy is shown in Fig.4.4.

CIFAR [15]:

This dataset consists of 60000 images of 100 object classes. Each object class belongs to one of the 20 *superclasses*. For example, the “fish” superclass contains 5 object classes: aquarium fish, flatfish, ray, shark, and trout. We build a two-layer hierarchy according to this superclass relation. In other words, the hierarchy has 20 internal nodes corresponding to the superclasses and 100 leaf nodes corresponding to the object classes. Several features have been extracted, such as, SIFT histogram [20], color histogram [28], and Local Binary Pattern [1]. Some sample images from CIFAR are shown in Fig.4.5, and a subset of its hierarchy is shown in Fig.4.6.

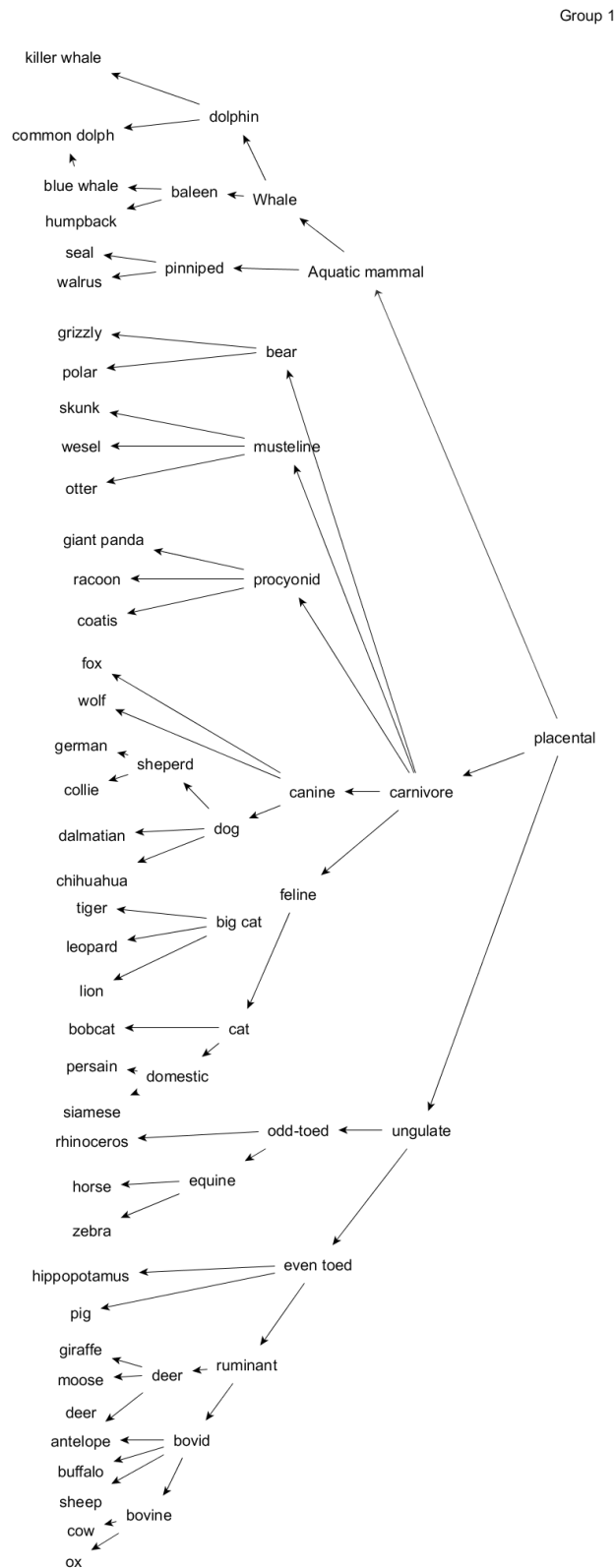


Figure 4.4: Sub-set of the AWA hierarchy.



