Recognition Using Regions

by

Chunhui Gu

A dissertation submitted in partial satisfaction of the
requirements for the degree of
Doctor of Philosophy

in

Engineering - Electrical Engineering and Computer Sciences

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Jitendra Malik, Chair
Professor Trevor Darrell
Professor Stephen Palmer

Spring 2012

Recognition Using Regions

Abstract

Recognition Using Regions

by

Chunhui Gu

Doctor of Philosophy in Engineering - Electrical Engineering and Computer Sciences

University of California, Berkeley

Professor Jitendra Malik, Chair

Multi-scale window scanning has been popular in object detection but it generalizes poorly to complex features (e.g. nonlinear SVM kernel), deformable objects (e.g. animals), and finer-grained tasks (e.g. segmentation). In contrast to that, regions are appealing as image primitives for recognition because: (1) they encode object shape and scale naturally; (2) they are only mildly affected by background clutter; and (3) they significantly reduce the set of possible object locations in images.

In this dissertation, we propose three novel region-based frameworks to detect and segment target objects jointly, using the region detector of Arbeláez et. al TPAMI2010 as input. This detector produces a hierarchical region tree for each image, where each region is represented by a rich set of image cues (shape, color and texture). Our first framework introduces a generalized Hough voting scheme to generate hypotheses of object locations and scales directly from region matching. Each hypothesis is followed by a verification classifier and a constrained segmenter. This simple yet effective framework performs highly competitively in both detection and segmentation tasks in the ETHZ shape and Caltech 101 databases.

Our second framework encodes image context through the region tree configuration. We describe each leaf of the tree by features of its *ancestral set*, the set of regions on the path linking the leaf to the root. This ancestral set consists of all regions containing the leaf and thus provides context as inclusion relation. This property distinguishes our work from all others that encode context either by a global descriptor (e.g. "gist") or by pairwise neighboring relation (e.g. Conditional Random Field).

Intra-class variation has been one of the hardest barriers in the category-level recognition, and we approach this problem in two steps. The first step studies one prominent type of intra-class variation, viewpoint variation, explicitly. We propose to use a mixture of holistic templates and discriminative learning for joint viewpoint classification and category detection. A number of components are learned in the mixture and they are associated with canonical viewpoints of the object through different levels of supervision. In addition, this approach has a natural extension to the continuous 3D viewpoint prediction by discriminatively learning a linear appearance model locally at each discrete view. Our systems

significantly outperform the state of the arts on two 3D databases in the discrete case, and an everyday-object database that we collected on our own in the continuous case.

The success of modeling object viewpoints motivates us to tackle the generic variation problem through component models, where each component characterizes not only a particular viewpoint of objects, but also a particular subcategory or pose. Interestingly, this approach combines naturally with our region-based object proposals. In our third framework, we form visual clusters from training data that are tight in appearance and configuration spaces. We train individual classifiers for each component and then learn to aggregate them at the category level. Our multi-component approach obtains highly competitive results on the challenging VOC PASCAL 2010 database. Furthermore, our approach allows the transfer of finer-grained semantic information from the components, such as keypoint locations and segmentation masks.

To my parents Yunfang and Weiguo, and my wife Tingting

Contents

List of Figures

List of Tables

Acknowledgments

Working on a Doctor degree for six years is a big investment in my life, and I am very grateful to have worked with quite a few wonderful people during this period that has made my investment truly worthwhile. First with full respect, I would like to thank my advisor, Professor Jitendra Malik, for being a passionate teacher, a patient trainer, and an inspirational thinker. It is amazing how much he has influenced me on his philosophy of computer vision, and still I wish I could learn more from him. His quote "what is not worth doing is not worth doing fast" directed me since the beginning of my PhD to identify and tackle *big picture* vision problems that were usually difficult and required great effort in the long run. In addition, I highly appreciate his dedication to train me to be a qualified researcher in every aspect, such as improving presentation and writing skills. The countless practice of my CVPR09 talk in Miami (with the last one done in his suite at midnight 10 hours before the actual talk next morning) has become part of my PhD memory.

I would like to thank my qualification and dissertation committees, Professor Trevor Darrell and Professor Stephen Palmer, for insightful discussion and feedback on my work. Thanks Professor Avideh Zackor for helping me with my first paper submission during PhD as an extension of my Digital Image Processing class project, as well as serving on my qualification committee. Thanks Professor Ruzena Bajcsy and Professor Shankar Sastry for discussion on various vision research opportunities in the early days of my PhD.

I would like to thank all my collaborators without whom neither my papers nor this dissertation would be accomplished. Thanks Dr. Pablo Arbeláez for "solving vision" together during my entire PhD and sharing both frustration and excitement. Thanks Joseph Lim for turning ideas into practice super fast. Thanks Dr. Xiaofeng Ren for an amazing and productive collaboration on some challenging problem that we did not plan to do in the beginning and explored from scratch. Two things that I missed the most about Seattle were its weather and research debates with Xiaofeng. Last but not least, thank Dr. Yuanqing Lin and Dr. Kai Yu from the NEC Lab America for working closely on combining their high-performance features with our system (and Fedex for solving the large-scale data transfer problem between Berkeley and Cupertino).

From 545 Soda to 750 Sutardja Dai, the "Big J Crew" has always been a supporting team to both my research and personal life at Berkeley. In addition to Pablo, I would like to thank Alex Berg, Andrea Frome, Hao Zhang and Michael Maire for advice and guidance, Subhransu Maji, Chetan Nandakumar, Patrik Sundberg, Lubomir Bourdev, Jon Barron, Thomax Brox and Cees Snoek for discussion and debates, and Bharath Hariharan, Georgia Gkioxari, Jeannette Chang and Saurabh Gupta for the new excitement.

Finally, thank my parents for their endless love, care, and sacrifice. Thanks Tingting for her company and support in my PhD journey.

Chapter 1

Introduction

Since late 90s, multi-scale window scanning has been the dominant strategy for object detection[81, 100, 23, 32]. Although researchers have shown success with this strategy on rigid and rectangular-shaped objects(e.g. face, pedestrian, car and various sign), the limitations of window scanning are still quite obvious - classifying tens to hundreds of thousands of windows per image is computationally expensive, and a naive implementation using complex features or classifiers (e.g. radial basis function (rbf) kernel of Support Vector Machines) is usually intractable; its performance on deformable or non rectangular-shaped objects, such as animals, is quite low; its bounding box output is de-linked from segmenting out the pixels of an object from the background.

These limitations motivate us to use regions as image primitives for object detection and segmentation. Here, we refer to regions as connected components whose boundaries are detected only based on bottom-up image cues, such as [46, 83, 6]. Compared to windows, regions have several appealing properties. First of all, they encode object scale and shape information naturally, so there is no need to search for object sizes. Secondly, since they are perceptually meaningful, they specify the domain on which to compute various features without being affected by clutter from background. Thirdly, region generation is purely bottom-up and class independent, thus the total number of extracted regions per image is fixed and on the order of hundreds, significantly smaller than the number of windows in the scanning approach. This reduction allows the use of more complex features or classifiers to verify object proposals at each region. The main contribution of this thesis is to introduce several region-based recognition frameworks that combine detection with segmentation, and obtain high performance on a number of object recognition benchmarks.

In early days, the computer vision community, by and large, did not have faith in the ability of generic grouping processes to deliver contours or regions of sufficiently high accuracy for recognition. This situation has been changed thanks to the emergence of recent advances in contour [65] and region detection [6] which significantly outperformed conventional methods such as Mean Shift[21] and Normalized Cuts[88] on various benchmark databases. In this dissertation, all region-based frameworks were developed on top of the output of [6], that is, for each input image, a region tree representation encoding both global and local

information of the image. Our first framework[42] utilizes region matching to propose object locations through a generalized Hough voting process. A pair of second layer classifier and segmenter efficiently verifies the detection proposals and refines the pixels of objects with respect to background, again, using regions as basic entities.

The importance of context for recognition has been widely acknowledged in the computer vision community but it is often represented by a holistic descriptor of an image[72], or pairwise potentials of neighboring pixels/segments[43, 51, 90, 78, 99, 9, 39, 49, 77]. However, contextual cues are naturally encoded through a "partonomy" of the image, the hierarchical representation relating parts to objects and to the scene. Our second *ancestry* framework[62] naturally encodes image context as inclusion relation in the region tree configuration, and achieves a significant performance boost in the multi-class image pixel classification task by incorporating contextual information.

The last topic of this dissertation attributes to modeling object intra-class variation. Intra-class variation is probably one of the most difficult cases to handle in object recognition and one main cause of low performance. We study this problem in two steps. In the first step, we experimented on a constrained problem where viewpoint changes are the main sources of intra-class variation[41]. We propose to use a mixture of holistic templates and discriminative learning to classify object viewpoints and detect objects jointly. A number of components are learned in the mixture and they are associated with canonical viewpoints of the object. We analyze the model under three conditions with different levels of supervision, with the viewpoint labels of training data being fully known, partially known, and unknown. Our systems significantly outperform the state of the arts on two 3D databases. We also investigate the extension of this model to the continuous 3D viewpoint prediction by discriminatively learning a linear appearance model locally at each discrete view. We build an everyday-object database on our own, where object viewpoints are recorded by an IMU device mounting firmly to the video camcorder.

In the second step, we study a more general case, where intra-class variations can also be induced by visually heterogenous subcategory or image occlusion/truncation. Motivated by the first step study, we know that natural image statistics suggests that objects belonging to the same category tend to form clusters in the appearance and configuration space, for instance, a majority of passenger aeroplanes are photographed in either side or 45-degree view. We refer to such visual clusters as *components*. We show that learning a model for each component independently is easier and more accurate than attempting to characterize all components in a monolithic model. We propose potential object locations and sizes from regions, and use extra annotation data to enable accurate global alignment among objects, so that objects are tight in the configuration space within a component. As a result, our approach has a significant advantage over monolithic models that it enables tasks that are finer-grained than bounding box prediction, such as prediction of object keypoint locations and segmentation mask. Our multi-component approach obtains highly competitive results on the challenging VOC PASCAL 2010 database.

1.1 Outline

Chapter 2 shows our input data for all developed region-based recognition frameworks - a region tree decomposition as a result of the region detection on an image. Then it describes the region-based Hough voting pipeline, including selecting discriminative exemplar regions for voting, the main voting algorithm, the verification classifier and the constrained segmenter. Our results show that our pipeline is not only more accurate, but also more efficient than multi-scale window scanning approach in detection and segmentation.

Chapter 3 illustrates our ancestry framework for modeling image context. We compare three weight learning approach on the ancestral nodes in order to fully understand the advantages of each approach. The huge performance gap between our models with/without ancestry reflects the power of using contextual cues for significantly improved category prediction.

Chapter 4 and 5 both address the problem of intra-class variations, where in Chapter 4 the main variation comes from viewpoint changes, and variation in Chapter 5 is unconstrained. In Chapter 4, in the discrete viewpoint prediction, we introduce a latent max-margin learning framework that predicts object classes and viewpoint labels simultaneously. We also extend this model to fit naturally into continuous viewpoint prediction. In Chapter 5, we show the alignment and training of each component model, and bounding box proposal directly from extracted regions which produces less than on average 500 windows per image. Our multi-component models obtain highly competitive results on the PASCAL VOC 2010 test set, compared to all other leading methods. Furthermore, unlike monolithic detection techniques, our approach allows the transfer of finer grained semantic information, such as keypoint locations and segmentation masks.

Chapter 2

Regions-based Hough Voting

2.1 Introduction

Ever since the early work on face detection in the late 90s [81, 100], the dominant strategy for object detection in a scene has been multi-scale scanning. A fixed size and shape window is swept across the image, and the contents of the window are input to a classifier which gives an answer to the question: is there an instance of object category C (face, car, pedestrian, etc.) in the window? To find objects of different sizes, the image is sub-sampled in a pyramid, typically with neighboring levels being a quarter octave ($\sqrt[4]{2}$) apart. This strategy continues to hold in recent papers, such as [23] on pedestrian detection and [32] on the PASCAL challenge. Various speed-ups have been offered over time, ranging from cascades [100], branch and bound strategies [54] to more efficient classifier evaluation [66].

Yet, there is something profoundly unsatisfying about this approach. First of all, classification of a window as containing, say, a horse, is not the same as segmenting out the pixels corresponding to a horse from the background. Hence, some post-process relying on quite different cues would be required to achieve that goal. Secondly, the brute-force nature of window classification is not particularly appealing. Its computational complexity is proportional to the product of the number of scales, locations, and categories. Thirdly (and this may matter more to some than to others), it differs significantly from the nature of human visual detection, where attention is directed to certain locations based on low-level salience as well as high-level contextual cues, rather than uniformly to all locations.

So what is the alternative? The default answer going back to the Gestalt school of visual perception, is in "perceptual organization". Low and middle level vision furnishes the entities on which recognition processes can operate. We then have a choice of what these entities should be: points, curves or regions? Over the last decade, low-level interest point-based features, as proposed by [87] and [64], have tended to dominate the discourse. The computer vision community, by and large, didn't have faith in the ability of generic grouping processes to deliver contours or regions of sufficiently high accuracy for recognition.

Our belief is that recent advances in contour [65] and region detection [6] make this a

propitious time to build an approach to recognition using these more spatially extended and perceptually meaningful entities. This paper focuses on using regions, which have some pleasant properties: (1) they encode shape and scale information of objects naturally; (2) they specify the domains on which to compute various features, without being affected by clutter from outside the region.

While definitely a minority trend, there has been some relevant work in the last decade using regions/segments which we review briefly. [46] estimates the 3D geometric context of a single image by learning local appearance and geometric cues on super-pixels. [83] uses a normalized cut-based multi-layer segmentation algorithm to identify segmented objects. This line of work suffers initially from unreliable regions produced by their segmentation methods. The work from [68] and [96] is most similar to our approach. However, in addition to the problem of unstable regions, [68] takes regions as whole bodies of objects and ignores local parts, while [96] represents objects as region trees but also exploits structural cues of the trees for matching and such cues may not be reliable.

Starting with regions as the basic elements of our approach, we use a generalized Hough-like voting strategy for generating hypotheses of object location, scale and support. Here, we are working in a long-standing tradition in computer vision [26, 8, 64, 57, 73, 67].

The rest of this paper is organized as follows. Section 2.2 overviews our method and describes the use of regions as elementary units. Section 2.3 describes a discriminative learning framework for region weighting. Section 2.4 describes our main recognition algorithm which has three stages: (1) voting, (2) verification, and (3) segmentation. We show our experimental results in Section 2.5, and conclude in Section 2.6. Figure 2.1 shows some of our final detection and segmentation results.

2.2 Overview of the Approach

The pipeline of our region-based recognition framework is as follows: first, each image is represented by a bag of regions derived from a region tree as shown in Figure 2.2. Regions are described by a rich set of cues (shape, color and texture) inside them. Next, region weights are learned using a discriminative max-margin framework. After that, a generalized Hough voting scheme is applied to cast hypotheses of object locations, scales, and support, followed by a refinement stage on these hypotheses which deals with detection and segmentation separately.

