Learning to Share Visual Appearance for Multiclass Object Detection

Ruslan Salakhutdinov, Antonio Torralba, Josh Tenenbaum
Massachusetts Institute of Technology
rsalakhu@mit.edu, torralba@csail.mit.edu, jbt@mit.edu

Abstract

We present a hierarchical classification model that allows rare objects to borrow statistical strength from related objects that have many training examples. Unlike many of the existing object detection and recognition systems that treat different classes as unrelated entities, our model learns both a hierarchy for sharing visual appearance across 200 object categories and hierarchical parameters. Our experimental results on the challenging object localization and detection task demonstrate that the proposed model substantially improves the accuracy of the standard single object detectors that ignore hierarchical structure altogether.

1. Introduction

As we move around the world, some objects are encountered very frequently (everyday, we see many different people, cars, trees, buildings, chairs, etc.), other objects are less frequent (encountering only a few different instances per day, such as televisions or mugs), other objects are quite rare (e.g., speakers, teapots, suitcases, docks), or extremely rare (seen only a few times each year or less, such as elephants, or aircraft carriers). This distribution of learning data is very different to the distributions generally used when training object recognition algorithms. Current work on learning from few examples generally creates a setting in which there are N object classes, with M training examples available per class, with M being small. This setting is artificial as it is becoming increasingly easy to collect large amounts of training data [23, 18], at least for a subset of the object classes. In addition, this artificial distribution of training data is likely to be quite different to the distribution of data encountered by humans, or by the mobile agents, moving around the world.

In this work we focus on the more realistic setting in which we have some classes containing lots of training data and many classes containing little data. Our goal is to use frequent classes to help to learn rare classes for which it is harder to collect the training data. This biased distribution is quite frequent and emerges in most natural training domains. One of the most common examples is when looking at the frequency of words. The distribution of words approximates the Zipf's law [32]. A distribution similar to Zipf's law also has been found in several large object databases (e.g., what and where [26], labelme [23]). Other datasets have a uniform distribution over available data per class (e.g. Caltech 101, ImageNet [18], by making a big effort during the collection process in order to keep a uniform distribution over the object samples).

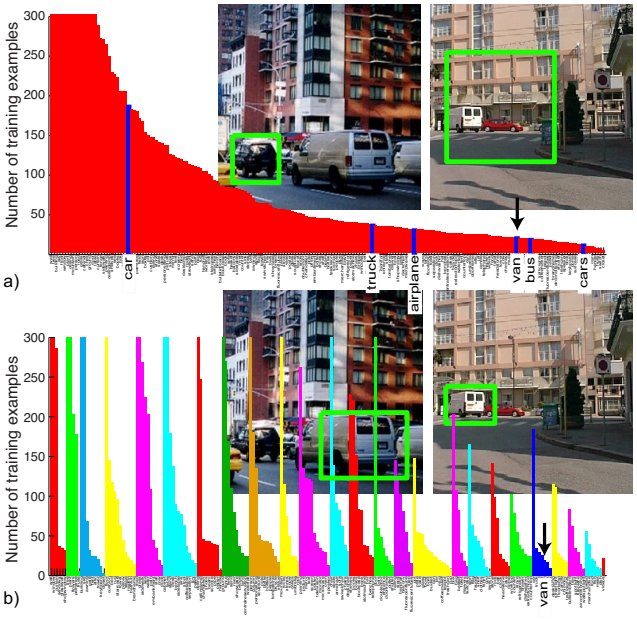

Figure 1. a) Distribution of amount of training data available per object class. Objects are sorted by decreasing amount of data. The distribution is similar to the Zipf's law, with 9 objects out of 200 accounting for 50% of all the available training data. The two images show the output of a detector trained to detect *vans* using only the training data available for the *van* class. b) Objects are grouped into clusters of objects with similar visual appearances. In this plot clusters (denoted by different colors) are sorted by the cluster mass (sum of all the samples available) and, within each cluster, objects are sorted by decreasing amount of data. Rare objects are likely to be inside a cluster with some very frequent objects. The images show detections of vans on test images without (top) and with (bottom) sharing.

Fig. 1.a shows the distribution of amount of annotated data available for 200 object categories from the database

that we will use in this paper, the SUN'09 database [31]. This database has been created by downloading scene images and then annotating the objects within each scene. There has been no effort into creating a uniform distribution over the amount of labeled data per object category. Indeed, the distribution of objects is similar to the Zipf's law with 9 objects out of 200 accounting for 50% of all the available training data. More than 100 object categories have fewer than 50 training examples each. Learning to recognize rare objects is a challenging task as we can not rely on a large amount of training data to build a reliable detector. Interestingly, if we localize a set of visually similar object classes (e.g., cars, trucks, vans, etc.) we observe that rare objects are visually similar to very frequent objects. Fig. 1.b shows the same distribution but now sorted so that similar objects are nearby (we will explain how this ordering is obtained in section 4). This new reordered distribution opens the door for effective transfer of information to better generalize from few training examples by borrowing strength from related object classes that have lots of data available.

2. Previous work

In this paper we will focus on the detection task (localizing an object within an image). Most studies on transfer learning for object recognition have focused on multiclass recognition without a background class (saying if a crop image contains an object out of M possible classes [20, 14, 25, 11, 29]). The object localization task requires efficient solutions that can be part of a window scanning. In addition, and more fundamental, the detection task needs to focus on the problem of distinguishing the objects from the background class which requires strong discriminative models in order to get good recognition performance. Few multiclass object detection systems have been proposed showing improved performance with respect to independently trained algorithms (e.g., [15, 30, 21]). Independently of the task being solved, another aspect that differentiates between multiclass detectors is the type of information shared across categories:

Sharing parts: Some of the first models in multiclass object recognition shared information via a global prior that modeled the distribution of appearance and geometry of generic object parts [8]. Discriminative voting models can share the voting components (i.e., parts) across different classes [15, 30, 2, 21, 17]. Hierarchical models also share information across classes by representing objects as compositions of shared components [13, 27, 25, 1].