This dataset consists of 5250 images of 107 shoe classes. Each shoe class belongs to one of the 10 *superclasses*. For example, the “Boots ” superclass contains 13 shoe classes: Ariat-Westren boots, Boges-Rain boots, Timber-Land boots , Justin-Western boots, etc. We build a two-layer hierarchy according to this superclass relation. In other words, the hierarchy has 10 internal nodes corresponding to the superclasses and 107 leaf nodes corresponding to the shoes classes. We extract several features for representing both local and global features including SIFT histogram [20], color histogram [28], Local Binary Pattern [1], and GIST [22]. Some sample images from

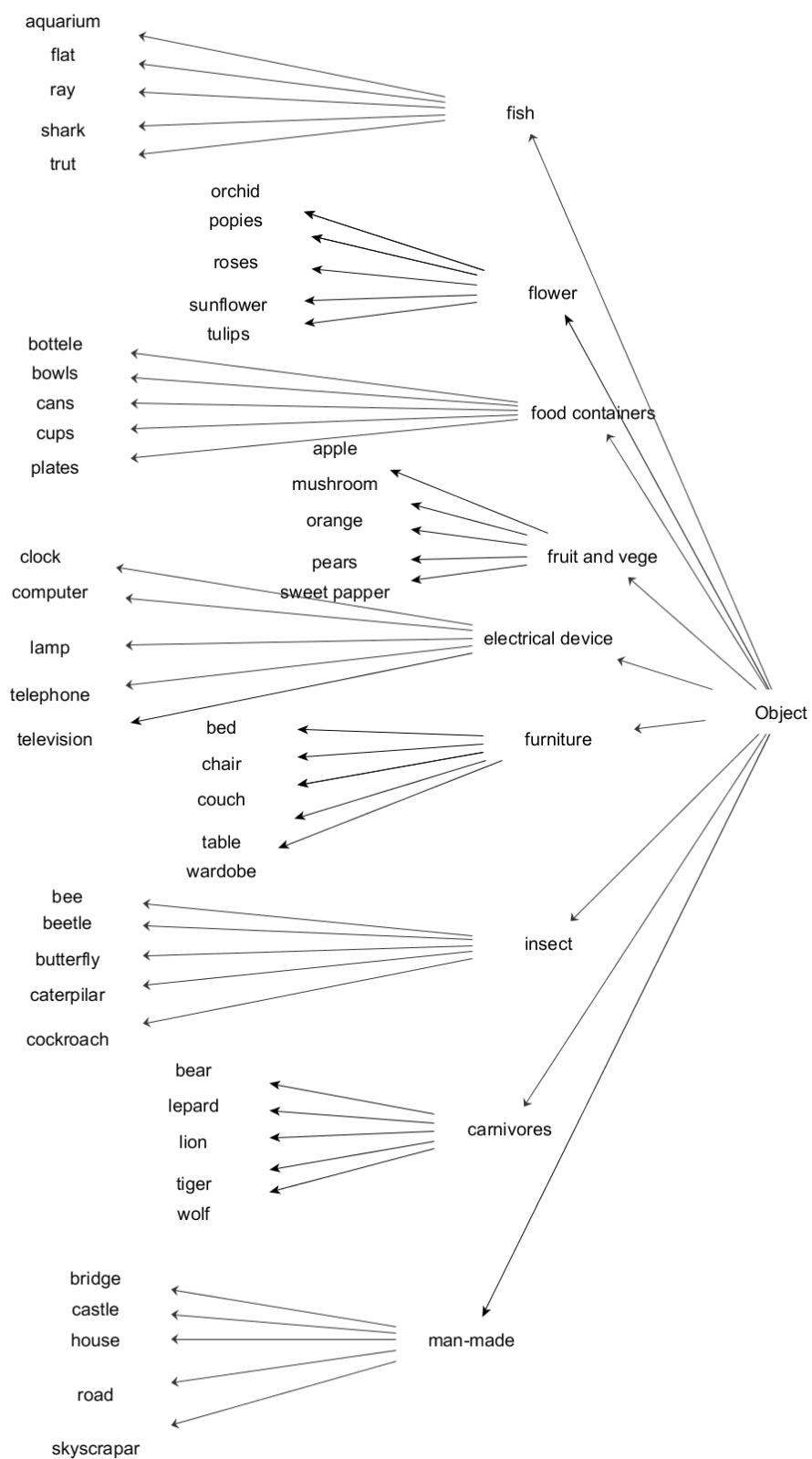


Figure 4.6: Sub-set of the CIFAR hierarchy.