Region Extraction

We start by constructing a region tree using the hierarchical segmentation engine of [6]. The regions we consider are the nodes of that tree, including the root which is the entire image. We use them as the basic entities for our approach.

Figure 2.2 presents an example of our region trees, as well as a bag of regions representing the input image.

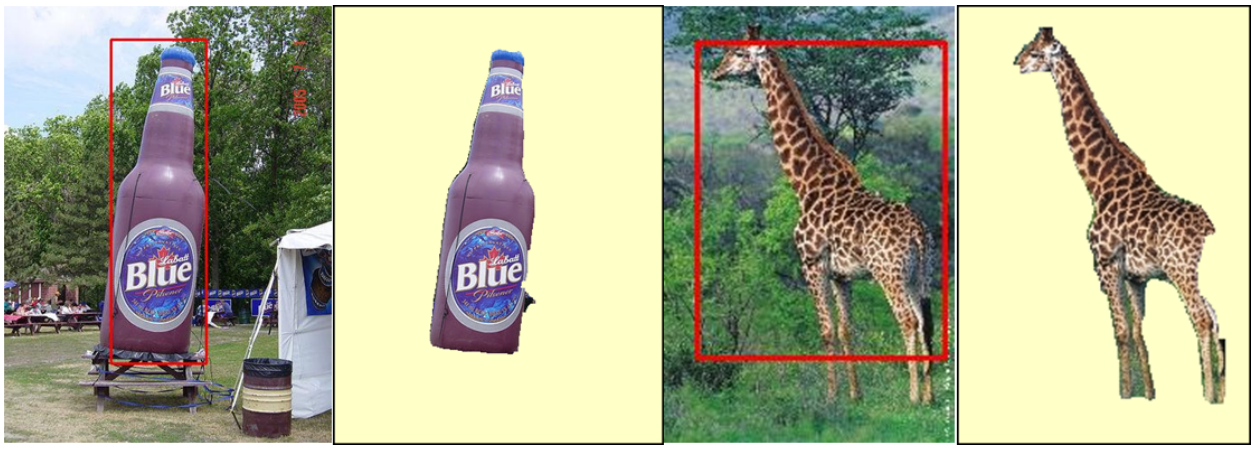


Figure 2.1: Detection and segmentation results on two examples in the ETHZ shape database using our unified approach.

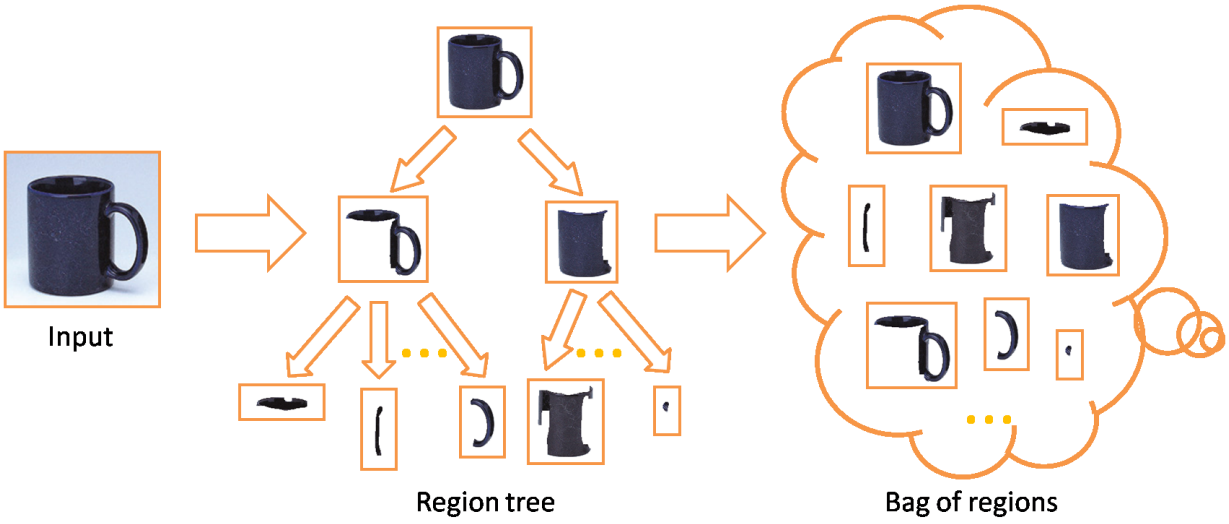
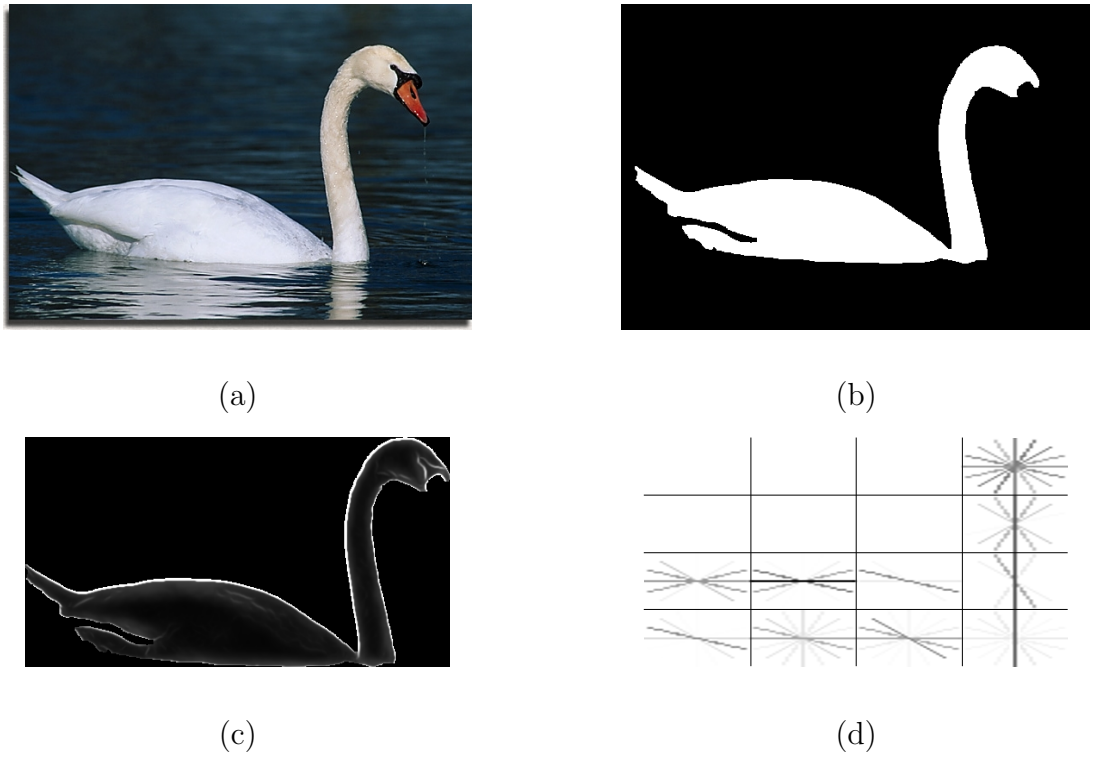


Figure 2.2: The "bag of regions" representation of a mug example. Regions are collected from all nodes of a region tree generated by [6]. Therefore, these regions range in scale from super pixels to the whole image. Note that here "bag" implies discarding tree structure.



(a) (b)

(c) (d)

Figure 2.3: The "contour shape" region descriptor. (a) Original image, (b) A region from the image, (c) gPb representation of the region in (b), (d) Our contour shape descriptor based on (c). Descriptors using other image cues are computed in the same manner.

Region Description

We describe a region by subdividing evenly its bounding box into an $n \times n$ grid, as illustrated in Figure 2.3. In the experiments reported, we use $n = 4$. Each cell encodes information only inside the region. We capture different region cues from the cells, and each type of cue is encoded by concatenating cell signals into a histogram. In this paper, we consider the following region cues:

- Contour shape, given by the histogram of oriented responses of the contour detector gPb [65]
- Edge shape, where orientation is given by local image gradient (computed by convolution with a $[-1\ 0\ 1]$ filter along x- and y-axes). This captures high frequency information (e.g. texture), while gPb is designed to suppress it.
- Color, represented by the L^*, a and b histograms in the CIELAB color space
- Texture, described by texton histograms

Distances between histograms of region cues are characterized using χ^2 measure.

Our region representation has several appealing properties. Firstly, the scale invariant nature of region descriptors enables us to compare regions regardless of their relative sizes. Secondly, background clutter interferes with region representations only mildly compared to interest point descriptors. Thirdly, our region descriptor inherits insights from recent popular image representations such as GIST [72], HOG [23] and SIFT [64]. At the coarsest scale, where the region is the root of the tree, our descriptor is similar to GIST. At the finest scale, when the regions are the leaves of the tree, our representation resembles the SIFT descriptor.

2.3 Discriminative Weight Learning

Not all regions are equally significant for discriminating an object from another. For example, wheel regions are more important than uniform patches to distinguish a bicycle from a mug. Here, we adapt the framework of [37] for learning region weights. Given an exemplar $\mathcal{I}$ containing one object instance and a query $\mathcal{J}$, denote $f_i^{\mathcal{I}}, i = 1, 2, \ldots, M$ and $f_j^{\mathcal{J}}, j = 1, 2, \ldots, N$ their bags of region features.

The distance from $\mathcal{I}$ to $\mathcal{J}$ is defined as:

$$\mathcal{D}(\mathcal{I} \to \mathcal{J}) = \sum_{i=1}^{M} w_i^{\mathcal{I}} d_i^{\mathcal{I}\mathcal{J}} = \langle w^{\mathcal{I}}, d^{\mathcal{I}\mathcal{J}} \rangle, \tag{2.1}$$

where $w_i^{\mathcal{I}}$ is the weight for feature $f_i^{\mathcal{I}}$, and

$$d_i^{\mathcal{I}\mathcal{J}} = \min_j d(f_i^{\mathcal{I}}, f_j^{\mathcal{J}}) \tag{2.2}$$

is the elementary distance between $f_i^{\mathcal{I}}$ and the closest feature in $\mathcal{J}$. Note that the exemplar-to-query distance is asymmetric, i.e., $\mathcal{D}(\mathcal{I} \to \mathcal{J}) \neq \mathcal{D}(\mathcal{J} \to \mathcal{I})$.

In the weight learning stage, supposing $\mathcal{I}$ is an object of category $\mathcal{C}$, we find a pair of $\mathcal{J}$ and $\mathcal{K}$ such that $\mathcal{J}$ is an object of the same category $\mathcal{C}$ and $\mathcal{K}$ is an object of a different category. The learning algorithm enforces the following condition:

$$\begin{aligned} \mathcal{D}(\mathcal{I} \to \mathcal{K}) &> \mathcal{D}(\mathcal{I} \to \mathcal{J}) && (2.3)\\ \Longrightarrow \langle w^{\mathcal{I}}, d^{\mathcal{I}\mathcal{K}} \rangle &> \langle w^{\mathcal{I}}, d^{\mathcal{I}\mathcal{J}} \rangle && (2.4)\\ \Longrightarrow \langle w^{\mathcal{I}}, x^{\mathcal{I}\mathcal{J}\mathcal{K}} \rangle &> 0, && (2.5) \end{aligned}$$

where $x^{\mathcal{I}\mathcal{J}\mathcal{K}} = d^{\mathcal{I}\mathcal{K}} - d^{\mathcal{I}\mathcal{J}}$. Supposing we construct T such pairs for $\mathcal{I}$ from the training set, thus $x_1, x_2, \ldots, x_T$ (we dropped the superscripts for clarity). The large-margin optimization is formulated as follows:

$$\begin{aligned} \min_{w,\xi} \quad & \frac{1}{2} w^T w + C \sum_{i=1}^{T} \xi_i && (2.6)\\ s.t.: \quad & w^T x_i \geq 1 - \xi_i, \xi_i \geq 0, \forall i = 1, 2, \ldots, T && (2.7)\\ & w \succeq 0. && (2.8) \end{aligned}$$

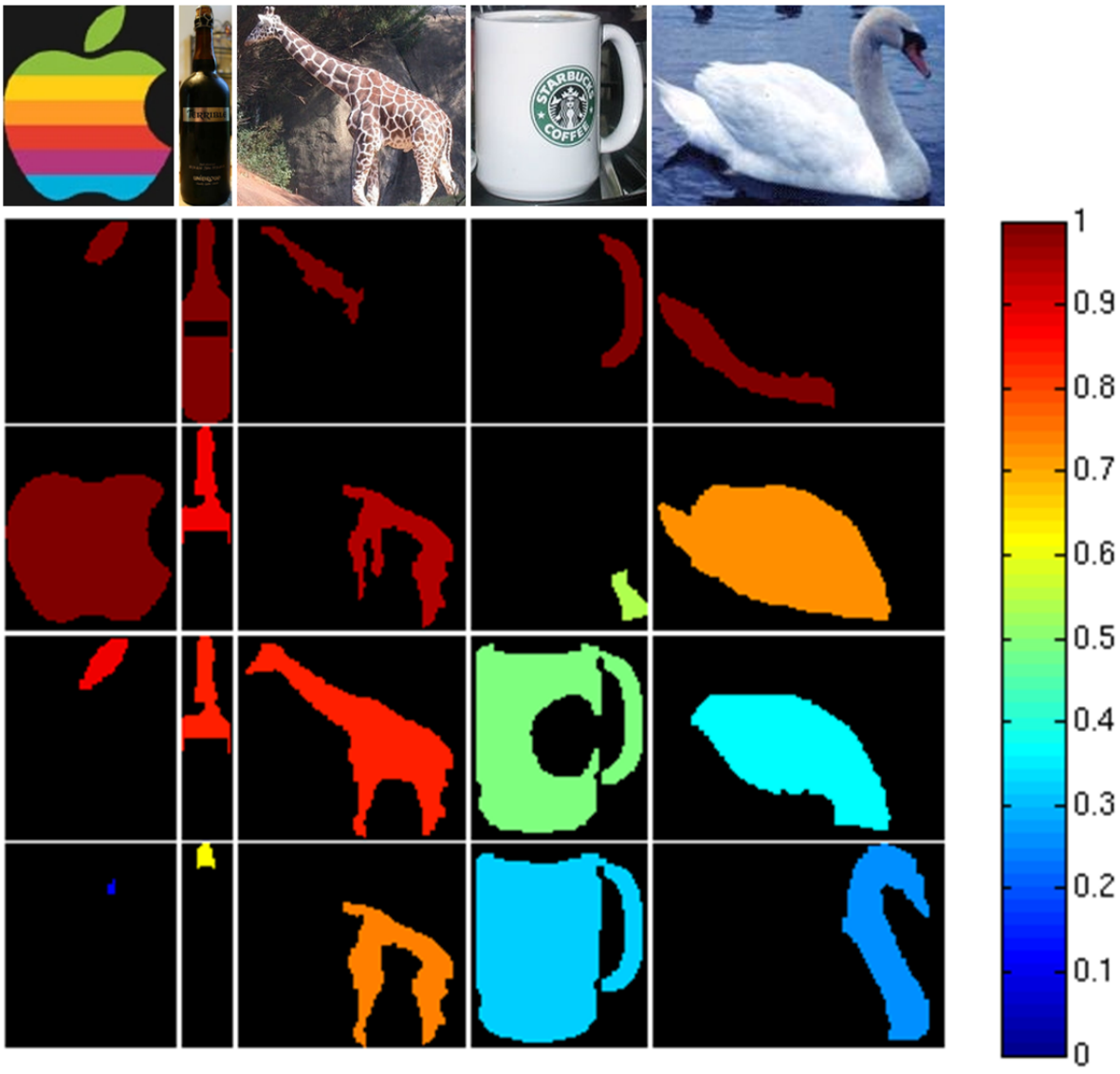


Figure 2.4: Weight learning on regions. For each column, the top image is the exemplar, and the bottom four are regions in order of highest learned weight. Note that the most discriminative regions (leaf and body of the apple logo, handle of the mug) have the highest weights from learning. (best viewed in color)