Sharing attributes: Representations based on attributes [16] define a finite vocabulary that is common to all categories, with each category using a subset of the attributes. Training each attribute benefits from the data available from multiple classes, just as it happens with voting schemes that share parts. However, it is unclear if the best properties to be shared correspond to meaningful attributes.

Sharing transformations: the distribution of appearances of each object class can be modeled as being caused by two components: a canonical object appearance, and a set of transformations that can be applied to the canonical object to generate new object instances. The transformation can be shared across different classes. These methods can be applied for object classes with well defined sets of transformations such as letters [20, 28], or faces [28].

Regularization of classifier parameters: In these models there is no emphasis on the representation (objects are described by long feature vectors). Object classes transfer information by regularizing the space of classifier parameters [22, 29].

Sharing training examples: a different strategy for sharing information is, instead of working in the parameter space of the classifier, to reuse training examples from other categories to train a new category. Examples borrowed from other classes are given a lower weight in the cost function [12]. When objects are organized along a hierarchical taxonomy, nodes at different levels in the hierarchy share the training samples of all the children [19].

When sharing features across classes, one important challenge is to decide what should be the dependency among classes, i.e., to decide with which classes a new class should share information. Reliably deciding with which classes to share information can be difficult, as the decision has to be taken from the few training examples available. Models using a global prior [8, 17, 6] ignore this issue by sharing information across all categories, which provides only small benefits. Other models use external non-visual hierarchies such as wordnet [9] (e.g., [19, 12, 16]). Other models learn the relations between categories [30, 21, 22, 1, 25, 13, 29].

3. Detection Model: A Preview

Consider a challenging problem of detecting and localizing objects from a wide variety of categories such as cars, chairs, trees. Many current state-of-the-art object detection (and object recognition) systems use rather sophisticated models, based on multiple parts with separate appearance and shape components, that can cope with changes in illumination, viewpoint, shape and other visual properties. However, many of these systems [4, 10] detect objects by testing sub-windows and scoring corresponding test patches $\mathbf{x}$ with a *linear function* of the form:

$$y = \boldsymbol{\beta}^\top \Phi(\mathbf{x}), \tag{1}$$

where $\Phi(\mathbf{x})$ may represent a vector of different image features at multiple scales (e.g. HOG feature pyramid), and $\boldsymbol{\beta}$ represents a vector of model parameters.

In this work we focus on training detection systems for multiple object classes. Our goal is to learn a hierarchi-

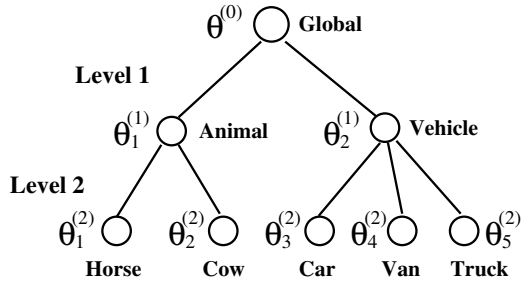

Figure 2. Hierarchical Classification Model: Parameters of each class are given by the sum of parameters along the tree.

cal model so that rare objects can borrow statistical strength from related frequent objects. Our overall framework for learning a hierarchical classification model is shown in Fig. 2, where parameters of each class are given by the sum of parameters along the tree [24, 5, 7]. For example, the parameter vector of the 'van' class is given by:

$$\boldsymbol{\beta}^{(van)} = \boldsymbol{\theta}^{(0)} + \boldsymbol{\theta}_2^{(1)} + \boldsymbol{\theta}_4^{(2)}, \tag{2}$$

where $\boldsymbol{\theta}^{(0)}$ is the *global* parameter vector shared across all classes, $\boldsymbol{\theta}^{(1)}$ is the first-level, or *super-class*, parameter vector shared between groups of related classes (e.g. 'vehicle' and 'animal'), and $\boldsymbol{\theta}^{(2)}$ is the second-level, or *class-specific*, parameter vector.

Sharing the common super-class parameters allows us to introduce prior correlations between parameters of nearby classes in the hierarchy. Classes that share a common parent node are likely to have more similar parameter vectors *a-priori*. Observe that setting $\boldsymbol{\theta}^{(0)}$ and $\boldsymbol{\theta}^{(1)}$ to zero would recover the standard classification setting of Eq. 1. As we demonstrate in our experimental results, learning how to organize visually related objects into a coherent hierarchy can substantially improve model performance.

4. Hierarchical Classification Model

We now introduce a general Bayesian framework for learning hierarchical classification models and learning how to group classes into super-classes for parameter sharing.

4.1. Learning Separate Classification Models

Consider a classification problem where we observe a dataset $\mathcal{D}$ of N labeled training examples and we do not assume any hierarchical structure. Each example belongs to one of K classes (e.g. 200 object classes), and each class $k \in \{1, ..., K\}$ contains a set of n_k labeled examples: $\mathcal{D}^{(k)} = (\mathbf{x}_1^{(k)}, y_1^{(k)}), ..., (\mathbf{x}_{n_k}^{(k)}, y_{n_k}^{(k)})$. We let $\mathbf{x}_i^{(k)} \in \mathrm{R}^D$ denote the input feature vector of length D for the training case i that belongs to class k, and $y_i^{(k)}$ be the corresponding class label. We further assume a binary representation for class labels[1], i.e. $y_i^{(k)} \in \{-1, 1\}$, indicating whether a training example i belongs to class k.