Figure 4.7: Sample images of the Yahoo Shoes dataset.

Yahoo shoes are shown in Fig.4.7, and a subset of its hierarchy is shown in Fig.4.8.

4.2 Evaluation methodology

Each dataset in Sec. 4.1 has a hierarchy. This hierarchy is constructed based on the WordNet (e.g. ImageNet65 dataset [9]) or constructed based on the additional information of the dataset (e.g. Yahoo shoe dataset). We first evaluate our proposed approaches A and B (Sec.3.1, and Sec. 3.2) using the predefined hierarchies from the datasets (Sec.4.2.1). We then evaluate our proposed approaches using the learned hierarchies (Sec.4.2.2). In our evaluation, we calculate the mean of per-class accuracies and compare the results with several baseline methods.

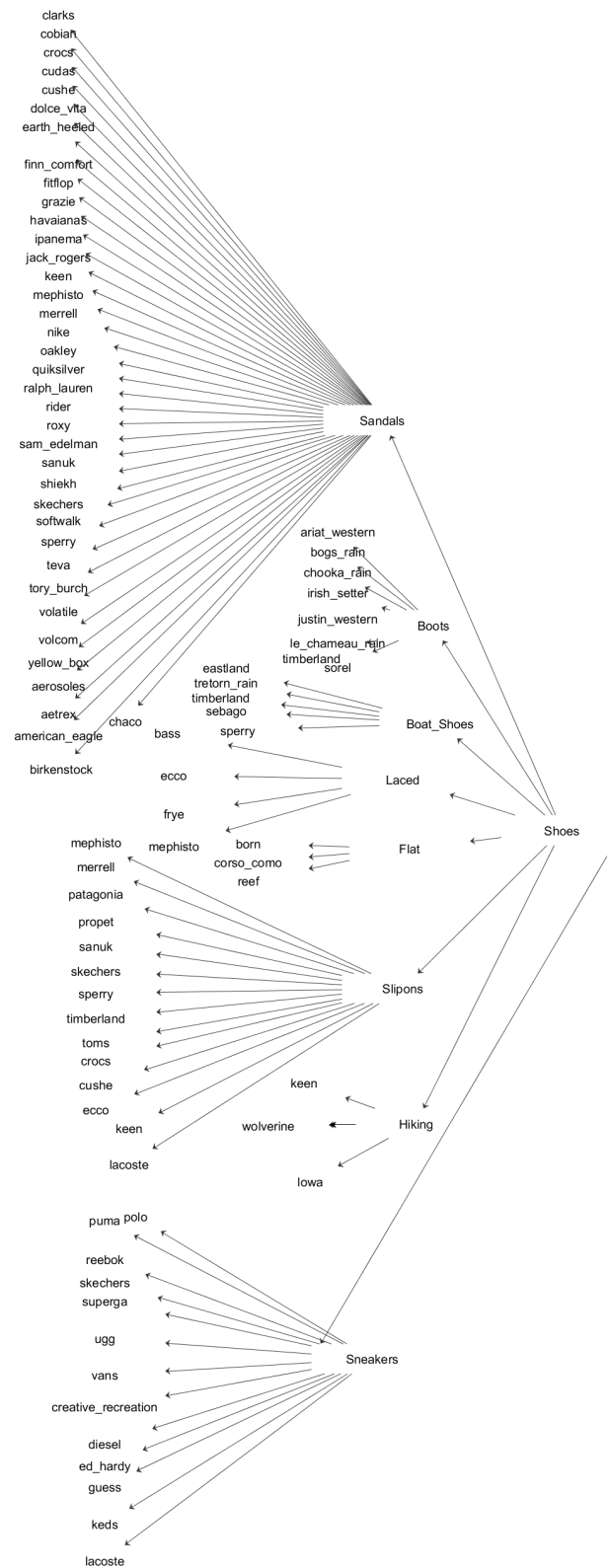


Figure 4.8: Sub-set of the Yahoo Shoes hierarchy.