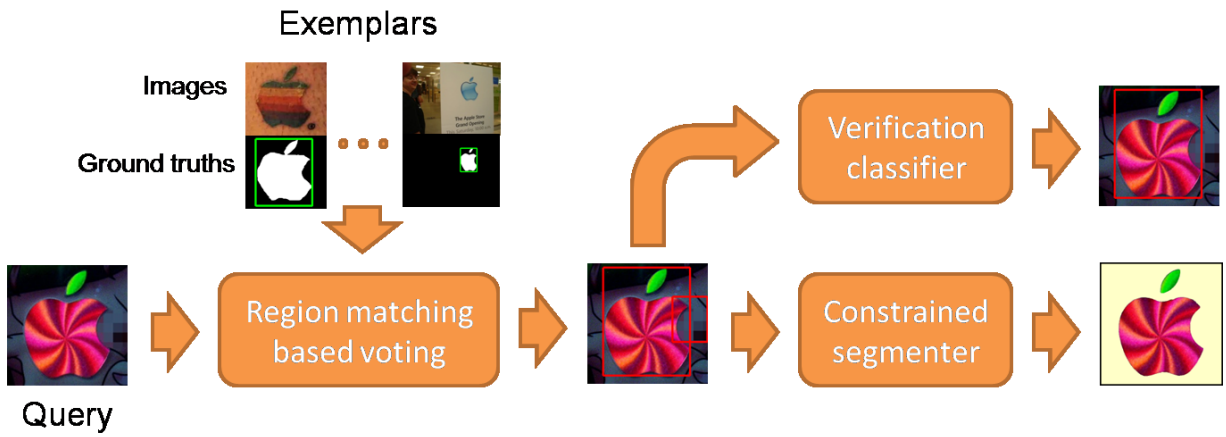


Figure 2.5: The pipeline of our object recognition algorithm consist of three stages. For an input query image, the voting stage casts initial hypotheses of object positions, scales and support based on matched regions from exemplars. These hypotheses are the inputs of the next stages and are refined through a verification classifier and a constrained segmenter, respectively, to obtain final detection and segmentation results. Figure 2.6 describes details of the voting stage, and Figure 2.7 illustrates the segmentation pathway.

When integrating multiple cues for a single region, we learn one weight for each cue. Figure 2.4 shows some examples of learned weights on regions when contour shape cue is used.

As in [37], we model the probability of query $\mathcal{J}$ being in the the same category as exemplar $\mathcal{I}$ by a logistic function:

$$p(\mathcal{I}, \mathcal{J}) = \frac{1}{1 + \exp[-\alpha_{\mathcal{I}} \mathcal{D}(\mathcal{I} \to \mathcal{J}) - \beta_{\mathcal{I}}]} \tag{2.9}$$

where $\alpha_{\mathcal{I}}$ and $\beta_{\mathcal{I}}$ are parameters learned in training.

2.4 Detection and Segmentation Algorithms

Our unified object recognition framework contains three components: voting, verification and segmentation. For a given query image, the voting stage casts initial hypotheses of object positions, scales and support based on region matching. These hypotheses are then refined through a verification classifier and a constrained segmenter, respectively, to obtain final detection and segmentation results. Figure 2.5 depicts the pipeline of our recognition algorithms for the apple logo category. The query image is matched to each apple logo exemplar in the training set, whose ground truth bounding boxes and support masks are both given as inputs. All region weights are determined as in Section 2.3.

Voting

The goal here, given a query image and an object category, is to generate hypotheses of bounding boxes and (partial) support of objects of that category in the image. To achieve it, we use a generalized Hough voting scheme based on the transformation between matched regions as well as the associated objects in the exemplars.

Specifically, given exemplar $\mathcal{I}$, its ground truth bounding box $B^{\mathcal{I}}$ and support mask $M^{\mathcal{I}}$, we match a region $R^{\mathcal{I}}$ in $\mathcal{I}$ to another region $R^{\mathcal{J}}$ in query $\mathcal{J}$. Then the vote for the bounding box $\hat{B}$ of the object in $\mathcal{J}$ is characterized by:

$$\theta_{\hat{B}} = \mathcal{T}(\theta_{B^{\mathcal{I}}} |\, \theta_{R^{\mathcal{I}}}, \theta_{R^{\mathcal{J}}}) \tag{2.10}$$

where $\theta = [x, y, s_x, s_y]$ characterizes the center coordinates $[x, y]$ and the scales $[s_x, s_y]$ of a region or bounding box, and $\mathcal{T}$ is some pre-defined transformation function with its parameters derived by the matched regions $\theta_{R^{\mathcal{I}}}$ and $\theta_{R^{\mathcal{J}}}$.

A voting score is also assigned to each box by combining multiple terms:

$$S_{vot}(\hat{B}) = \tilde{w}_{R^{\mathcal{I}}} \cdot g(d_{R^{\mathcal{I}}}, d_{R^{\mathcal{J}}}) \cdot h(R^{\mathcal{I}}, R^{\mathcal{J}}) \tag{2.11}$$

where $\tilde{w}_{R^{\mathcal{I}}}$ is the learned weight of $R^{\mathcal{I}}$ after normalization, $g(d_{R^{\mathcal{I}}}, d_{R^{\mathcal{J}}})$ characterizes similarity between descriptors $d_{R^{\mathcal{I}}}$ and $d_{R^{\mathcal{J}}}$, and $h(R^{\mathcal{I}}, R^{\mathcal{J}})$ penalizes region shape differences between two regions.

In general, $\mathcal{T}$ in Eqn.2.10 can be any given transformation function. In our experiments, we restrict our transformation model to allow only translation and scaling in both x- and y-axes. Thus, in the x-direction:

$$x^{\hat{B}} = x^{R^{\mathcal{J}}} + (x^{B^{\mathcal{I}}} - x^{R^{\mathcal{I}}}) \cdot s_x^{R^{\mathcal{J}}} / s_x^{R^{\mathcal{I}}} \tag{2.12}$$

$$s_x^{\hat{B}} = s_x^{B^{\mathcal{I}}} \cdot s_x^{R^{\mathcal{J}}} / s_x^{R^{\mathcal{I}}} \tag{2.13}$$

and same equations apply to the y-direction. Figure 2.6 illustrates such generalized Hough voting based on a pair of matched regions.

Eqn.2.11, 2.12 and 2.13 summarize bounding box voting between one pair of matched regions. An early rejection is applied to the voted box either if its voting score is too low or if the box is (partially) outside the image. For all matched regions between a query $\mathcal{J}$ and all exemplars of one category, we generate a set of bounding boxes accordingly for objects of that category in $\mathcal{J}$ for each pair of regions. Finally, we cluster these bounding boxes by a mean-shift [21] algorithm in the feature space θ_B. Here, we favor mean-shift over other clustering methods because it allows adaptive bandwidth setting for different clusters. Thus, two large bounding boxes are more likely to merge than two small boxes if they differ in the same amount in the feature space.

One main advantage of this voting algorithm based on region matching is that it can recover the full support of an object if only a small fraction of that object (e.g., the leaf of the apple logo or the handle of the mug) is matched. It gives not only position but also reliable scale estimation of the bounding boxes. It also allows for aspect ratio deformation of bounding boxes during transformation.

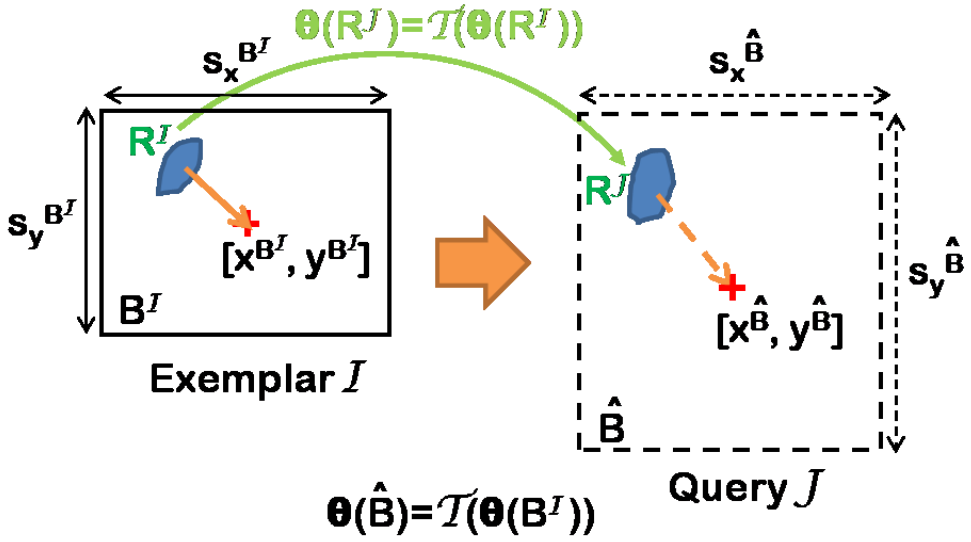


Figure 2.6: Voting stage. This shows a Hough voting scheme based on region matching using a specific transformation function. $\theta = [x, y, s_x, s_y]$ includes the center coordinates $[x, y]$ and the scales $[s_x, s_y]$ of a bounding box. $\mathcal{T}$ transforms a ground truth bounding box $B^{\mathcal{I}}$ of $R^{\mathcal{I}}$ to a new bounding box $\hat{B}$ of $R^{\mathcal{J}}$ based on matching between $R^{\mathcal{I}}$ and $R^{\mathcal{J}}$. This transformation provides not only position but also scale estimation of the object. It also allows for aspect ratio deformation of bounding boxes.

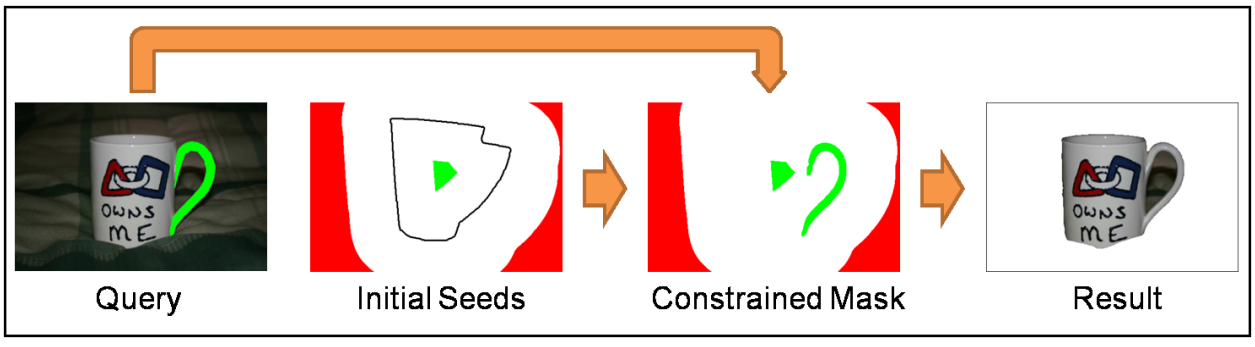


Figure 2.7: Segmentation stage. The initial seeds (green for object and red for background) are derived from transformation of the exemplar mask (with black boundary). The constrained mask is a combination of the seeds and the matched part (mug handle in this case). Note that our method is able to recover the complete object support from one of its parts.

Verification

A verification classifier is applied to each bounding box hypothesis from voting. In general, any object model, *e.g.*, [32, 66], can be applied to each hypothesis. However, in order to fully exploit the use of region representation, we follow the method of [37] using the region weights derived in Section 2.3.

The verification score of a bounding box $\hat{B}$ with respect to category C is defined as the average of the probabilities of $\hat{B}$ to all exemplars of category C:

$$S_{ver}(\hat{B}) = \frac{1}{N}\sum_{i=1}^{N} p(\mathcal{I}_{c_i}, \hat{B}) \tag{2.14}$$

where $\mathcal{I}_{c_1}, \mathcal{I}_{c_2}, \ldots, \mathcal{I}_{c_N}$ are all exemplars of category C, and $p(\mathcal{I}_{c_i}, \hat{B})$ are computed using Eqn.2.9. The overall detection score $S_{det}(\hat{B})$ of $\hat{B}$ for category C is a combination of the voting score $S_{vot}(\hat{B})$ and the verification score $S_{ver}(\hat{B})$, for instance, the product of the two:

$$S_{det}(\hat{B}) = S_{vot}(\hat{B}) \cdot S_{ver}(\hat{B}) \tag{2.15}$$

Segmentation

The segmentation task we consider is that of precisely extracting the support of the object. It has been addressed in the past by techniques such as OBJ CUT [50]. In our framework, the region tree is the result of bottom-up processing; top-down knowledge derived from the matched exemplar is used to mark some of the leaves of the region tree as definitely belonging to the object, and some others as definitely background. We propagate these labels to the rest of the leaves using the method of [4], thus getting the benefit of both top-down and bottom-up processing.

More precisely, let $\mathcal{I}$, $M^{\mathcal{I}}$ and $B^{\mathcal{I}}$ be the exemplar, its ground truth support mask and bounding box, respectively. Then, for a region $R^{\mathcal{I}}$ in $\mathcal{I}$ and one of its matching region $R^{\mathcal{J}}$ in the query image $\mathcal{J}$, we compute $\mathcal{T}(M^{\mathcal{I}})$, the transformation of the ground truth mask $M^{\mathcal{I}}$ on $\mathcal{J}$. $\mathcal{T}(M^{\mathcal{I}})$ provides an initial top-down guess for the location, scale and shape of the object in $\mathcal{J}$. Its complement provides the top-down guess for the background. Since we do not want to have the segmentation be completely determined by these top-down guesses, we allow for a zone of "don't know" pixels in a fixed neighborhood of the boundary of the transformed exemplar mask, and consider as the priors for object and background only pixels greater than a given Euclidean distance from the boundary of the transformed ground truth mask $\mathcal{T}(M^{\mathcal{I}})$. Since we have the constraint that the whole matched region $R^{\mathcal{J}}$ must be part of the object, we union this with the object mask to produce the "constrained mask".