For binary classification problems, we can use a simple logistic regression model. In particular, for each class k we model the probability of a positive instance as:

$$p(y_i^{(k)} = 1 | \boldsymbol{\beta}^{(k)}) = \frac{\exp({\boldsymbol{\beta}^{(k)}}^\top \mathbf{x}_i^{(k)})}{1 + \exp({\boldsymbol{\beta}^{(k)}}^\top \mathbf{x}_i^{(k)})}, \tag{3}$$

where $k = 1, ..., K$ ranges over classes, and $\boldsymbol{\beta}^{(k)} \in \mathrm{R}^D$ is the vector of length D of unknown parameters, or regression coefficients, for class k. We further place *zero-mean spherical* Gaussian priors over model parameters $\boldsymbol{\beta} = \{\boldsymbol{\beta}^{(1)}, ..., \boldsymbol{\beta}^{(K)}\}$:

$$p(\boldsymbol{\beta}) = \prod_{k=1}^{K} p(\boldsymbol{\beta}^{(k)}) = \prod_{k=1}^{K} \mathcal{N}(0, \frac{1}{\lambda}\mathrm{I}), \tag{4}$$

where $\mathcal{N}(\mu, \Sigma)$ denotes a Gaussian distribution with mean μ and covariance Σ. The log of the posterior distribution over the unknown parameters is proportional to:

$$\log p(\boldsymbol{\beta}|\mathcal{D}) \propto \sum_{k=1}^{K} \sum_{i=1}^{n_k} \log p(y_i^{(k)} | \mathbf{x}_i^{(k)}, \boldsymbol{\beta}^{(k)}) + \log p(\beta^{(k)}).$$

It is informative to make an explicit connection between our approach and other commonly used non-probabilistic models. It can be easily verified that maximizing the log-posterior over parameters $\boldsymbol{\beta}$ (or finding a MAP estimate) is equivalent to minimizing the cross-entropy *loss function* with quadratic regularization terms:

$$E = \sum_{k=1}^{K} \left[\sum_{i=1}^{n_k} Loss(y_i^{(k)}, \mathbf{x}_i^{(k)}, \boldsymbol{\beta}^{(k)}) + \frac{\lambda}{2} ||\boldsymbol{\beta}^{(k)}||^2 \right], \tag{5}$$

where the cross entropy objective is given by:

$$Loss(y, \mathbf{x}, \beta) = - \sum_{j \in \{-1,1\}} \mathrm{I}\{y = j\} \log p(y = j | \mathbf{x}, \boldsymbol{\beta}). \tag{6}$$

The cross-entropy objective is convex and can be optimized efficiently using standard convex optimization solvers. The formulation of Eq. 5 is quite general. If, instead of probabilistic log-loss, we were to use a hinge-loss:

$$Loss(y, \mathbf{x}, \beta) = \max(0, 1 - y\boldsymbol{\beta}^\top \mathbf{x}), \tag{7}$$

then we would recover the classical SVM objective.

The above probabilistic classification model, which we call **SingleClass**, treats classes as unrelated entities without introducing any hierarchical structure. Indeed, the objective of Eq. 5 decomposes into K sub-problems, which amounts to fitting K separate binary classification models without borrowing any information between related classes.

[1]This is a standard '1 *vs.* all' classification setting.

4.2. Learning when Class Hierarchy is Available

Suppose that our original K classes are partitioned into S higher-level classes, or super-classes, e.g. 'horse' and 'sheep' belong to the 'animal' super-category, whereas 'car' and 'truck' belong to the 'vehicle' super-class (see Fig. 2). We will represent such partition by a vector $\mathbf{z}$ of length K, each entry of which is $z_k \in \{1, ..., S\}$. Hence z_k specifies which super-category the basic-level class k belongs to.

Our overall approach for modelling this structure is to assign to each node in the hierarchy a separate parameter vector. We then associate each class with one of the leaf nodes in the tree, whose parameter vector is given by the sum of all parameters along the tree leading to that leaf node:

$$\boldsymbol{\beta}^{(k)} = \boldsymbol{\theta}^{(0)} + \boldsymbol{\theta}^{(2)}_{z_k} + \boldsymbol{\theta}^{(2)}_{k}, \tag{8}$$

where $\boldsymbol{\theta}^{(0)}, \boldsymbol{\theta}^{(1)}_s, \boldsymbol{\theta}^{(2)}_k \in \mathrm{R}^D$ represent the global, super-class, and class-specific parameter vectors. For each class, we model the probability of a positive instance using Eq. 3. We further place zero-mean spherical Gaussian priors over hierarchical parameters $\boldsymbol{\beta} = \{\boldsymbol{\theta}^{(0)}, \boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)}\}$:

$$\begin{aligned} p(\boldsymbol{\theta}^0) &= \mathcal{N}\big(0, \frac{1}{\lambda_0}\mathrm{I}\big), \quad p(\boldsymbol{\theta}^{(1)}_s) = \mathcal{N}\big(0, \frac{1}{\lambda_1}\mathrm{I}\big), \\ p(\boldsymbol{\theta}^{(2)}_k) &= \mathcal{N}\big(0, \frac{1}{\lambda_2}\mathrm{I}\big), \end{aligned} \tag{9}$$

where $s = 1, ..., S$ and $k = 1, ...K$. Observe that in this probabilistic model, that we call **SharedClass**, classes that belong to the same super-category are likely to have more similar parameter vectors.