Method \ Dataset	ImageNet65	AwA	CIFAR	Yahoo Shoes
Raw features	23.8%	23.1%	25.7%	57.1%
Cao et al. [6]	29.7%	24.5%	28.6%	60.4%
Our approach A	36.2%	27.5%	30.5%	64.70%
Our approach B	37.85%	29.30%	31.75%	64.85%

Table 4.1: Results of overall accuracies for our approaches A and B (Sec. 3.1, and Sec. 3.2) compared with two baseline methods: raw features, and Cao et al. [6], on four datasets.

4.2.1 Experimental results of our approaches

In this section, we use the predefined tree-shaped hierarchies to compare the performance of our proposed approaches (Sec.3.1, and Sec.3.2) with two related methods: raw features, and Cao et al. [6].

- Raw features: this baseline method learns a non-linear SVM based on the low-level features.
- Cao et al. [6]: similar to our approaches, this baseline method also uses scores of binary SVM classifiers as mid-level features. The difference is that it only considers pairs of concepts from leaf nodes. So it ignores the hierarchical structure of the object classes.

The overall accuracies of these two baselines are shown in the first two rows of Table 4.1. The overall accuracies of our approaches A and B are shown in the last two rows of Table 4.1. We can see that our proposed approaches perform significantly better than the raw features method and Cao et al. method [6]. Also, we can see that our approach B outperforms our approach A. As we have seen in chapter 3,

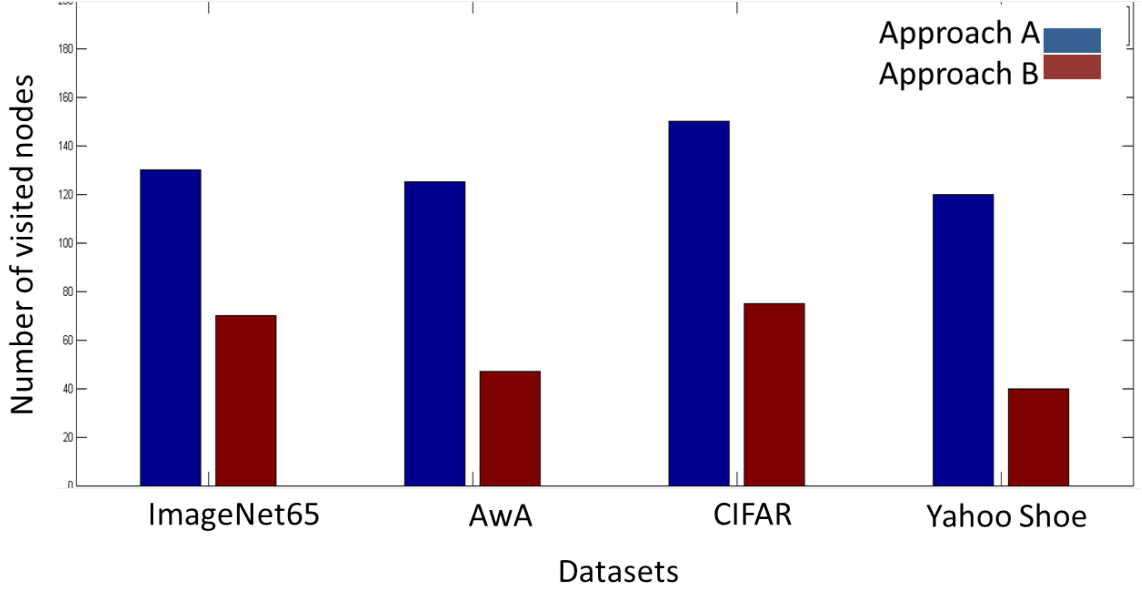


Figure 4.9: Comparison of the average number of nodes visited by test images between approach B (Sec.3.2) and approach A (Sec.3.1) on the four datasets. Approach B visits significantly less nodes due to the pruning strategy.