Thus, we construct a segment $\mathcal{M}$ on the query by using both the exemplar mask and the low-level information of the query image, as illustrated in Figure 2.7. As an early rejection test, we compute the overlap between $\mathcal{M}$ and the transformed mask $\mathcal{T}(M^{\mathcal{I}})$, and discard it if the score is low.

We also assign a score $S_{seg}(\mathcal{M})$ to $\mathcal{M}$ based on matched regions $R^{\mathcal{I}}$ and $R^{\mathcal{J}}$:

$$S_{seg}(\mathcal{M}) = \tilde{w}_{R^{\mathcal{I}}} \cdot g(d_{R^{\mathcal{I}}}, d_{R^{\mathcal{J}}}) \tag{2.16}$$

where $\tilde{w}_{R^{\mathcal{I}}}$ and $g(d_{R^{\mathcal{I}}}, d_{R^{\mathcal{J}}})$ are defined in Section 2.4. Thus, we define the confidence map of $\mathcal{J}$ to $\mathcal{I}$ based on $R^{\mathcal{I}}$ as the maximal response of each region in $\mathcal{J}$. The final confidence map for $\mathcal{J}$ for a given category is the double summation of these confidence maps over all regions in $\mathcal{J}$, and over all exemplars of that category.

2.5 Experimental Results

We evaluate our object recognition method on the ETHZ shape and the Caltech 101 databases.

ETHZ Shape

The ETH Zurich shape database (collected by V. Ferrari et. al. [35]) consists of five distinctive shape categories (applelogos, bottles, giraffes, mugs and swans) in a total of 255 images. It is a challenging database because target objects appear over a wide range of scales and locations (see Figure 2.11). In particular, we mark object support in the images as ground truth masks for our segmentation task.

Initially, we construct region trees for images. This gives on average $100 \sim 200$ regions per image. Since color and texture cues are not very useful in this database, we only use gPb-based contour shape cues as region features. In the weight learning stage, we construct exemplar images and their similar/dissimilar pairs in the following way: we take the bounding boxes of objects in training as exemplars. For each exemplar, similar instances are the bounding boxes containing objects of the same category as the exemplar, and dissimilar instances are the ones containing objects of different categories as well as a collection of background regions, all in the training set.

In the voting stage, we choose the functions in Eqn.2.11 as:

$$g(d_{R^{\mathcal{I}}}, d_{R^{\mathcal{J}}}) = \max\{0, 1 - \sigma \cdot \chi^2(d_{R^{\mathcal{I}}}, d_{R^{\mathcal{J}}})\} \tag{2.17}$$

$$h(R^{\mathcal{I}}, R^{\mathcal{J}}) = \mathbf{1}[\alpha \leq \mathrm{Asp}(R^{\mathcal{I}})/\mathrm{Asp}(R^{\mathcal{J}}) \leq 1/\alpha] \tag{2.18}$$

where $\chi^2(\cdot)$ specifies the chi-square distance, and $\mathrm{Asp}(R)$ is the aspect ratio of the bounding box of R. The last equation enforces aspect ratio consistency between matched regions. In the experiment, we use $\sigma = 2$ and $\alpha = 0.6$.

We split the entire set into half training and half test for each category, and the average performance from 5 random splits is reported. This is consistent with the implementation in [34] which reported the state-of-the-art detection performance on this database. Figure 2.8 shows our comparison to [34] on each of the categories. Our method significantly outperforms [34] on all five categories, and the average detection rate increases by 20% ($87.1 \pm 2.8\%$ with

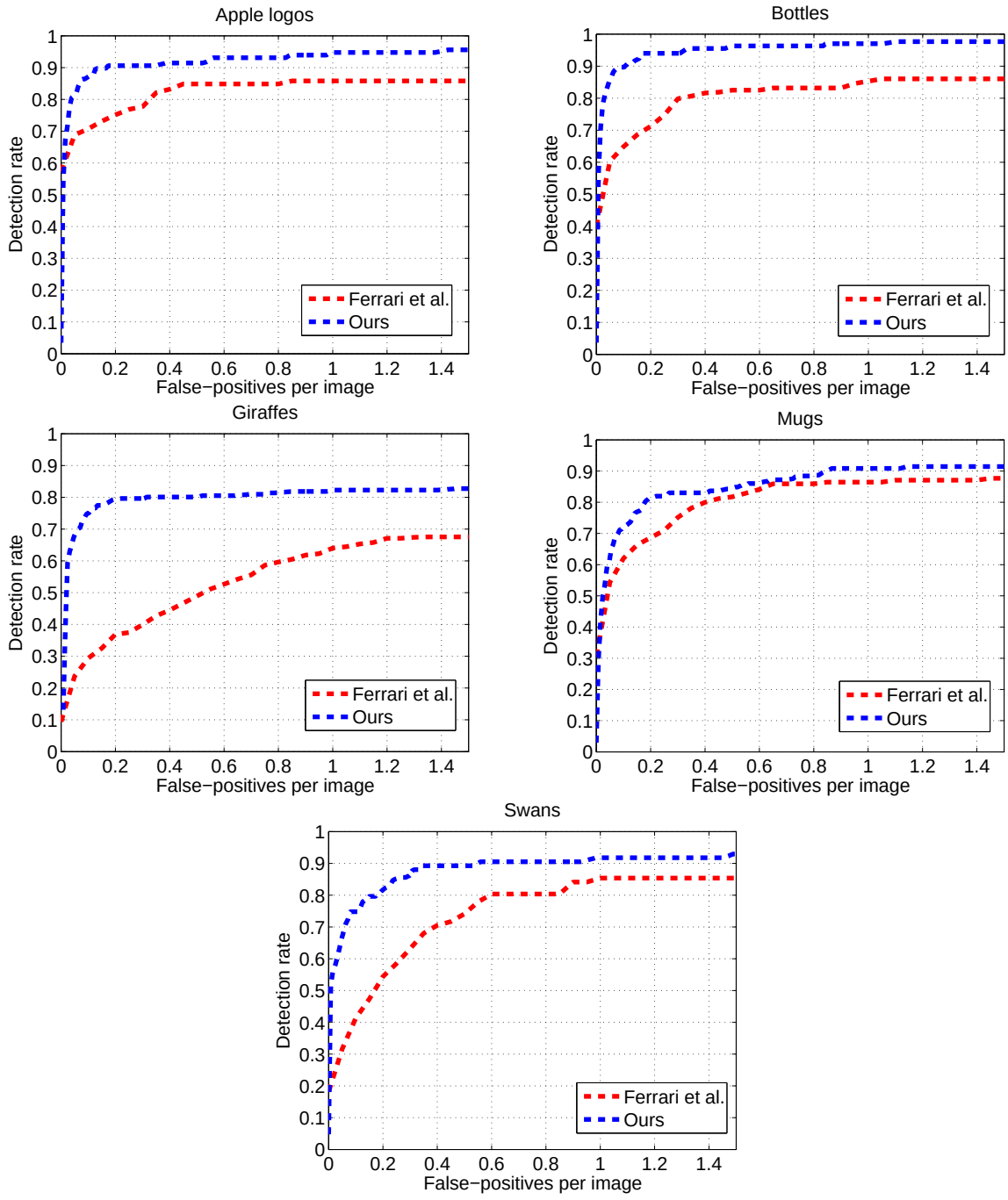


Figure 2.8: Comparison of detection performance with Ferrari et al. [34] on the ETHZ shape database. Each plot shows the detection rate as a function of false positives per image (FPPI) under the PASCAL criterion (a detected bounding box is considered correct if it overlaps $\geq 50\%$ "intersection over union" with the ground truth bounding box). Our method significantly outperforms theirs over all five categories at every FPPI point between $[0, 1.5]$.

Categories	Voting only	Verify only	Combined
Applelogos	87.2 ± 9.0	85.4 ± 5.3	90.6 ± 6.2
Bottles	93.0 ± 3.0	93.2 ± 5.4	94.8 ± 3.6
Giraffes	79.4 ± 1.3	73.6 ± 5.5	79.8 ± 1.8
Mugs	72.6 ± 12.0	81.4 ± 5.4	83.2 ± 5.5
Swans	82.2 ± 10.0	80.8 ± 9.7	86.8 ± 8.9
Average	82.9 ± 4.3	82.9 ± 2.8	$\mathbf{87.1 \pm 2.8}$

Table 2.1: Object detection results in ETHZ shape. Detection rates (%) at 0.3 FPPI based on only voting scores, only verification scores, and products of the two are reported, for each individual category and the overall average over 5 trials.

Categories	Bounding Box	Segments
Applelogos	50.2 ± 7.7	77.2 ± 11.1
Bottles	73.0 ± 2.6	90.6 ± 1.5
Giraffes	34.0 ± 0.7	74.2 ± 2.5
Mugs	72.2 ± 5.1	76.0 ± 4.4
Swans	28.8 ± 4.2	60.6 ± 1.3
Average	51.6 ± 2.5	$\mathbf{75.7 \pm 3.2}$

Table 2.2: Object segmentation results in ETHZ shape. Performance (%) is evaluated by pixel-wise mean Average Precision (AP) over 5 trials. The mean APs are computed both on the bounding boxes obtained in Section 2.4, and the segments obtained in Section 2.4.

Categories	Sld. Windows	Regions	Bnd. Boxes
Applelogos	$\sim 30,000$	115	3.1
Bottles	$\sim 1,500$	168	1.1
Giraffes	$\sim 14,000$	156	6.9
Mugs	$\sim 16,000$	189	5.3
Swans	$\sim 10,000$	132	2.3

Table 2.3: A comparison of the number of sliding windows, regions, and bounding boxes that need to be considered for different categories in ETHZ shape. The number of regions for each category is the average number of regions from images of that category. The number of bounding boxes is the average number of votes from Section 2.4 that need to obtain full recall of objects. The number of sliding windows is estimated in the Appendix.

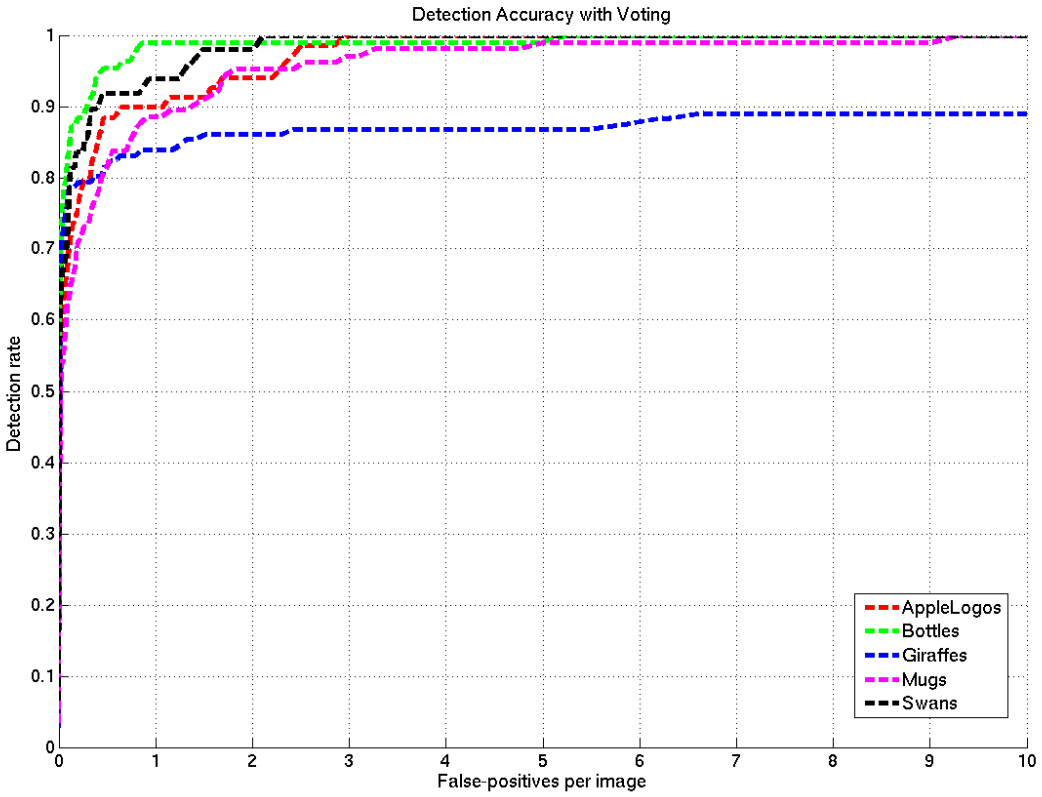


Figure 2.9: On ETHZ, detection rate after voting only. We achieve almost full recall of object bounding boxes in test images at a low FPPI rate of 3-5 (except for giraffes in which the system suffers from inconsistent hand labeling of ground truth bounding boxes).

respect to their 67.2%) at false positive per image (FPPI) rate of 0.3 under the PASCAL criterion. Detection rates on individual categories are listed in Table 2.1.

We also evaluate segmentation performance on each of the 5 categories using mean average precision (AP) of pixel-wise classification. AP is defined by the area underneath the recall-precision curve. Table 2.2 shows the precision accuracies. The overall mean AP on the object segments using our constrained segmentation algorithm achieves $75.7 \pm 3.2\%$, significantly higher than on the bounding boxes from voting. Examples of object detection and segmentation results are shown in Figure 2.11.

Table 2.3 compares the number of sliding windows, regions, and bounding boxes that need to be considered for different categories. We show that our voting scheme obtains 3-4 orders of magnitude reduction on the number of windows compared to the standard sliding window approach.

Image cues	5 train	15 train	30 train
(R) Contour shape	41.5	55.1	60.4
(R) Edge shape	30.0	42.9	48.0
(R) Color	19.3	27.1	27.2
(R) Texture	23.9	31.4	32.7
(R) All	40.9	59.0	65.2
(P) GB	42.6	58.4	63.2
(R) Contour shape+(P) GB	44.1	**65.0**	**73.1**
(R) All + (P) GB	45.7	64.4	72.5

Table 2.4: Mean classification rate (%) in Caltech 101 using individual and combinations of image cues. (R) stands for region-based, and (P) stands for point-based. (R)All means combining all region cues (Contour shape+Edge shape+Color+Texture). We notice that cue combination boosts the overall performance significantly.

Caltech-101

The Caltech-101 database (collected by L. Fei-Fei et. al. [31]) consists of images from 101 object categories (excluding the background class). The significant variation in intra-class pose, color and lighting makes this database challenging. However, since each image contains only a single object, usually large and aligned to the center, we bypass the voting step and consider the entire image as the bounding box of the object. Thus, we use this database to benchmark only our verification step.

We follow the standard approach for evaluation. For each category, we randomly pick 5, 15 or 30 images for training and up to 15 images in a disjoint set for test. Each test image is assigned a predicted label, and mean classification rate is the average of the diagonal elements of the confusion matrix.