Drawing on connection to the SingleClass model, the parameter learning problem can be formulated as minimizing the negative of the log-posterior with respect to $\boldsymbol{\beta}$:

$$\begin{aligned} E \quad = \quad & Loss(\mathbf{Y}, \mathbf{X}, \boldsymbol{\beta}, \mathbf{z}) + \frac{\lambda_0}{2}||\boldsymbol{\theta}^{(0)}||^2 \\ & + \frac{\lambda_1}{2}\sum_{s=1}^{S}||\boldsymbol{\theta}^{(1)}_s||^2 + \frac{\lambda_2}{2}\sum_{k=1}^{K}||\boldsymbol{\theta}^{(2)}_k||^2, \end{aligned} \tag{10}$$

where the $Loss(\mathbf{Y}, \mathbf{X}, \boldsymbol{\beta}, \mathbf{z})$ function is given by Eq. 6 but with parameters $\boldsymbol{\beta}$ respecting the given fixed tree structure.

4.3. Modelling the number of super-categories

So far we have assumed that our model is presented with a partition vector $\mathbf{z}$, that defines a fixed two-level tree hierarchy. This model corresponds to a standard hierarchical Bayesian model that assumes a fixed hierarchy for sharing parameters. If, however, we are not given a predefined tree structure, we need to infer the distribution over the possible partitions of the basic-level categories into super-categories. To this end, we place a nonparametric Chinese Restaurant Prior (CRP) over $\mathbf{z}$, which allows the flexibility of having an unknown and potentially unbounded number of groups, or super-classes, over the basic-level classes. In particular, the CRP prior recursively extends a partition over $k-1$ objects to a new object as follows:

$$P(z_k = s | z_1, ..., z_{k-1}) = \begin{cases} \frac{m^k}{k-1+\gamma} & m^k > 0 \\ \frac{\gamma}{k-1+\gamma} & k \text{ is new} \end{cases}, \tag{11}$$

where m^k is the number of object classes previously assigned to super-category s and γ is the concentration parameter that controls the probability of creating a new super-category. Unlike many conventional hierarchical Bayesian models, here we infer both the model parameters as well as the hierarchy for sharing those parameters.

In this work, due to computational reasons, we primarily focus on learning two-level tree hierarchies. However, extension to learning multi-level tree structured model can be accomplished using a nested CRP prior [3], which extends CRP to nested sequence of partitions, one for each level of the tree.

5. Model Learning

Within our hierarchical framework, inferences about model parameters at all levels of hierarchy as well as the partition structure $\mathbf{z}$ can be made by running a Markov chain whose stationary distribution is the posterior distribution over the model parameters and the partition vector: $p(\boldsymbol{\theta}, \mathbf{z} | \mathbf{X}, \mathbf{Y})$. In many application domains, such as the one considered in this paper, we will be interested in finding the most probable tree structure and the most probable configuration of the model parameters (maximum a-posteriori (MAP) solution). When the tree structure $\mathbf{z}$ is unknown, our inference process will alternate between fixing the tree $\mathbf{z}$ and maximizing over the model parameters $\boldsymbol{\beta}$ and then fixing $\boldsymbol{\beta}$ while maximizing over the tree structure.

Learning parameters: Given the current tree structure $\mathbf{z}$, the problem of parameter learning is the same as the one discussed in section 4.2 when the tree hierarchy is known. Indeed, we can write down the objective we want to minimize using Eq. 10. This objective can optimized efficiently using an iterative coordinate-descend procedure, summarized in Algorithm 1. Observe that optimizing super-class (and class-specific) parameters can be performed in parallel. For example, given the global and super-class parameters $\{\boldsymbol{\theta}^{(0)}, \boldsymbol{\theta}^{(1)}\}$, the objective of Eq. 10 decomposes into K separate problems (as was the case for the SingleClass model), and can optimized in parallel. This significantly speeds up model training, particularly when dealing with a large number of classes.

Learning the tree structure: Consider inferring the assignment z_k, or inferring which super-category the basic-level class k should be assigned to (see Fig. 3). Given the current setting of the model parameters $\boldsymbol{\beta}$ and combining

Algorithm 1 Parameter Optimization.

Given $\mathbf{z}$: the tree structure, S: the number of super-categories, and K: the number of basic-level categories.
repeat
  Optimize $\boldsymbol{\theta}^{(0)}$ using Eq. 10, given $\boldsymbol{\theta}^{(1)}$ and $\boldsymbol{\theta}^{(2)}$.
  for $s = 1 : S$ (run in parallel) **do**
    Optimize $\boldsymbol{\theta}_s^{(1)}$ using Eq. 10, given $\boldsymbol{\theta}^{(0)}$ and $\boldsymbol{\theta}^{(2)}$.
  end for
  for $k = 1 : K$ (run in parallel) **do**
    Optimize $\boldsymbol{\theta}_k^{(2)}$ using Eq. 10, given $\boldsymbol{\theta}^{(0)}$ and $\boldsymbol{\theta}^{(1)}$.
  end for
until Converged

the likelihood term with the CRP(γ) prior, the posterior over the assignment z_k is proportional to:

$$p(z_k|\boldsymbol{\beta}, \mathbf{z}_{-k}, \mathbf{Y}^{(k)}, \mathbf{X}^{(k)}) \propto p(\mathbf{Y}^{(k)}|\boldsymbol{\beta}^{(k)}, z_k, \mathbf{X}^{(k)}) p(z_k|\mathbf{z}_{-k}), \quad (12)$$

where $\mathbf{z}_{-k}$ denotes a vector $\mathbf{z}$ but with z_k omitted. The prior term $p(z_k|\mathbf{z}_{-k})$ is given by the CRP of Eq. 11 and the likelihood for class k takes the following form:

$$p(\mathbf{Y}^{(k)}|\boldsymbol{\beta}^{(k)}, z_k, \mathbf{X}^{(k)}) = \prod_i^{n_k} p(y_i^{(k)}|\mathbf{x}_i^{(k)}, z_k, \boldsymbol{\beta}^{(k)}). \quad (13)$$