approach A (Sec.3.1) selects concept pairs similarly to approach B. However, it uses the concatenation of all binary classifiers as the mid-level feature, whereas approach B uses the concatenation of some relevant binary classifiers. In other words, the hierarchical structure is used in approach A when selecting concept pairs, but not used when constructing the final mid-level image representation. The time complexity of our approaches is linear to the number of the binary classifiers which are associated with all the internal nodes in the tree-shaped hierarchy. The time complexity for exploring the paths in our approach B is sub-linear in the number of nodes in the hierarchy.

In addition, compared with approach A (Sec.3.1), approach B (Sec.3.2) has the additional advantage that we only need to apply a subset of the binary SVM classifiers on a given image. In Fig. 4.9, we visualize the average number of nodes visited on

Method \ Dataset	ImageNet65	AwA	CIFAR	Yahoo Shoes
Single path [4]	28.14%	24.50%	27.41%	59.43%
All paths	36.70%	29.30%	31.10%	63.88%
Our approach B	37.85%	29.30%	31.75%	64.85%

Table 4.2: Overall accuracies for our approach B (Sec.3.2) compared with two baseline methods: single path method, and all paths method, on four datasets.

test images for both approach B and approach A. We can see that approach B visits significantly less nodes than approach A. This demonstrates that our pruning strategy is very effective.

To further illustrate the trade-off between accuracy and efficiency in our approach B (Sec.3.2), we also consider the following two hierarchy-based baseline methods:

- Single path: this method is essentially the same as Bengio et al. [4]. At each internal node, it chooses only one child to visit. In the end, we will reach a leaf node. This leaf node will give the predicted label. This method is very efficient, since it only needs to explore one single path in the hierarchy.
- All paths: this method is the extreme case of our approach, whereas no children are pruned.

The comparison with these two baselines is shown in Table 4.2. We can see that although exploring a single path is efficient, the performance is much worse. The reason is that if any internal node picks the wrong child to visit, the error cannot be corrected by any descendants. This issue can be addressed by exploring more paths in the hierarchy, which is what our approach B does (Sec. 3.2.2). We show examples of some predictions of our approach B compared with the single path method in