To exploit multiple image cues, we extract four types of region descriptors (two types of shape, color and texture, all described in Section 2.2), as well as one point descriptor (Geometric Blur or GB [11]). Table 2.4 lists the mean classification rates with different combinations of these image cues. We observe a performance gain (from 55.1% to 59.0% under 15 training) by combining different region cues in our method. In addition, a second and significant boost in performance is obtained by combining region contour shape with point GB cues (from 58.4% to 65.0% under 15 training). This boost illustrates that region based descriptors complements conventional point based descriptors (e.g. SIFT) in recognition. Our method achieves competitive performance in this database in comparison with other recently published approaches in Figure 2.10.

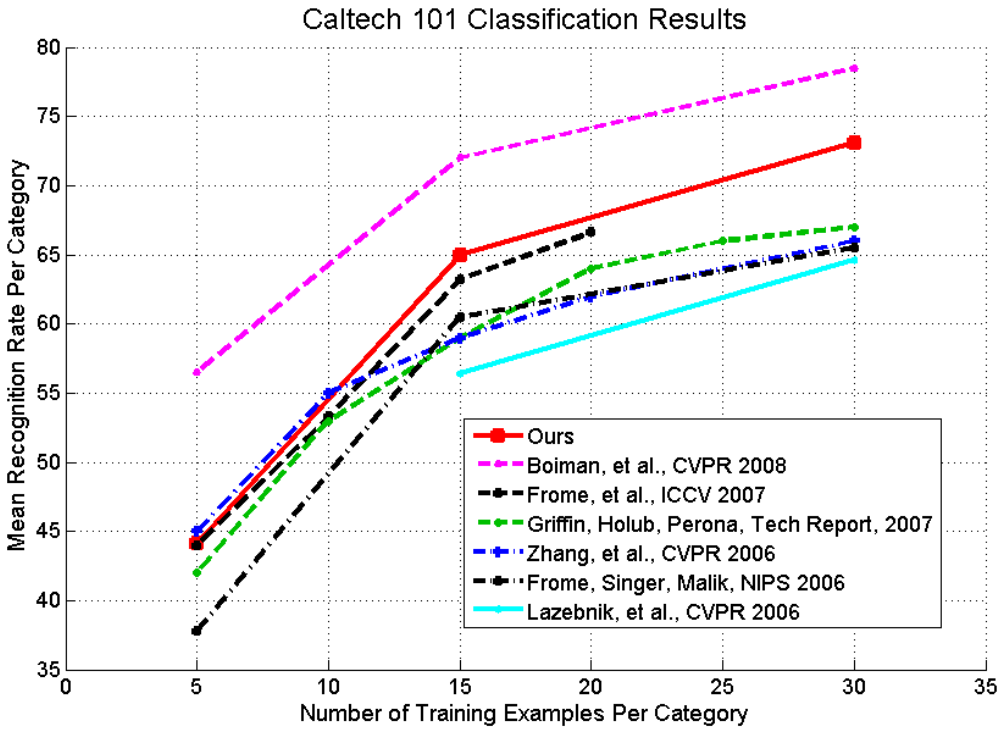


Figure 2.10: Mean recognition rate (%) over number of training images per category in Caltech 101. With 15 and 30 training images per category, our method outperforms [38, 40, 102, 37] and [56] but [13].

Figure 2.11: Detection and segmentation results in the ETHZ shape database.

2.6 Conclusion

In this paper, we have presented a unified framework for object detection, segmentation, and classification using regions. Building on a novel region segmentation algorithm which produces robust overlaid regions, we have reported state-of-the-art detection performance on the ETHZ shape database, and competitive classification performance on the Caltech 101 database. We have further shown that (1) cue combination significantly boosts recognition performance; (2) our region-based voting scheme reduces the number of candidate bounding boxes by orders of magnitude over standard sliding window scheme due to robust estimation of object scales from region matching.

Appendix

We compute the optimal sliding window parameter choices with respect to the ground truth labeling of the test set in ETHZ shape. This gives us an estimate of the total number of candidates a sliding window classifier would need to examine in order to achieve full recall. To this end, we first compute relative scales of objects with respect to image sizes in the test set. We denote the minimum and maximum scales as S_{min}, and S_{max}. So $0 < S_{min} < S_{max} < 1$. Next, we assume that the minimum span between neighboring windows in each image axis is a quarter of the minimum scale. Then for each level of window scale, we have roughly $1/(S_{min}/4)^2$ candidate locations. As for searching over scales, we make a second assumption that the neighboring levels are 1/8 octave apart. Then the number of scales needed to cover the range of $[S_{min}, S_{max}]$ is $8\log_2(S_{max}/S_{min})$. So if we ignore aspect ratio change of objects, the estimate of the number of windows N becomes

$$N = 1/(S_{min}/4)^2 \cdot 8\log_2(S_{max}/S_{min}) \quad (2.19)$$

$$= 128\log_2(S_{max}/S_{min})/S_{min}^2 \quad (2.20)$$

Chapter 3

Context as Region Ancestry

3.1 Introduction

The role of context in visual perception has been studied for a long time in psychology [75, 12, 45]. Visual context can be defined by the scene embedding a particular object [75], and by the semantic relations among different objects [12]. Thus, contextual information for a table is provided by the dining room where it lies, but also by the presence of chairs around it and dishes on its top.

In the computer vision community, despite early attempts as in [91], the importance of context has only recently been widely acknowledged. Its operationalization often relies on the definition of a holistic descriptor of the image. For instance, the seminal work of Oliva and Torralba [72] aimed at capturing the "gist" of the scene. In the case of multi-class segmentation, many recent approaches express relations among objects as pairwise potentials on a probabilistic model [43, 51, 90, 78, 99, 9, 39, 49, 77].

However, contextual cues are naturally encoded through a "partonomy" of the image, the hierarchical representation relating parts to objects and to the scene. In this paper, we argue in favor of using the hierarchy of regions produced by a generic segmentation method in order to model context.

Concretely, given an image, we first construct a region tree using the hierarchical segmentation algorithm of [6]. The leaves of the tree are the regions of the finest segmentation considered, the root is the entire image and the nodes represent regions ordered by inclusion.

We consider the leaves of the tree as elementary units and represent each unit by features on the set of regions on the path linking it to the root. Borrowing a metaphor from genealogy, we call this set of regions the *ancestral set* of the leaf, with the noteworthy difference that ancestors are in our case *spatial*, rather than temporal.

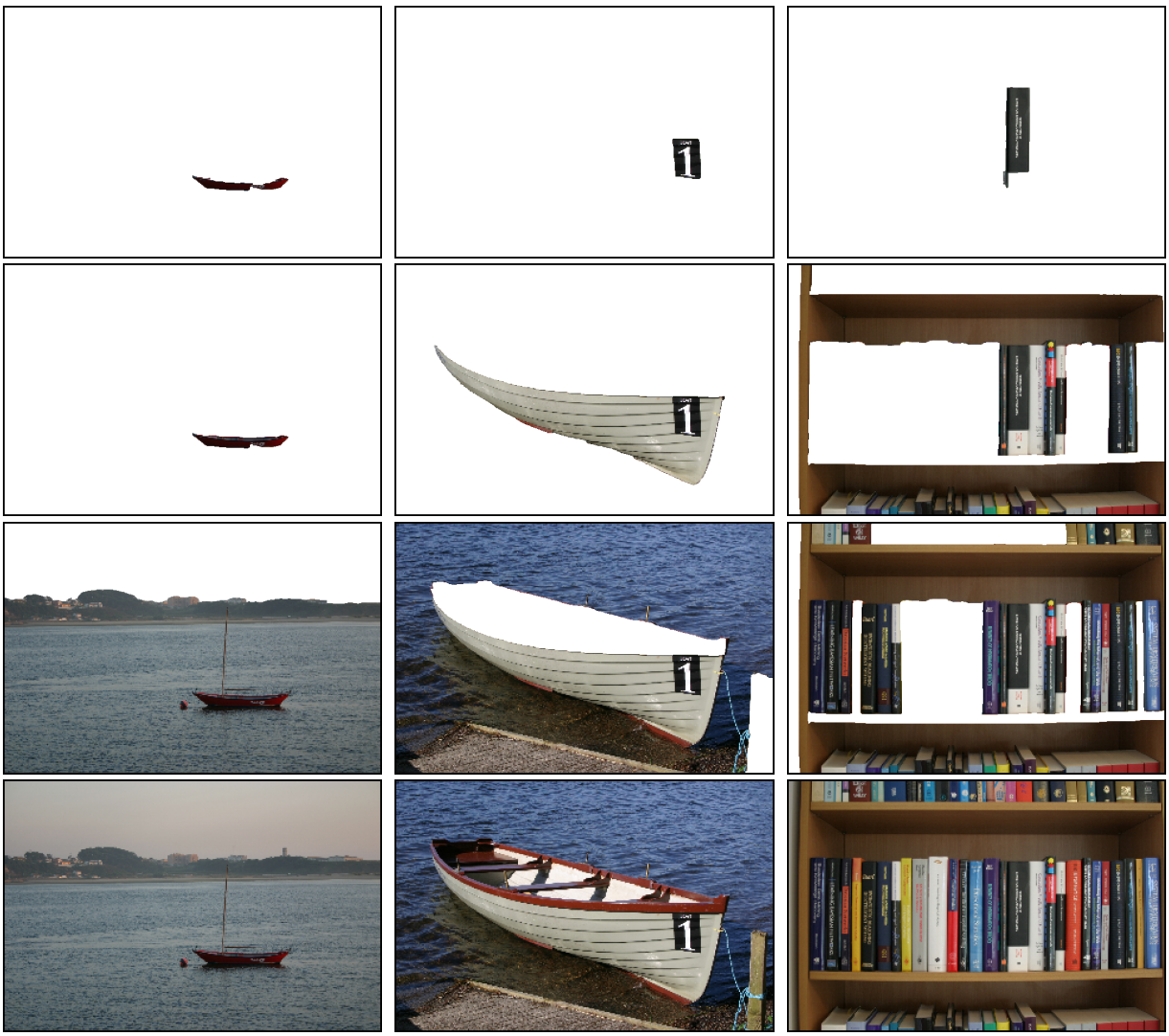


Figure 3.1: Region ancestry encodes discriminative parts, objects and scenes. Each column shows one leaf of the segmentation tree (top) and three of its ancestors. We measure similarity between leaves by comparing their ancestors and learn their importance on a discriminative framework.

Figure 3.2: Example of ancestral set. From left to right: • Schematic representation of segmentation tree and example of ancestral set; the leaves are in this case the nodes $\{A, B, C, D, E, F, G, H\}$ and the ancestral set of leaf D is $AS(D) = \{D, J, M, O\}$. • Example of ancestral set on a real image. • Original image. • Hierarchical boundaries produced by the segmentation algorithm. • Finest segmentation, containing the leaves of the tree.

By progressively enlarging the window of analysis, the ancestral set of a leaf region encodes information at all scales, ranging form local parts of objects, to the whole scene. When making a decision about the category of a leaf or the scene, we expect discriminative features to be different at each level. For instance, in the example of Fig. 3.1, shape information may be informative for the immediate ancestors of the leaf, but color and texture may be more important at the scene level. In order to take into account such differences, we define the dissimilarity between leaves as a weighted sum of distances among ancestral features and learn their importance in a discriminative framework.

In order to illustrate the power of region ancestries as a model of context, we address two different tasks: class-specific segmentation, where a category label is assigned to each pixel in the image, and scene classification, where the whole image is labeled according to the type of scene depicted. We report results on the MSRC dataset for the former and on the MIT scene dataset for the latter, obtaining competitive performance on both tasks. Our experiments show that including contextual information provides a significant improvement of 20% in performance on the MSRC dataset, as shown in Table 3.1.

The rest of the paper is organized as follows. Section 2 reviews previous work. Section 3 introduces the building blocks of our approach. Section 4 describes the learning frameworks considered. Section 5 presents our method for classifying leaves based on ancestry. Section 6 is devoted to experiments. Finally, Section 7 contains some concluding remarks.

3.2 Related Work

A large number of recent approaches to multi-class segmentation address the problem using a probabilistic framework and, more specifically, conditional Markov random fields (CRFs) [43, 51, 90, 78, 99, 9, 39, 49, 77].

These methods reason on a graph, where the nodes are usually entities extracted from the image, e.g. pixels or patches [43, 90], superpixels [39, 9], or regions from multiple segmentations [49]. Context is in this case understood as relations between entities and modeled by a pairwise potential between nodes. This term is often defined between adjacent entities and is used to enforce spatial consistency on the labels [43, 90]. Other approaches consider a fully connected graph and can therefore model larger-range connections [51, 77, 78]. Some recent methods apply CRFs after an initial estimation of the labels and are therefore able to express more semantic relations. For example, in [78], the pairwise potential captures the co-occurrence of categories in the same image, which is learned, either from the training data, or from external resources. Gould et al. [39] introduce a local feature that encodes relative location among categories. Similarly, [51] propose a two-layer CRF, the first one acting on pixels and the second one modeling relations between categories.

A second type of methods introduce contextual cues by considering a holistic image descriptor or a prior on the categories present in the image. In [89], this prior is provided by a global image classifier. In [76], the entities are intersections of regions from multiple segmentations and contextual information is included by describing each region with descriptors computed on the region mask and on the whole image.

Context has also been used to improve object detection. In [71], a holistic descriptor is used to narrow the search space of an object detector to a set of likely locations. Heitz and Koller [44] improve the detection of rigid objects by learning spatial relations with amorphous categories. A different approach for modeling context is the work of Hoeim et al. [46], who estimate the three dimensional layout of a scene by labeling pixels according to surface orientations. A similar problem is addressed by Sudderth et al. [93], by using a hierarchical version of Dirichlet porcesses. Graphical models are also used in [60] in order to infer scene categories.

Our approach differs significantly from previous models of context. First, we reason on a hierarchy of regions produced by a generic segmentation engine. Second, our model emphasizes the relation of inclusion, rather than adjacency or co-occurrence.

3.3 Comparing Leaves by Ancestry

In this section, we first define and describe how we obtain ancestral sets from a segmentation tree and the method to compare them.

We use the segmentation algorithm of [6], that constructs a region tree starting from a contour detector. This method is a fast and parameter-free generic grouping engine. Furthermore, when applied on the output of the high-quality contour detector gPb [6], it

significantly outperforms all available segmentation approaches, occupying the first place in the Berkeley Segmentation Dataset and Benchmark [36].

The output of the low-level segmenter is an image of weighted boundaries (see Fig. 3.2). When thresholded, the weighted boundary image produces the segmentation corresponding to a uniform cut in the tree.

We define the **ancestral set**, or **ancestry**, of a region R in a tree $\mathcal{T}$ as the set of regions on the path linking R to the root:

$$A(R) = \{R' \in \mathcal{T} \mid R \subseteq R'\} \tag{3.1}$$

The elementary units of our approach are the leaves of a segmentation tree, often referred as *superpixels*. They are the elements of the finest partition of the image considered. Figure 3.2 presents an example. Note that the finest partition can be any level in the hierarchy, depending on the task.