For faster inference, instead of running a Markov chain Monte Carlo, we simply compute the most probable assignment of $z_k \in \{1, ..., S+1\}$:

$$z_k^* = \arg\max_{z_k} \log p(z_k|\boldsymbol{\beta}, \mathbf{z}_{-k}, \mathbf{Y}^{(k)}, \mathbf{X}^{(k)}) \quad (14)$$
$$= \arg\max_{z_k} \left[\log p(\mathbf{Y}^{(k)}|\boldsymbol{\beta}^{(k)}, z_k, \mathbf{X}^{(k)}) + \log p(z_k|\mathbf{z}_{-k}) \right].$$

We note that the basic-level class can either be placed under one of the existing S super-categories, or create its own new super-category, $S+1$, if it is sufficiently different from all of the remaining classes. With a nonparametric prior, the number of super-classes may grow or shrink, which allows us to automatically search over the number of super-classes.

Ideally, in order to allow a more efficient search over the possible structures, we would want to integrate over the model parameters $\boldsymbol{\beta}$: $p(\mathbf{Y}|\mathbf{z}, \mathbf{X}) = \int_{\boldsymbol{\beta}} p(\mathbf{Y}|\boldsymbol{\beta}, \mathbf{z}, \mathbf{X}) p(\boldsymbol{\beta})$. However, integrating over the model parameters can be computationally intractable (as is the case in this paper). A simpler alternative, which we adopt in this work, is to replace the intractable integration with a single point estimate that maximizes the log-posterior of the label. In this case, the most probable assignment of $z_k \in \{1, ..., S+1\}$:

$$z_k^* = \arg\max_{z_k} \left[\max_{\beta^{(k)}} \left[\log p(\mathbf{Y}^{(k)}|\boldsymbol{\beta}^{(k)}, z_k, \mathbf{X}^{(k)}) + \log p(\boldsymbol{\beta}^{(k)}) \right] + \log p(z_k|\mathbf{z}_{-k}) \right], \quad (15)$$

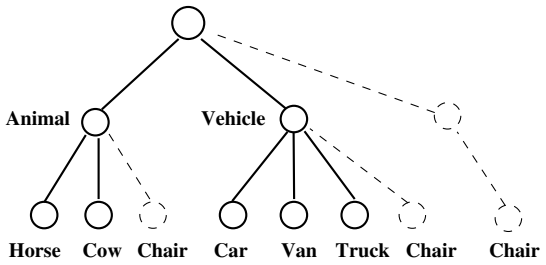

Figure 3. Learning a tree hierarchy. The 'chair' node can be attached to one of the existing super-categories: 'animal' or 'vehicle', or create its own novel super-category.

where the inner maximization over parameters $\beta^{(k)}$ can be performed efficiently using Eq. 10 (see Algorithm 1). The above formula has a very intuitive interpretation. It is composed of three terms: The log-likelihood term $\log p(\mathbf{Y}|\boldsymbol{\beta}, \mathbf{z}, \mathbf{X})$ that measures the data fit, the log-prior term $\log p(\boldsymbol{\beta})$ that measures the complexity of model parameters (parameter regularization), and the log-prior term $\log p(\mathbf{z})$ that measures the complexity of tree structures (prefers simpler trees).

To gain some intuition as to how the tree structure learning proceeds, consider attaching the 'chair' class to one of the existing branches in the tree (see Fig. 3). The 'chair' node can be attached to one of the existing super-categories: 'animal' or 'vehicle', or create its own new super-category[2]. We can then evaluate the log-posterior of the 'chair' class belonging to each super-class using Eq. 15. This in turn will allow us to either assign 'chair' to one of the existing super-classes, or create a new super-class.

The inference process continues to alternate between relearning the tree structure, given all objects classes, and refitting model parameters until convergence. This alternating joint optimization will allow us to reach the local maximum of the log-posterior probability. As we show in our experiments, this algorithm can discover a coherent tree hierarchy for sharing parameters across 200 visual categories.

6. Experimental results

Our proposed hierarchical classification framework can be directly applied to learning an object detection model of [4, 10]. In particular, [10] use a binary classification model that scores an example $\mathbf{x}$ with a linear function: $y = \boldsymbol{\beta}^\top \Phi(\mathbf{x})$, where $\Phi(\mathbf{x})$ represents a concatenation of the HOG feature pyramid plus part displacement features, and $\boldsymbol{\beta}$ represents model parameters. To get a probabilistic prediction, we use the logistic regression model of Eq. 3:

$$p(y = 1|\boldsymbol{\beta}) = \frac{\exp(\boldsymbol{\beta}^\top \Phi(\mathbf{x}))}{1 + \exp(\boldsymbol{\beta}^\top \Phi(\mathbf{x}))}, \quad (16)$$

[2]For faster inference, we fit class-specific parameters $\theta^{(2)}$ to each one of three super-classes, while holding the other parameters fixed.

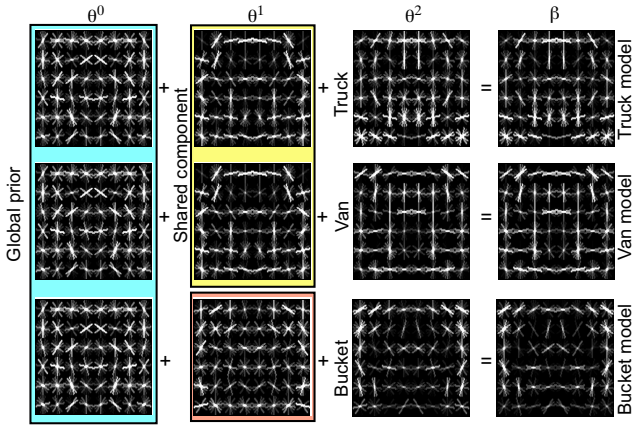

Figure 4. Learned hierarchical parameters for the 'truck', 'van', and 'bucket' object categories (only the root filter is shown). Image intensity was normalized for better visualization.