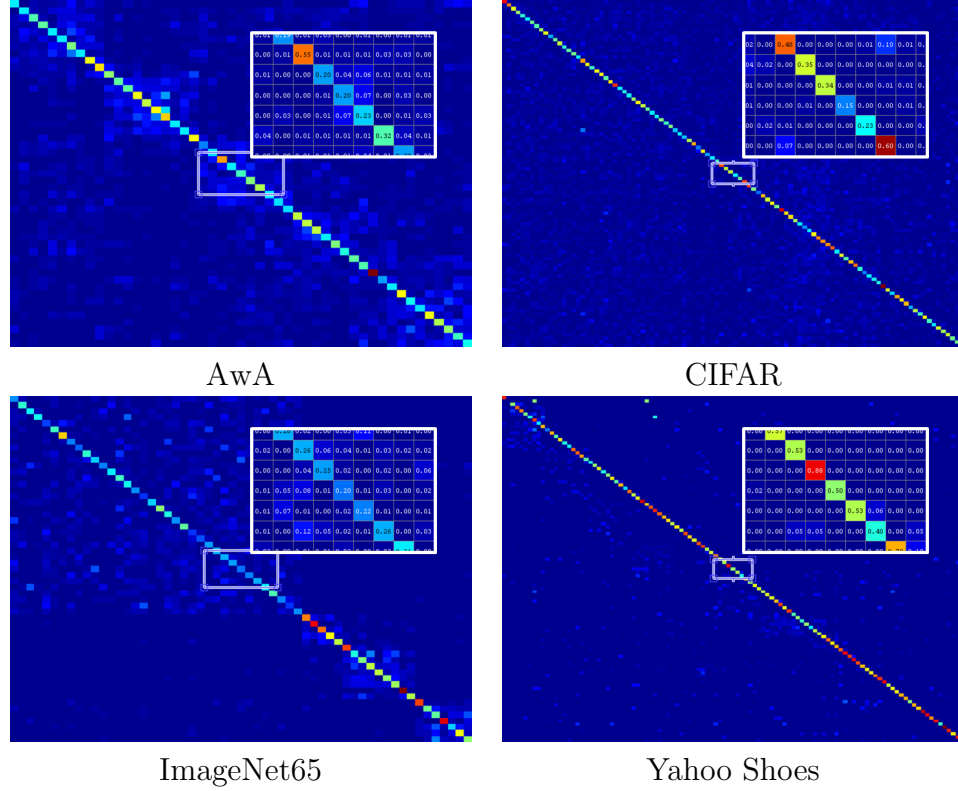


Figure 4.10: Confusion matrices of approach A (Sec.3.1) on four datasets.

Fig. 4.12, .

To show the per-class accuracies of our approaches, we illustrate the confusion matrices for the four datasets using approach A (Fig. 4.10), and approach B (Fig. 4.11).

To demonstrate the overall performance of our approaches, we show some examples of predictions of our approaches compared with baseline methods (Fig. 4.13).

4.2.2 Experimental results of building tree-shaped hierarchy

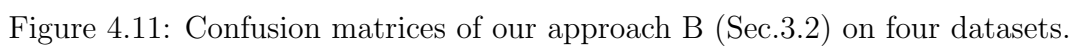
In this section, we evaluate the performance of our approaches with learned tree-shaped hierarchies (Sec.3.3), and compare it with the performance when using the

predefined tree-shaped hierarchies.

Method \ Dataset	ImageNet	AwA	CIFAR	Yahoo Shoes
Our approach A (Sec.3.1)	36.24	27.50%	30.53%	64.70%
Our approach B (Sec.3.2)	35.85%	29.30%	31.75%	62.85%
Our approach A (learned hierarchy)	38.70%	30.54%	32.90%	66.85%
Our approach B (learned hierarchy)	39.63%	33.72%	32.20%	65.12%

Table 4.3: Comparison of the performance of our approaches using the predefined tree-shaped hierarchy (first two rows), and using the learned tree-shaped hierarchy (last two rows).

The results in Table 4.3 show that using the learned tree-shaped hierarchies give better results than using the predefined tree-shaped hierarchies on these datasets.