We describe each region node in the tree using various features (e.g. color, texture, shape) and represent each leaf r by the descriptors of its ancestral set, $F(r)$,

$$F(r) = \{f_1, \ldots, f_M\}, \tag{3.2}$$

where $M = |A(r)| \cdot Q$, and Q is the total number of features per region.

In order to measure the similarity between two leaves, we first consider elementary distances between region descriptors of the same type. For simplicity, we use a single notation $d(\cdot, \cdot)$, although they can be different depending on the type of descriptor.

We define the **dissimilarity vector** of a leaf s with respect to a fixed leaf r as:

$$\mathbf{d_r}(s) = [d_r^1(s), \ldots, d_r^M(s)], \tag{3.3}$$

where:

$$d_r^i(s) = \min_{f_j \in F(s)} d(f_i, f_j),\ i = 1, \ldots, M. \tag{3.4}$$

and the comparison is done only among features of the same type.

We then define the **dissimilarity** of s with respect to r as a weighted sum of elementary distances:

$$D_r(s) = \sum_{i=1}^{M} w_i\, d_r^i(s) = \langle \mathbf{w}, \mathbf{d_r}(s) \rangle, \tag{3.5}$$

where $\mathbf{w}$ is the weight vector learned using the framework of the next section. Note that, in general, D is not symmetric, i.e., $D_r(s) \neq D_s(r)$.

3.4 Learning the Importance of Ancestors

As motivated in the introduction, in an ancestry, descriptors of ancestors in various scales contribute differently to the dissimilarity measures. For instance, the shape descriptor could

be the most predominant cue for the body of an aeroplane, whereas its ancestor, the scene of the plane in the sky, could be best described by color. We address this issue by learning a set of weights for the leaf as well as its ancestors from the training data.

We adopt an exemplar-based matching framework for our purpose. Precisely, given an exemplar leaf r of class $C(r)$, and a set of leaves $\{s_1, \ldots, s_N\}$ of classes $\{C(s_1), \ldots, C(s_N)\}$, all from the training set, first we compute the dissimilarity vectors of s_i with respect to r, $\{\mathbf{d_r}(s_1), \ldots, \mathbf{d_r}(s_m)\}$. We also denote $D_r(s_i) = \mathbf{w}^T\mathbf{d_r}$ for each s_i, the dissimilarity of s_i with respect to r.

We introduce and compare three learning approaches in the following, all of which attempt to decrease $D_r(s_i)$ and increase $D_r(s_j)$ for $C(r) = C(s_i)$ and $C(r) \neq C(s_j)$.

Logistic Regression

The first method to find the weight vector $\mathbf{w}$ is to use a binary logistic classifier to learn directly a mapping from the input vector $\mathbf{d_r}(s)$ to a probabilistic output that measures the probabilities of two leaves having the same class:

$$P(C(s) = c|C(r) = c, \mathbf{w}) = \frac{1}{1 + \exp\{-\mathbf{w^T}\mathbf{d_r}(s)\}} \tag{3.6}$$

Given $\mathbf{d_r}(s_i), i = 1, 2, \ldots, N$, the L_2-regularized large-margin optimization is formulated as follows:

$$\min_{\mathbf{w}} \ \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N} \log(1 + \exp(-y_i\mathbf{w^T}\mathbf{d_r}(s_i))) \tag{3.7}$$

Where

$$y_i = 2 \cdot \mathbf{1}_{[C(s_i)=C(r)]} - 1, i = 1, 2, \ldots, N \tag{3.8}$$

Logistic regression is also used in the two other methods below but as a post-processing, in order to normalize a raw dissimilarity to the range $[0, 1]$. More details will be stated in Section 5.

Support Vector Machines

A second option is to consider a binary linear support vector machine, using $\mathbf{d_r}(s_i), i = 1, 2, \ldots, N$ as feature vectors and y_i's defined in Equation (3.8) as binary labels. Compared to the logistic regression classifier, we replace in this case the loss function in (3.7) for a hinge loss:

$$\min_{\mathbf{w},\xi} \ \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N} \xi_i \tag{3.9}$$

$$s.t.: \ \mathbf{w}^T\mathbf{d_r}(s_i) \geq 1 - \xi_i, \ \xi_i \geq 0, \ i = 1, \ldots, N \tag{3.10}$$

Rank Learning

A third approach for learning the importance of ancestors is the framework of [37]. Unlike the linear-SVM approach that enforces a maximum margin between two sets of data, this learning algorithm enforces maximum margins between pairs of data points from two sets.

Again, given the exemplar r, we consider a second leaf s with the same category as r, and a third leaf t belonging to a different category, we have:

$$D_r(t) > D_r(s) \tag{3.11}$$
$$\Rightarrow \langle \mathbf{w}, \mathbf{d_r}(t) \rangle \; > \; \langle \mathbf{w}, \mathbf{d_r}(s) \rangle \tag{3.12}$$
$$\Rightarrow \langle \mathbf{w}, \mathbf{x} \rangle \; > \; 0, \tag{3.13}$$

where $\mathbf{x} = \mathbf{d_r}(t) - \mathbf{d_r}(s)$.

A set of T such triplets, $\{\mathbf{x}_1, \ldots, \mathbf{x}_T\}$, is constructed from $\{s_1, \ldots, s_N\}$. The large-margin optimization is then formulated as follows:

$$\min_{\mathbf{w},\xi} \; \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{T}\xi_i \tag{3.14}$$
$$s.t.: \; \mathbf{w}^T\mathbf{x_i} \geq 1 - \xi_i, \; \xi_i \geq 0, \; \forall i = 1, \ldots, T \tag{3.15}$$
$$\mathbf{w} \succeq 0. \tag{3.16}$$

3.5 Leaf Classification

We use the learned dissimilarities between exemplars and test data for leaf classification. When a test leaf is compared to each one of the exemplar leaves, the associated dissimilarities are not directly comparable because they are derived from different learned weights and thus their values may have different ranges. To address this, we do a second round training for each exemplar by fitting a logistic classifier to the binary training labels and dissimilarities, so that dissimilarities are converted to probabilities. We omit this procedure when the weights are learned using the logistic regression because the optimization automatically returns the probabilities outputs that are directly comparable.

To predict the category label for a leaf, we define the confidence score of the leaf s to a category c as the average of the probabilities from the exemplar leaves of that category. That is:

$$Score(s|c) = \frac{1}{|C|}\sum_{j:C(j)=c} p_j(s) \tag{3.17}$$

Where $p_j(s)$ is the probability of s with respect to exemplar j from the logistic classifier.

Figure 3.3: Example of merging the per category confidence maps into a single segmentation. Top: Original image, initial partition, ground truth and final segmentation. Bottom: Confidence maps for the categories grass, tree, sky and aeroplane. The confidence maps of all the categories are used to produce the final segmentation.

In practice, in order to make the confidence score more robust to outliers, we compute the average by using only the top 50% of the probabilities. The test leaf is assigned to the category label with the largest confidence score.

The final segmentation is then obtained by assigning to each pixel the predicted category label of the leaf where it lies.

In Figure 3.3, we show the confidences maps for four categories and the final segmentation. Each category's confidence map is obtained by assigning the leaf region with the confidence score.

3.6 Experiments

Implementation Details

Since our purpose is to explore the power of our context model, we describe each region in the segmentation tree with standard features.

We represent color by concatenating the marginal histograms in the CIELAB space. Texture is encoded by following the texton approach [58], where filter-bank responses are clustered with the k-means algorithm. In the experiments presented, we consider a codebook of 250 universal textons and 30 bins per color channel.

Shape is encoded by placing an nxn grid on the bounding box of the region and measuring oriented contour energy on each grid cell. We use both gPb and Sobel filters for this purpose. We then concatenate the responses in each cell to obtain a single shape descriptor. In the results below, we consider 8 orientations and $n = 3$, for a total of 144 dimensions.

Additionally, for the regions on an ancestral set, we consider the normalized coordinates of the leaf's centroid as absolute location features. We compare histogram features using χ^2 as elementary distance and location features using Euclidean distance.

We use the linear SVM and logistic regression implementations of LibLinear [30].

Class-specific Segmentation

We conduct experiments on the MSRC database [90], containing 591 images, with objects from 21 categories. In order to compare our results with other methods, we use the standard split of the database and measure performance by computing the average pixelwise classification accuracy across categories. This metric is preferable to the overall classification accuracy because it does not favor the more common categories.

Table 3.1 shows the improvement obtained by using the ancestral set instead of the leaves for the learning frameworks considered. The improvement in performance is important in all the cases. Figure 3.4 presents the confusion matrix of our method. Figure 3.5 presents some qualitative results.

Table 3.2 compares our method against recent approaches. We obtain state-of-the-art performance in 8/21 categories and on average classification accuracy. It's worth noting that [99, 39] report an average performance of 64%; however, these results are not included in Table 3.2 because they were obtained on different splits of the dataset and are therefore not directly comparable.

Scene classification

We test our system on the MIT scene dataset for the scene classification task. The dataset is composed by 2688 images belonging to 8 scene categories. It is divided in 800 images for training, with 100 per category, and 1888 images for testing. In this case, instead of assigning a label to each pixel, we assign to the image the label of the confidence map with highest average value.

Table 3.3 presents the results. We perform comparably to [72], showing again a significant improvement by the use of ancestral sets. Figure 3.6 shows the confusion matrix of our method.

3.7 Conclusions

We introduced a new approach for modeling contextual information by considering the ancestral sets of leaves on a segmentation tree. We validate our approach on multi-class segmentation and scene classification tasks.

In this paper, we have emphasized the role of ancestors but, of course, context is also captured by siblings, uncles, aunts, etc. Extending our framework to consider these relation-

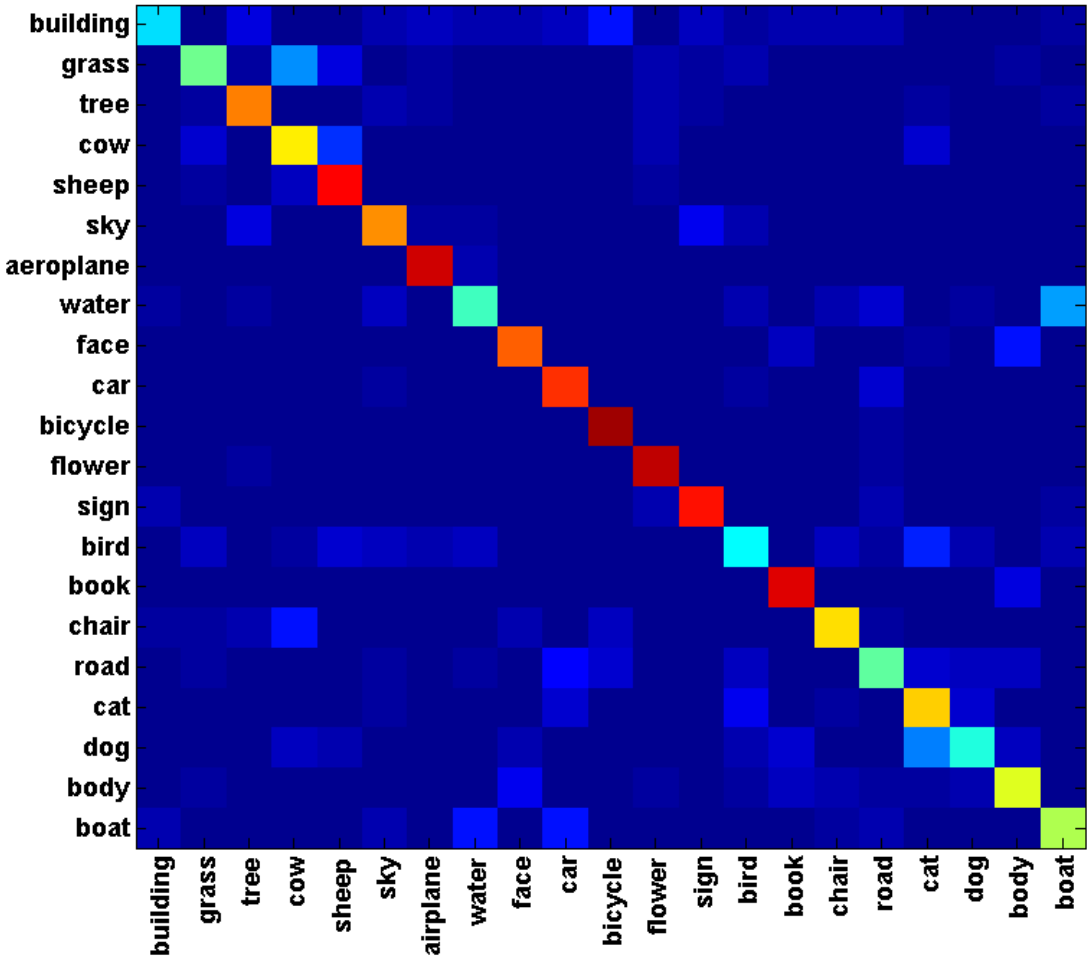


Figure 3.4: Confusion matrix for MSRC dataset. Average performance is **67%**

ships would allow us to exploit co-occurrence or relative location cues between categories [78, 39].

	Leaf Only	Ancestral Set	Improvement
LR	47%	**67%**	**+20%**
SVM	**48%**	62%	+14%
RL	45%	57%	+12%

Table 3.1: Results on MSRC. Comparison of the performance of our method with the different learning approaches considered in Section 4 and by using only the leaves or the full ancestral set. Contextual information provided by the ancestry significantly improves performance in all cases. The best result is obtained with the simple logistic classifier.

Method	Building	Grass	Tree	Cow	Sheep	Sky	Aeroplane	Water	Face	Car	Bicycle	Flower	Sign	Bird	Book	Chair	Road	Cat	Dog	Body	Boat	Average (%)
[90]	62	**98**	**86**	58	50	83	60	53	74	63	75	63	35	19	92	15	**86**	54	19	64	7	58
[9]	**68**	94	84	37	55	68	52	71	47	52	85	69	54	5	85	21	66	16	**49**	44	32	55
[76]	**68**	92	81	58	65	**95**	85	**81**	75	65	68	53	35	23	85	16	83	48	29	48	15	60
[89]	49	88	79	**97**	**97**	78	82	54	**87**	74	72	74	36	24	**93**	51	78	**75**	35	**66**	18	**67**
(L)	34	74	66	49	46	83	56	49	85	34	55	50	44	37	28	16	61	33	35	28	16	47
(AS)	33	47	74	64	86	72	**91**	44	78	**83**	**96**	**93**	**86**	**37**	90	**65**	46	67	41	59	**53**	**67**

Table 3.2: Comparison of our results with other recent methods in MSRC. We obtain the best performance in 8/21 categories, all of them objects. Results reported for our method correspond to weight learning with logistic regression, using only the leaves (L) and the ancestral set (AS).