In [10], optimization over $\boldsymbol{\beta}$ is performed using a linear SVM objective with a standard hinge-loss[3] as in Eq. 7.

By modelling probabilistic predictions as in Eq. 16, learning the tree hierarchy can be performed as described in section 5. However, since we are optimizing, rather than computing the full posterior, we can undergo a probabilistic interpretation, and replace the maximization of the log-posterior with minimization of the hinge-loss. This allows us to reuse an efficient C++ implementation of [10] with minimal changes. The regularization parameters λ (Eq. 10) and CRP parameter γ (Eq. 11) are set to one. We evaluate our model using the PASCAL VOC 2008 protocol. It took three iterations for our algorithm to converge, which is about five times more expesive compared to training a SingleClass model.

6.1. Details of the Dataset

We divide SUN'09 dataset [31] into two sets: one for training and the other one for testing. The training set contains 4,082 images with 32,855 training examples, and the test set contains 9,518 images with 75,362 test examples. The 200 object categories used in our experiments (see Fig. 1) contain a wide variety of classes ranging from from regions (e.g., road, sky, field) to well defined objects (e.g., car, sofa, refrigerator, sink, mug, bed) and highly deformable objects (e.g., river, towel, curtain). Nine objects (out of 200) account for 50% and 83 objects account for 90% of all the training examples. There are 17 classes with more than 300 examples, and 14 with more than 500, and 109 classes containing less than 50 training examples.

6.2. Results

In all of our experiments we compare performance of our hierarchical SharedClass model to the following three baseline models. The first model, SingleClass, ignores hierarchical structure altogether and is trained using '1 vs. all' setting. The second model, called WordnetTree, uses wordnet [9] to define a *fixed tree hierarchy*[4]. The third model, called GlobalPrior, uses a single global prior $\boldsymbol{\theta}^{(0)}$ shared across all visual categories without learning a hierarchy for parameter sharing. This model, similar in spirit to [8], could learn a set of useful features common to all object categories.

[3]The actual objective involves so-called deformable parts that are learned using a latent SVM framework. But his difference is transparent for our formulation.

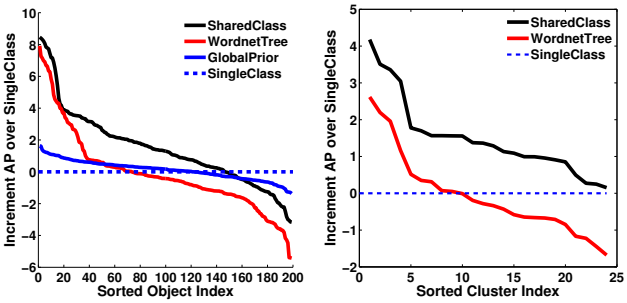

Figure 5. **Left:** Improvement of the SharedClass, WordnetTree and GlobalPrior models over the SingleClass model. Object categories are sorted sorted by the improvement in the detection task. **Right:** The average of the AP improvement of all objects within each of the 24 learned clusters. Clusters are sorted by the improvement in the detection task.

Table 1. AP averaged over all 200 object categories for the SingleClass, SharedClass, WordnetTree and GlobalPrior models.

Model	Shared	GlobalPrior	Single	WordNet
AP	**8.34**	6.87	6.98	6.76

We first tested the ability of our model to learn coherent super-categories. Figure 1 shows 200 basic-level categories along with a typical partition that our model discovers Observe that many of the super-categories contain semantically similar basic-level categories. For example, some of the discovered clusters contain: {'chair', 'armchair', 'seats', 'swivel chair', 'deck chair'}; {'table', 'side table', 'coffee table', 'desk', 'stand','stool'}; {'car', 'bus', 'truck', 'van', 'airplane'} Fig. 4 visualizes a hierarchical structure of the learned models for the 'truck', 'van', and 'bucket' objects.

Figure 5 (left panel) displays the improvement in average precision-recall (AP) of SharedClass, WordnetTree, and GlobalPrior models for all object categories over the SingleClass model. Observe that over 150 categories benefit in different degrees from learning a hierarchy for parameter sharing. Five objects with the largest improvement in AP are: 'shop window' (+8.46), 'double door' (+8.40), 'van' (+8.32), 'armchair' (+8.22) and 'coffee table' (+8.07). We note that all of these objects borrow visual appearance from other frequent objects, including 'window', 'door', 'chair', 'car', and 'table'. Five objects with the largest decrease in AP are: 'umbrella' (-3.16), 'merchandise' (-3.14), 'toy' (-3.05), 'meat' (-2.98), and 'fruits' (-2.81). Many of these

[4]For fair comparison, we constructed 24 semantically similar super-categories using the WordNet

Table 2. AP improvement of the SharedClass over the SingleClass model as a function of the number of the training examples.

Number of Training Examples	Number of Objects	Average AP improvement
1-10	9	1.24
11-20	29	1.62
20-50	**73**	**1.89**
51-100	32	1.86
101-300	40	1.41
> 301	17	0.83

objects are abstract, and their visual appearance is very different from other object categories. They often get attached to the 'wrong' clusters which introduces a negative transfer.