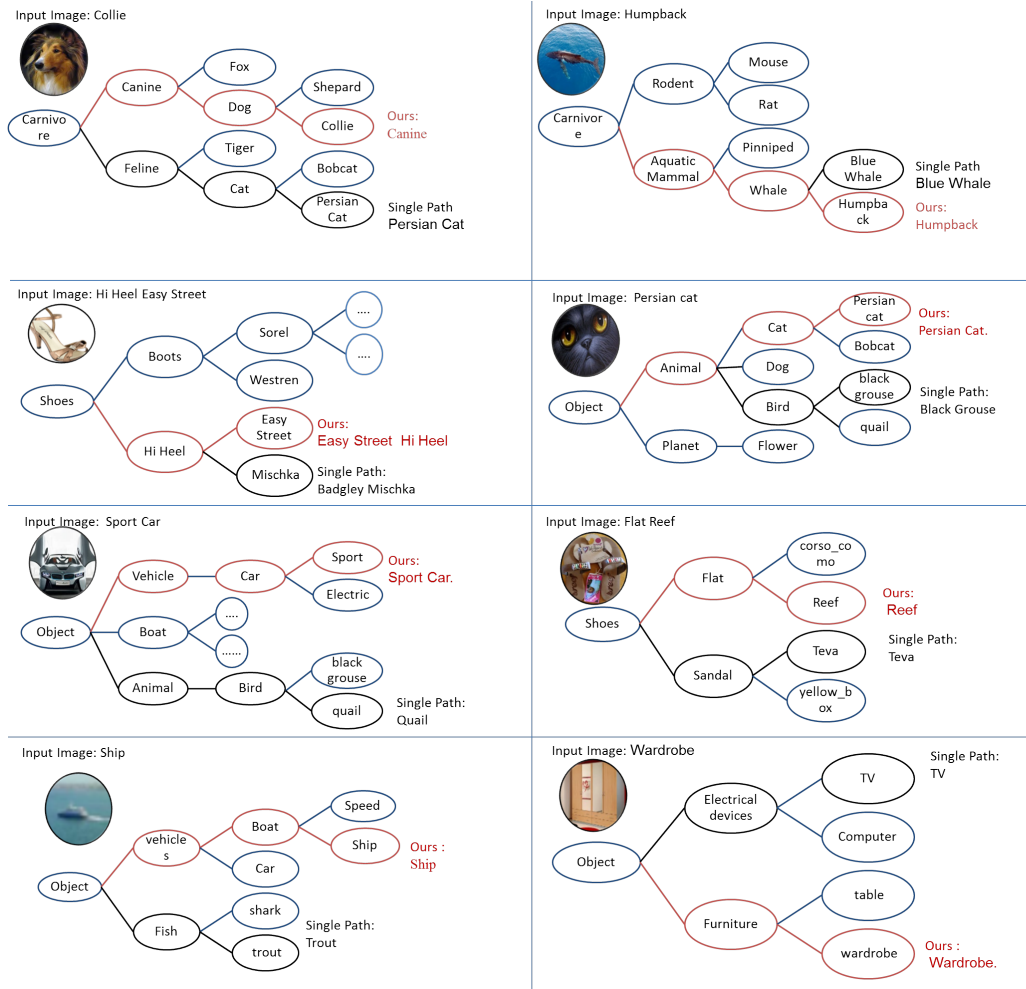






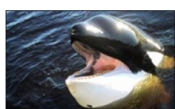




Figure 4.12: Some examples of our approach B predictions (Sec. 3.2) compared with single path method.

			
Raw features	Rat	Raccoon	Otter
Cao et al. [6]	Bear	Tiger	Collie Dog
Our approach A	Beaver	Bobcat	Fox
Our approach B	Beaver	Bobcat	Fox

			
Raw features	Limousine	House	Cloud
Cao et al. [6]	Pheasant Bird	Sweet pepper	Bed
Our approach A	Speed Boat	Can	Television
Our approach B	Speed Boat	Can	Television

			
Raw features	Gorilla	Bed	Fireboat
Cao et al. [6]	Pheasant Bird	Table	Racer
Our approach A	Dolphin	Wardrobe	Sport Car
Our approach B	Killer whale	Wardrobe	Sport Car

			
Raw features	Flat Reef	Beaver	Caw
Cao et al. [6]	Badgley Mischka	Grizzly bear	Zebra
Our approach A	Badgley Mischka	Grizzly bear	Horse
Our approach B	Easy Street Heel	Grizzly bear	Horse

Figure 4.13: Some examples of our approaches predictions compared with baseline methods.

Chapter 5

Conclusion

Although low-level image features have shown promise in many applications, their use is inherently limited since they do not capture high-level semantic information about object classes. When we move towards high-level recognition tasks, these low-level features often do not offer enough discriminative powers.

In this thesis, we have proposed two approaches that construct new mid-level feature representations for image classification. Our proposed approaches exploit the rich semantic information in the hierarchical structure to construct a rich set of binary classifiers for every object pair.

In approach A, we have constructed the mid-level representation by concatenating the output scores from all the binary classifiers in the hierarchy. In approach B, we have only considered the object pairs along the path of the given image to construct the mid-level representation. We have also proposed a method for constructing the hierarchy from data when the hierarchy is not readily available.

Our experimental results have shown the effectiveness of our approaches compared

with other methods in the literature. Also, our experimental results have shown the effectiveness of using the learned tree-shaped hierarchy compared with the predefined tree-shaped hierarchy.