Method	Building	City	Street	Highway	Coast	Country	Mountain	Forest	Average (%)
[72]	82	**90**	**89**	**87**	79	71	**81**	91	**84**
Ours (L)	89	27	83	44	**80**	67	48	96	67
Ours (AS)	**93**	81	88	64	77	**79**	80	**96**	82

Table 3.3: Results on MIT scene dataset. Using the ancestral set provides a significant boost in performance of 15% for this task. Results reported for our method correspond to weight learning with logistic regression, using only the leaves (L) and the ancestral set (AS).

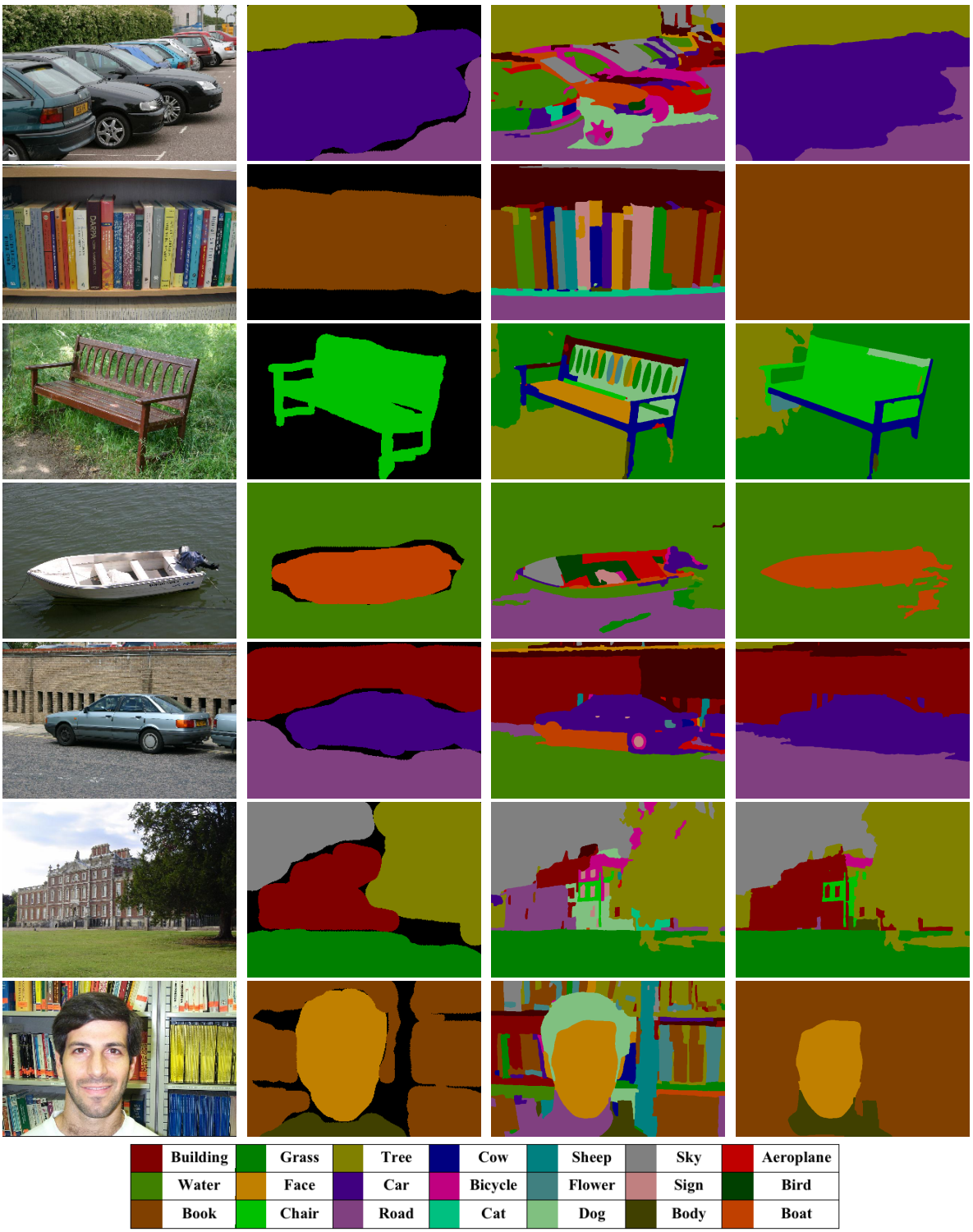


Figure 3.5: Example of results on MSRC dataset. From left to right: • Original image. • Ground truth segmentation. • Results of our method using only the leaves. • Results of our method using the ancestral set.

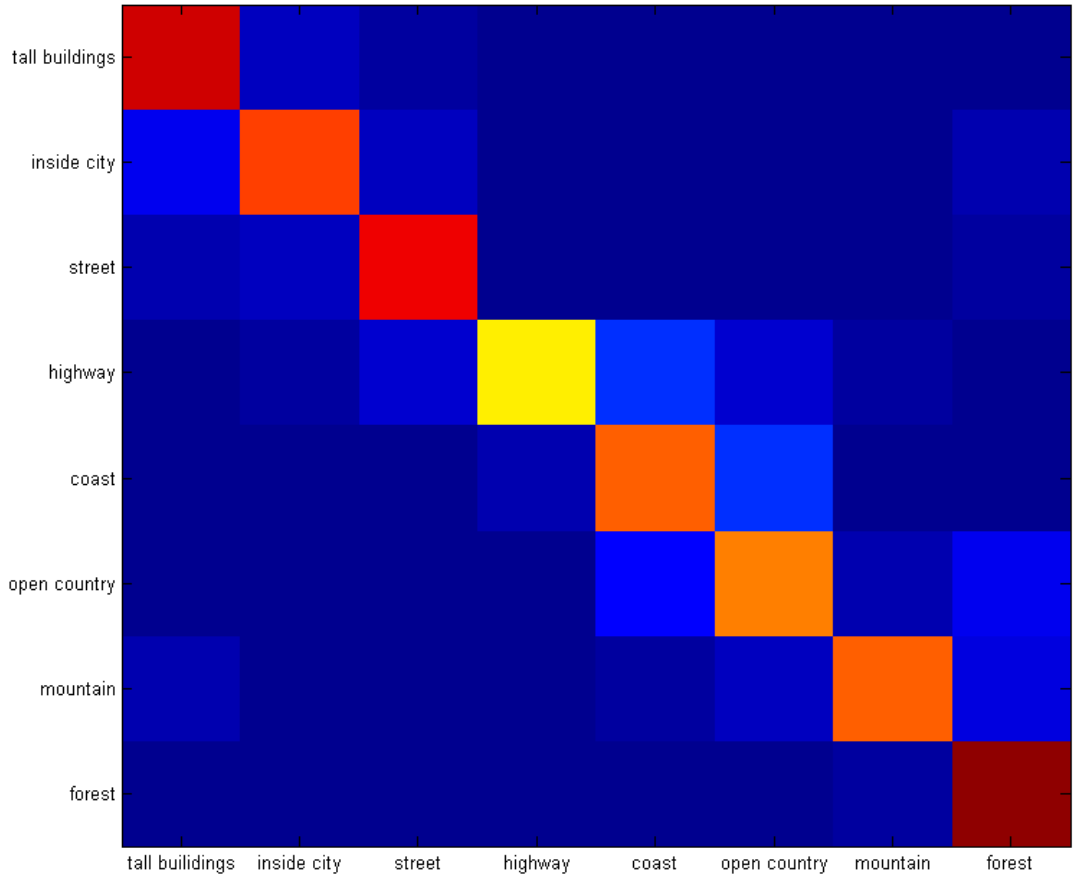


Figure 3.6: Confusion matrix for MIT scene dataset. Average performance is **82%**

Chapter 4

Mixture-of-Templates for Viewpoint Classification

4.1 Introduction

One fundamental property of visual sensing is that it is a projection process from a 3D world to a 2D image plane; much of the 3D information is lost in the projection. How to model and re-capture the 3D information from 2D views has been at the center of the computer vision research. One classical example is the *aspect graphs* of Koenderink and van Doorn [48], where a 3D object is modeled as a collection of inter-connected 2D views.

A complete understanding of objects in a visual scene comprises not only labeling the identities of objects but also knowing their poses in 3D. Most of the recent vision research has been devoted to the recognition problem, where huge progresses have been made: the SIFT matching framework [64] and the HOG models [23, 33] are good representatives of how much object recognition capabilities have progressed over the years. The 3D object pose problem have received much less but still considerable attention. The series of work from Savarese and Fei-Fei [86, 94, 92] are good examples of how people approach the 3D pose problem in modern contexts, where large benchmarks are established and evaluated for discrete viewpoint classification [86, 28].

There have been, however, divergent trends between object recognition and pose estimation. Latest progresses in object recognition employ discriminative templates directly trained from image gradients [33]; latest 3D pose models group features into parts and learn generative models of their relationships [94, 92].

We believe the two problems should be one and identical, that a good framework of object detection should be able to handle both category and viewpoint classification. In particular, discriminative learning, which has seen great successes in category classification, should readily apply to viewpoint classification.

In this work we present strong empirical proof that it is indeed the case: a discriminatively learned mixture of templates, extending the latent HOG framework of Felzenszwalb et al [33],

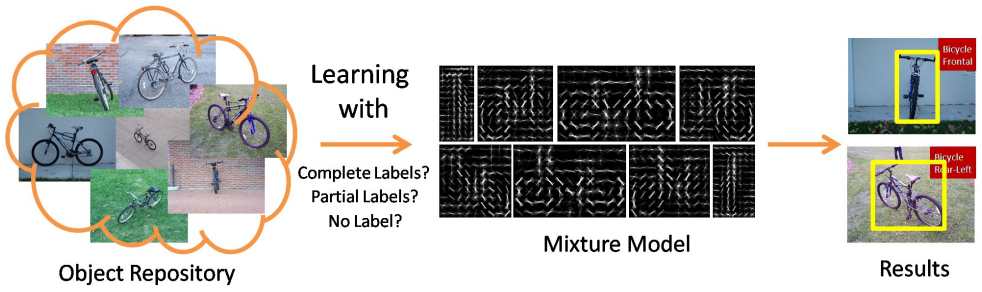


Figure 4.1: We propose to use a discriminative mixture of templates for object viewpoint classification. We discriminatively learn a large mixture of templates using HOG [23, 33] and show that the templates correspond well to the canonical views of an object, which are directly used for viewpoint classification and significantly outperform the state of the art. We show that the mixture model works well when trained with complete viewpoint labels (supervised), a subset of labels (semi-supervised), and no viewpoint labels (unsupervised). We then extend the mixture model for continuous pose prediction, again using a discriminative mixture of templates.

is capable of representing a large number of viewpoints (as components) and handling both category and viewpoint classification. A mixture-of-HOG model produces superior results for all the three cases of supervised (with groundtruth view labels), semi-supervised (with a subset of view labels) and unsupervised (no view labels) viewpoint learning (see Figure 5.1). Furthermore, the mixture-of-templates approach has a natural extension to the continuous case: we propose a continuous viewpoint model which linearly approximates local appearance variations at each discrete view. This model is discriminatively trained, just as in the discrete case, and outputs a continuous 3D viewpoint/pose.

We evaluate our approach on a number of 3D object databases, including the 3DObject Database of Savarese [86], the VOC2006 car database [28], and a dataset of our own for benchmarking continuous viewpoint estimation. We show that we significantly outperform the state-of-the-art results on all these challenging benchmarks: we improve the 8-way viewpoint classification accuracy on the 3DObject database from 57% to 74%, and that on the VOC2006 cars from 73% to 86%. For the continuous case, we show that our discriminative mixture model outperforms a number of baselines, including one using the closest discrete viewpoint and one using linear regression on top of the viewpoints.

4.2 Related Work

Understanding 3D objects and scenes from 2D views is the fundamental task of computer vision. In the early days vision researchers paid close attention to the 2D-to-3D correspondence, but many approaches were line-based and had many difficulties dealing with real-life images. The aspect graph of [48] presents a theory for modeling 3D objects with a set of

inter-connected 2D views. This theory has a sound psychological foundation (e.g. [17]) and has been very influential and underlies most approaches to 3D object recognition.

Estimating the 3D pose of objects is a classical problem, and many solutions have been developed using either local features (e.g. [24]) or shape outlines (e.g. [55]), usually assuming perfect knowledge of the object. With the maturation of local feature detection (as in SIFT and its variants), latest progresses on pose estimation have mostly been local-feature based (e.g. [20, 25]) and performed fairly well on instances of objects, preferably with texture.

There has been an increasing interest lately in 3D object pose classification, which aims at predicting a discrete set of viewpoints. A variety of approaches have been explored (e.g. silhouette matching [22] or implicit shape models [7] or virtual-training [19]). At the same time, many works on category-level classification also address the issue of multiple views (e.g. [95, 80]).

The series of work from Savarese and Fei-Fei [86, 94, 92] directly address the problem of 3D viewpoint classification at the category and are the most relevant for us. They have developed a number of frameworks for 3D viewpoints, most adopting the strategy of grouping local features into parts and learning about their relations. Similar approaches have been adopted in a number of other works (e.g. [52, 61]) that show promising results. The 3DObject dataset of Savarese et al [86] is a standard benchmark for viewpoint classification and has a systematic collection of object views. A number of categories from the PASCAL challenge [28], such as cars, are also annotated with viewpoints. We quantitatively evaluate our approach on these datasets.

The most recent progress in object recognition sees the use of discriminatively trained templates [23, 33, 98]. These techniques have been shown to perform very well on real-life cluttered images. In particular, the work of [33] presents a way to train mixture-of-components for object detection, and they illustrated the procedure with two components on cars and pedestrians. The context-based discriminative clustering work of [53] is similar in spirit. Our work is based on the mixture-of-HOG approach but focuses on viewpoints instead of categories. We explicitly handle viewpoints and train HOG models with a large number of viewpoints/components. We also develop approaches for semi-supervised and unsupervised learning of viewpoints, and extend the discrete viewpoint model to the continuous case.

4.3 Discrete Viewpoint Models

In this scheme, given an example object, the models for each category return a confidence score of the object being in that category as well as a *discrete* viewpoint label associated with a canonical pose of that category. In many cases, such poses have semantic meanings, for instance, the frontal/side views of a car. We design each of these models as a mixture of HOG-based templates corresponding to multiple canonical poses of the category. We formulate the score function of example x as

$$S_{\mathbf{w}}(x) = \max_{v \in \mathcal{V}} \ \langle w_v, \ \psi_v(x) \rangle = \max_{v \in \mathcal{V}} \ w_v^T \psi_v(x) \tag{4.1}$$

where $\mathbf{w} = \{w_1, w_2, \ldots, w_V\}$ are the learned mixture of templates, $\mathcal{V} = \{1, 2, \ldots, V\}$, V is the number of canonical viewpoints in the model, and $\psi_v(x)$ is the feature representation of x under viewpoint label v. Since the dimensions of templates can be different, $\psi_v(x)$ is designed to match the dimension of w_v.

Accordingly, the predicted viewpoint label of x is

$$\tilde{v}_d(x) = \arg\max_{v \in \mathcal{V}} \; w_v^T \psi_v(x) \tag{4.2}$$

where the subscript d indicates a *discrete* label.