Fig. 5, right panel, shows that *for each* of the 24 learned clusters, sharing detectors improves over the SingleClass baseline. The WordnetTree and GlobalPrior models, on the other hand, perform far worse compared to the SharedClass model. Table 1 further reveals that the use of WordNet actually decreases model performance. This is hardly surprising as semantically similar objects may not share visual similarity. Using a single global prior, on the other hand, only marginally improves upon SingleClass. This result clearly demonstrates that it is crucial to *learn the hierarchy*.

Table 2 shows improvement of the SharedClass as a function of the number of the training examples. Even though most of the performance gains occur for the non-frequent objects (containing 20-50 training examples), we also see substantial improvements for the frequent objects (containing over 300 object categories). This is quite surprising as it shows that even frequent objects can benefit from sharing visual appearance between similar objects.

Figure 6 displays a 200×200 confusion matrix for the SingleClass and SharedClass models. The confusion matrix of the SharedClass model is more block-structured, indicating that sharing objects is more likely to introduce errors between objects that belong to the same super-class (e.g. confuse trucks with cars, deck-chairs with chairs, etc.). Tables 3 and 4 precisely illustrate this point.

To provide a more intuitive understanding for why our hierarchical learning succeeds in discovering meaningful super-categories, consider a stand-alone car detector, trained on 185 training examples. Table 3 shows that the car detector achieves a very good AP of 59.02. However, a car detector is also able to detect other related, but rare object categories (e.g. 'van', 'bus', 'truck'). This is exactly what allows us to learn coherent clusters for parameter sharing. A stand-alone truck detector, on the other hand, confuses trucks with completely unrelated objects (e.g. 'sky', 'floor'). Of course, this is not surprising – there are just not enough training examples to learn a good truck detector. Learning a hierarchy, however, allows us achieve a far better AP (see table 4).

Finally, Fig. 7 shows detection results on some of the test images, where for each image we show a single most confident detection according to the SingleClass (top panel) and

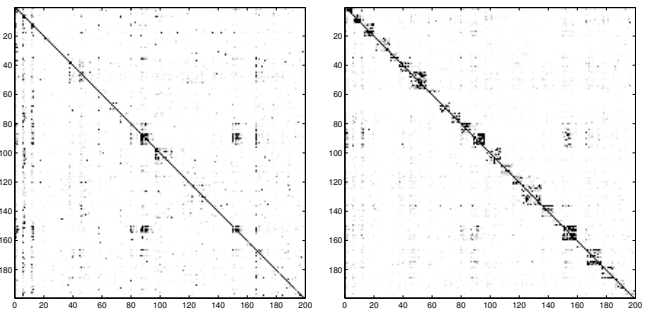

Figure 6. Confusion matrix for the SingleClass (left) and SharedClass (right) models.

Table 3. **SingleClass** 10 sample objects categories along with the three most confused object classes. Test AP is shown in parenthesis. Objects in bold represent frequent classes.

Object Category	Three Most Confused Objects		
car (59.02)	van (1.97)	truck (1.84)	bus (0.81)
bus (0.87)	wall (0.69)	building (0.22)	pot (0.10)
truck (2.88)	sky (1.82)	floor (1.53)	road (0.51)
van (8.19)	car (13.86)	truck (0.59)	tree (0.40)
chair (21.15)	armchair (0.91)	stool (0.67)	deck chair (0.35)
deck chair (2.31)	rug (0.46)	tree (0.16)	chair (0.11)
armchair (9.23)	chair (1.40)	floor (0.21)	table (0.19)
table (8.65)	desk (1.16)	stool (0.65)	coffee table (0.54)
coffee table (1.12)	table (9.21)	sofar (0.42)	floor (0.18)
desk (1.31)	floor (9.09)	building (4.55)	door (4.54)

Table 4. **SharedClass** 10 sample objects categories along with the three most confused object classes.

Object Category	Three Most Confused Objects		
car (59.21)	van (2.82)	truck (2.23)	bus (0.92)
bus (4.18)	car (4.65)	van (2.12)	truck (1.21)
truck (10.18)	car (12.56)	van (1.87)	bus (1.12)
van (16.51)	car (11.32)	truck (0.78)	bus (0.75)
chair (22.34)	armchair (1.33)	stool (0.51)	deck chair (0.29)
deck chair (8.56)	chair (2.37)	table (0.27)	armchair (0.19)
armchair (17.45)	chair (4.61)	deck chair (1.45)	desk (1.18)
table (11.24)	stool (2.12)	desk (1.91)	coffee table (0.63)
coffee table (9.19)	table (4.76)	side table (0.67)	sofar (0.36)
desk (6.14)	stand (2.27)	table (1.52)	armchair (1.43)

SharedClass (bottom panel) models. In many cases, the hierarchical model accurately localizes the correct object, but fails to place a bounding around the full object. We argue that this type of failure is much more tolerable compared to the SingleClass model, that often produces false detections of completely unrelated objects.

7. Conclusion

In this paper we presented a hierarchical classification model that allows rare objects to borrow statistical strength from related objects that may have many training instances. Our experimental results show that our model, in addition to efficient learning hierarchical parameters, is able to discover coherent super-categories from 200 object classes. We further demonstrated that our model substantially improves the accuracy of the stand-alone object detectors that do not learn the hierarchy for sharing parameters.