Bibliography

- [1] Timo Ahonen, Abdenour Hadid, and Matti Pietikainen. Face description with local binary patterns: Application to face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(12):2037–2041, 2006.
- [2] Somayah Albaradei, Yang Wang, Liangliang Cao, and Jia Li. Learning mid-level features from object hierarchy for image classification. In *Proceedings of IEEE Winter Conference on Applications of Computer Vision*, 2014.
- [3] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110(3):346–359, 2008.
- [4] Samy Bengio, Jason Weston, and David Grangier. Label embedding trees for large multi-class tasks. In *Advances in Neural Information Processing Systems*, pages 163–171, 2010.
- [5] Anna Bosch, Andrew Zisserman, and Xavier Munoz. Representing shape with a spatial pyramid kernel. In *Proceedings of the 6th ACM international conference on Image and video retrieval*, pages 401–408. ACM, 2007.

-
- [6] Liangliang Cao, Leiguang Gong, John R Kender, Noel C Codella, and John R Smith. Learning by focusing: A new framework for concept recognition and feature selection. In *IEEE International Conference on Multimedia and Expo*, pages 1–6. IEEE, 2013.
 - [7] Jia Deng, Alexander C Berg, and Li Fei-Fei. Hierarchical semantic indexing for large scale image retrieval. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 785–792. IEEE, 2011.
 - [8] Jia Deng, Alexander C Berg, Kai Li, and Li Fei-Fei. What does classifying more than 10,000 image categories tell us? In *European Conference on Computer Vision*, pages 71–84. Springer, 2010.
 - [9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255. IEEE, 2009.
 - [10] Jia Deng, Jonathan Krause, Alexander C Berg, and Li Fei-Fei. Hedging your bets: Optimizing accuracy-specificity trade-offs in large scale visual recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3450–3457. IEEE, 2012.
 - [11] Jia Deng, Sanjeev Satheesh, Alexander C Berg, and Fei Li. Fast and balanced: Efficient label tree learning for large scale object recognition. In *Advances in Neural Information Processing Systems*, pages 567–575, 2011.
 - [12] Tianshi Gao and Daphne Koller. Discriminative learning of relaxed hierarchy for

- large-scale visual recognition. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2072–2079. IEEE, 2011.
- [13] Leonard Kaufman and Peter J Rousseeuw. Partitioning around medoids (program pam). *Finding groups in data: an introduction to cluster analysis*, pages 68–125, 1990.
- [14] Trupti M Kodinariya and Prashant R Makwana. Review on determining number of cluster in k-means clustering. *International Journal*, 1(6):90–95, 2013.
- [15] Alex Krizhevsky, Ilya Sutskever, and Geoff Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1106–1114, 2012.
- [16] Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 951–958. IEEE, 2009.
- [17] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2169–2178. IEEE, 2006.
- [18] Li-Jia Li, Hao Su, Yongwhan Lim, and Li Fei-Fei. Objects as attributes for scene classification. In *Trends and Topics in Computer Vision*, pages 57–69. Springer, 2012.
- [19] Yuanqing Lin, Fengjun Lv, Shenghuo Zhu, Ming Yang, Timothee Cour, Kai

- Yu, Liangliang Cao, and Thomas Huang. Large-scale image classification: fast feature extraction and svm training. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1689–1696. IEEE, 2011.
- [20] David G Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.
- [21] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.
- [22] Aude Oliva and Antonio Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, 2001.
- [23] Sreemanananth Sadanand and Jason J Corso. Action bank: A high-level representation of activity in video. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1234–1241. IEEE, 2012.
- [24] Eli Shechtman and Michal Irani. Matching local self-similarities across images and videos. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.
- [25] Min Sun, Wan Huang, and Silvio Savarese. Find the best path: An efficient and accurate classifier for image hierarchies. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 265–272. IEEE, 2013.
- [26] Lorenzo Torresani, Martin Szummer, and Andrew Fitzgibbon. Efficient object

- category recognition using classemes. In *European Conference on Computer Vision*, pages 776–789. Springer, 2010.
- [27] Koen EA Van De Sande, Theo Gevers, and Cees GM Snoek. Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1582–1596, 2010.
- [28] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3485–3492. IEEE, 2010.
- [29] Yahoo research labs. Yahoo! shopping shoes image content, 2013.