Our proposed framework is general and can be directly applied to many other vision tasks, such as object recogni-

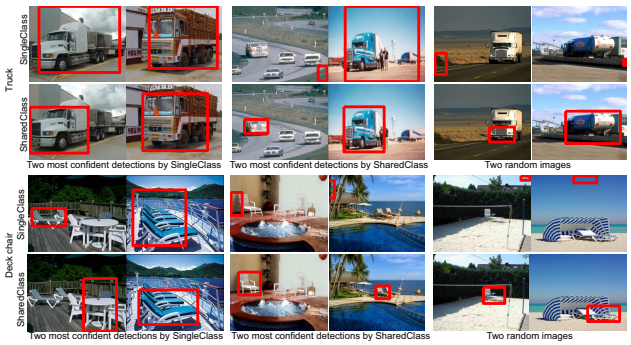

Figure 7. Detection examples of the SingleClass (top) and SharedClass (bottom) models. Each row contains 6 images: 2 most confident detection by SingleClass, 2 most confident detections by SharedClass, and 2 random;y sampled images among test images containing that particular class.

tion, and can be further extended to learning multi-layer hierarchies. We believe that these more flexible models combined with many more object categories will be able to further improve upon the current state-of-the-art.

Acknowledgments

This work was supported by NSERC, NTT Communication Sciences Laboratory, ONR MURI N000141010933, and CAREER Award No.0747120 to A.T.

References

[1] E. Bart, I. Porteous, P. Perona, and M. Welling. Unsupervised learning of visual taxonomies. In *CVPR*, 2008. 1482

[2] E. Bart and S. Ullman. Cross-generalization: learning novel classes from a single example by feature replacement. In *CVPR*, 2005. 1482

[3] D. M. Blei, T. L. Griffiths, M. I. Jordan, and J. B. Tenenbaum. Hierarchical topic models and the nested chinese restaurant process. In *NIPS*. MIT Press, 2003. 1484

[4] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, pages 886–893, 2005. 1482, 1485

[5] O. Dekel, J. Keshet, and Y. Singer. Large margin hierarchical classification. In *ICML*, volume 69. ACM, 2004. 1483

[6] T. Deselaers, B. Alexe, and V. Ferrari. Localizing objects while learning their appearance. In *ECCV*, 2010. 1482

[7] T. Evgeniou and M. Pontil. Regularized multi-task learning. In *ACM SIGKDD*, 2004. 1483

[8] L. Fei-Fei, R. Fergus, and P. Perona. A bayesian approach to unsupervised one-shot learning of object categories. In *IEEE Intl. Conf. on Computer Vision*, 2003. 1482, 1486

[9] C. Fellbaum. *Wordnet: An Electronic Lexical Database*. Bradford Books, 1998. 1482, 1486

[10] P. F. Felzenszwalb, R. B. Girshick, D. A. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Trans. Pattern Anal. Mach. Intell*, 32(9):1627–1645, 2010. 1482, 1485, 1486

[11] R. Fergus, H. Bernal, Y. Weiss, and A. Torralba. Semantic label sharing for learning with many categories. In *ECCV*, 2010. 1482

[12] R. Fergus, Y. Weiss, and A. Torralba. Semi-supervised learning in gigantic image collections. In *NIPS*, 2009. 1482

[13] S. Fidler, M. Boben, and A. Leonardis. Evaluating multi-class learning strategies in a generative hierarchical framework for object detection. In *NIPS 22*. 2009. 1482

[14] G. Griffin and P. Perona. Learning and using taxonomies for fast visual categorization. In *CVPR*, 2008. 1482

[15] S. Krempp, D. Geman, and Y. Amit. Sequential learning of reusable parts for object detection. Technical report, CS Johns Hopkins, 2002. 1482

[16] C. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *CVPR*, 2009. 1482

[17] K. Levi, M. Fink, and Y. Weiss. Learning from a small number of training examples by exploiting object categories. In *Workshop of Learning in Computer Vision*, 2004. 1482

[18] L. J. Li, G. Wang, and L. Fei-Fei. Imagenet. In *CVPR*, 2007. 1481

[19] M. Marszalek and C. Schmid. Semantic hierarchies for visual object recognition. In *CVPR*, 2007. 1482

[20] E. Miller, N. Matsakis, and P. Viola. Learning from one example through shared densities on transforms. In *CVPR*, volume 1, pages 464–471, 2000. 1482

[21] A. Opelt, A. Pinz, and A. Zisserman. Incremental learning of object detectors using a visual shape alphabet. In *CVPR (1)*, pages 3–10, 2006. 1482

[22] A. Quattoni, M. Collins, and T. Darrell. Transfer learning for image classification with sparse prototype representations. In *CVPR*, pages 1–8, 2008. 1482

[23] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. LabelMe: a database and web-based tool for image annotation. *Intl. J. Computer Vision*, 77(1-3):157–173, 2008. 1481

[24] B. Shahbaba and R. M. Neal. Improving classification when a class hierarchy is available using a hierarchy-based prior. *Bayesian Analysis*, 2(1):221–238, 2007. 1483

[25] J. Sivic, B. C. Russell, A. Zisserman, W. T. Freeman, and A. A. Efros. Unsupervised discovery of visual object class hierarchies. In *CVPR*, 2008. 1482

[26] M. Spain and P. Perona. Measuring and predicting importance of objects in our visual world. Technical report, California Institute of Technology, 2007. 1481

[27] E. Sudderth, A. Torralba, W. T. Freeman, and W. Willsky. Learning hierarchical models of scenes, objects, and parts. In *IEEE Intl. Conf. on Computer Vision*, 2005. 1482

[28] J. B. Tenenbaum and W. T. Freeman. Separating style and content with bilinear models. *Neural Computation*, 12:1247–1283, 2000. 1482

[29] T. Tommasi, F. Orabona, and B. Caputo. Safety in numbers: Learning categories from few examples with multi model knowledge transfer. In *CVPR*, pages 3081–3088, 2010. 1482

[30] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing features: efficient boosting procedures for multiclass object detection. In *CVPR*, pages 762–769, 2004. 1482

[31] J. Xiao, J. Hays, K. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010. 1482, 1486

[32] G. K. Zipf. *The Psychobiology of Language*. Houghton Mifflin, 1935. 1481