Features: we are in favor of HOG-based features because they encode spatial layout of object shape and handle well with intra-class and intra-viewpoint variations. We use the implementation of [33] for feature construction and normalization.

Detection: we adopt the standard framework of multi-scale window scanning for localizing objects in the image. The windows whose scores are higher than a learned threshold are picked as candidate detections, and non-max suppression is applied as postprocessing to remove redundant window detections.

Training

We extend the training algorithm of [33] to cope with viewpoint classification. For each category, we learn a mixture of V-component templates $\mathbf{w} = \{w_1, w_2, \ldots, w_V\}$ from a set of positive and negative training examples denoted by $\{x_1, x_2, \ldots, x_P\}$ and $\{z_1, z_2, \ldots, z_N\}$. Our learning framework attempts to "match" every positive example with at least one of these templates, and every negative example with none of the templates. Mathematically, the large margin optimization of this scheme is formulated as

$$(\mathbf{w}^*, \; \lambda^*) = \arg\min_{\mathbf{w}, \lambda} \sum_{v=1}^{V} \left\{ \frac{1}{2}\|w_v\|^2 + C_{Neg} \sum_{n=1}^{N} l\left(-w_v^T \psi_v(z_n)\right) + C_{Pos} \sum_{p=1}^{P} \lambda_v^p \cdot l\left(w_v^T \psi_v(x_p)\right) \right\} \tag{4.3}$$

subject to $\lambda_v^p \in \{0, 1\}$ and $\sum_{v=1}^{V} \lambda_v^p = 1, \; \forall \; p = 1, \ldots, P$. Here, λ are binary component labels. $l(s) = \max(0, 1 - s)$ is the hinge-loss function. C_{Pos} and C_{Neg} control the relative weights of the regularization term.

Our training procedure is directly based on [33]: each template w_v is initialized through a set of positive examples initially labeled as viewpoint v. In each iteration, all templates are updated simultaneously through data-mining hard negative examples and updating viewpoint labels λ of positive examples.

In [33], λ are considered as latent variables and thus the cost function does not enforce λ to match their true values. Here, we solve a more general problem which includes the scenarios when λ are partially or completely unknown. Furthermore, model initialization in [33] is solely based on aspect ratio; it is not designed for general viewpoint modeling and

thus far from optimal for our problem. We will show that a carefully designed initialization is necessary to learn reasonable templates for canonical viewpoints.

Denote $\{v_d(x_1), v_d(x_2), \ldots, v_d(x_P)\}$ as the groundtruth viewpoint labels of the positive examples. In the following, we consider three scenarios, where these labels are completely known, partially known, and unknown. We name them supervised, semi-supervised, and unsupervised cases, respectively.

Supervised Case

In the supervised case, each $\lambda_v^p = \mathbf{1}[v = v_d(x_p)]$ is fixed. The model is initialized by partitioning the positive examples into groups based on the viewpoint labels and learn one viewpoint template from each group. In the model update step, the optimization is reduced to a linear SVM formulation.

We note that although we do not change component labels during the training process, this is different from training each component independently, as the training process uses a single regularization constraint and enforces the margin on all the clusters simultaneously. This has proved to be critical in learning mixture models that are balanced and accurate for viewpoint prediction.

Semi-supervised Case

In the semi-supervised case, we first build a multi-viewpoint classifier using the positive examples that have known viewpoint labels. In practice, we use the libsvm multi-class classification toolbox[18] on the HOG features. Once the rest of the positive examples are classified, we initialize component templates based on either known or estimated labels. In the model update step, we fix the labels for those who have known viewpoint labels, and allow the others to change.

Unsupervised Case

In the unsupervised case, model initialization is crucial for accurate viewpoint classification, because no explicit constraint in the later stage of optimization is imposed on the viewpoint labels. [33] partitions positive examples into component groups based on a simple aspect ratio criterion. We use a Normalized Cut-based clustering scheme for initialization. We define an appearance distance between two positive examples x_i and x_j as

$$d(x_i, x_j) = \alpha \cdot \chi^2(\psi_0(x_i), \psi_0(x_j)) + (1 - \alpha) \cdot \|\text{Asp}(x_i) - \text{Asp}(x_j)\|_2 \tag{4.4}$$

where $\psi_0(x_i)$ is the HOG descriptor of x_i under a standard template size, and $\text{Asp}(x_i)$ is the normalized aspect ratio of the bounding box of x_i. Next, we convert the distances into affinity measurements using the exponential function and obtain the component groups by applying the Normalized Cut[88] algorithm on the resulting affinity matrix. This provides

us with relatively even partitionings on the positive examples, which is important for good unsupervised performance.

In the model update step, since Eqn. 4.3 describes an integer-based non-convex problem([53], [1]), one tractable solution is to iterate between optimizing $\mathbf{w}$ given fixed labels λ and optimizing λ given fixed template weights $\mathbf{w}$. The former is an SVM and the latter optimization step is simply

$$\lambda_v^p = \mathbf{1}[v = \arg\max_s \left(w_s^T x_p\right)] \ \forall p = 1, \ldots, P \tag{4.5}$$

4.4 Continuous Viewpoint Models

In the continuous viewpoint case, we are interested in estimating the real-valued *continuous* viewpoint angles of an example object in 3D, denoted by $\theta \in \mathcal{R}^3$, which uses the angle-axis representation. We assume that the camera projection of the object is orthographic so that given a fixed orientation θ, the appearance of the object only changes in scale.

To obtain θ for a test object x, we modify the mixture model in the discrete viewpoint case and reformulate the score function as

$$S_{\mathbf{w}}(x) = \max_{v \in \mathcal{V}, \Delta\theta} f(v, \Delta\theta) = \max_{v \in \mathcal{V}, \Delta\theta} \left(w_v + g_v^T \Delta\theta\right)^T \psi_v(x) - d(\Delta\theta) \tag{4.6}$$

$$\theta(x) = \theta_{v^*} + \Delta\theta^* \tag{4.7}$$

where $\mathbf{w} = \{w_v\}$ and $\psi_v(x)$ are the same as before. g_v are the "gradients" of the template w_v over θ at discrete viewpoint v. $\Delta\theta$ are the offset viewpoint angles of x with respect to the canonical viewpoint angles θ_v. $d(\cdot)$ is a quadratic loss function that confines $\theta(x)$ to be close to θ_v. Denote $\Delta\theta$ by their elements $[\Delta\theta_1, \Delta\theta_2, \Delta\theta_3]^T$, then $d(\Delta\theta) = \sum_{i=1}^3 d_{i1}\Delta\theta_i + d_{i2}\Delta\theta_i^2$. In Eqn. (4.7), v^* and $\Delta\theta^*$ are obtained when the score function reaches its maximum. The variables w_v, g_v, θ_v and d_{i1}, d_{i2} are learned from training data.

This continuous viewpoint model can be interpreted as follows: we partition the continuous viewpoint space into small chunks where each chunk has a canonical viewpoint. For every viewpoint in the same chunk, we approximate its template as a linear deformation of the canonical template with respect to the difference of viewpoint angles from the canonical angles. We show that in practice, this approximation is reasonable when the chunk size is relatively small, and the model produces viewpoint classification performance superior to a number of baseline methods.

Detection: The multi-scale window scanning is again applied for localizing objects in the image. To find optimal v and $\Delta\theta$ in Eqn. (4.6) at a given location, we first maximize $\Delta\theta$ over any fixed v

$$\frac{\partial f(v, \Delta\theta)}{\partial \Delta\theta_i} = g_v(i)^T \psi_v(x) - d_{i1} - 2d_{i2}\Delta\theta_i = 0 \tag{4.8}$$

Hence, we obtain

$$\Delta\theta_i(v) = \left(g_v(i)^T \psi_v(x) - d_{i1}\right) / 2d_{i2} \tag{4.9}$$

where $g_v(i)$ is the i'th column of g_v. Next, we enumerate over the discrete variable v with $\Delta\theta_i(v)$ and pick the pair with maximal score $S_{\mathbf{w}}(x)$.

Training

In training, for positive examples $\{x_1, x_2, \ldots, x_P\}$, their continuous viewpoint groundtruth labels $\{\theta_1, \theta_2, \ldots, \theta_P\}$ are given. Therefore, we rewrite the score function in Eqn. (4.6) as

$$\begin{aligned} f(v, \Delta\theta) &= (w_v + g_v \Delta\theta)^T \psi_v(x) - d(\Delta\theta) && (4.10) \\ &= \tilde{w_v}^T \tilde{\psi_v}(x) && (4.11) \end{aligned}$$

where

$$\begin{aligned} \tilde{w_v} &= [w_v,\ g_v(1),\ g_v(2),\ g_v(3),\ d_{11},\ d_{12},\ d_{21},\ d_{22},\ d_{31},\ d_{32}] \\ \tilde{\psi_v}(x) &= [\psi_v, \Delta\theta_1\psi_v, \Delta\theta_2\psi_v, \Delta\theta_3\psi_v, -\Delta\theta_1, -\Delta\theta_1^2, -\Delta\theta_2, -\Delta\theta_2^2, -\Delta\theta_3, -\Delta\theta_3^2] \end{aligned}$$

If all canonical viewpoint templates θ_v are known, $\psi_v(x)$ are completely observable and we can substitute $\tilde{w_v}$ and $\tilde{\psi_v}(x)$ for w_v and $\psi_v(x)$ in the training framework of the discrete viewpoint case. Now, θ_v are unknown, but we can initialize them from initial partitions of positive data (clustering on θ) and update them in each training iteration based on maximizing the cost function.

4.5 Experimental Evaluation: Discrete Viewpoints

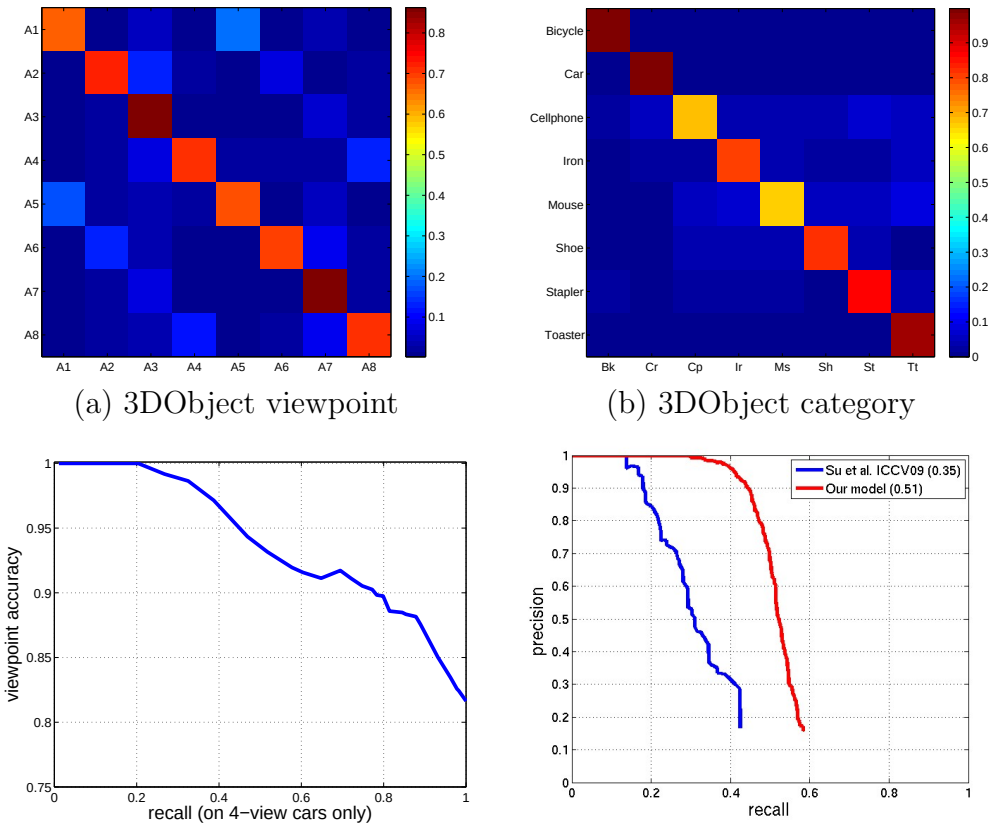
For discrete viewpoint classification, we evaluate our proposed models on two standard and challenging databases: the 3DObject[86] and the VOC2006 cars[28]. The 3DObject dataset consists of 10 categories and 8 discrete viewpoint annotations for each category. We exclude the head and the monitor categories as they are not evaluated in previous work. Quantitative results on viewpoint and category classification are evaluated by means of confusion matrix diagonals, and averaged by 5-fold training/test partitions. On the other hand, the VOC2006 car database consists of 469 car objects that have viewpoint labels (frontal, rear, left and right). In the experiments we only use these labeled images to train mixture viewpoint models, with the standard training/test partition. The detection performance is evaluated through precision-recall curve. For both databases, we try our best to compare with previous works that have the same complete set of evaluations.

In the following sub-sections, we analyze our results in three different levels of supervision on the training data: *supervised*, *semi-supervised*, *unsupervised*.

Supervised Case

Database	3DObject		VOC2006 cars		
Method	[86]	Ours	[94]	[92]	Ours
Viewpoint	57.2%	**74.2 ± 0.9%**	57.5%	73.0%	**85.7%**
Category	75.7%	**85.3 ± 0.8%**	-	-	-

Table 4.1: Supervised Case: viewpoint and category classification results (quantified by averages of confusion matrix diagonals). For category detection performance on the VOC2006 cars, we compare the precision-recall curves with [92] in Figure 4.2(d).



(a) 3DObject viewpoint (b) 3DObject category

(c) VOC cars viewpoint (d) VOC cars category

Figure 4.2: Supervised Case: viewpoint labels are all known for positive examples. **(a) (b)** Average confusion matrices of the viewpoint and category classifications in the 3DObject. **(c)** Viewpoint classification accuracy as a function of the object recall in the VOC cars. **(d)** Precision-recall curves of car detection. Note that our car model only trains on the 4-view cars and tests on the whole test images.

Table 4.1 summarizes the viewpoint and category classification results when the viewpoint labels of the positive training data are known. We significantly outperform [86], the state of the art on the 3DObject database, in both viewpoint and category classification. We also show a significantly higher (4-view) viewpoint classification rate on the VOC2006 car database compared to the earlier work of [94] and [92]. Figure 4.2 shows a close look of our results.

Note that in (a), the main viewpoint confusion pairs in 3DObject are those off by 180 degrees, for example, frontal vs. rear or left vs. right views. Category confusion matrix is shown in (b). (c) illustrates the change of viewpoint classification rate with object recall in VOC2006 cars. The curve suggests that the viewpoint classification accuracy increases with lower recall (and thus higher precision/category detection). (d) compares the precision-recall curves of [92] with ours. Note that even our car mixture model only covers 4 views, it still produces superior performance comparing to [92] in detection.

Semi-supervised Case

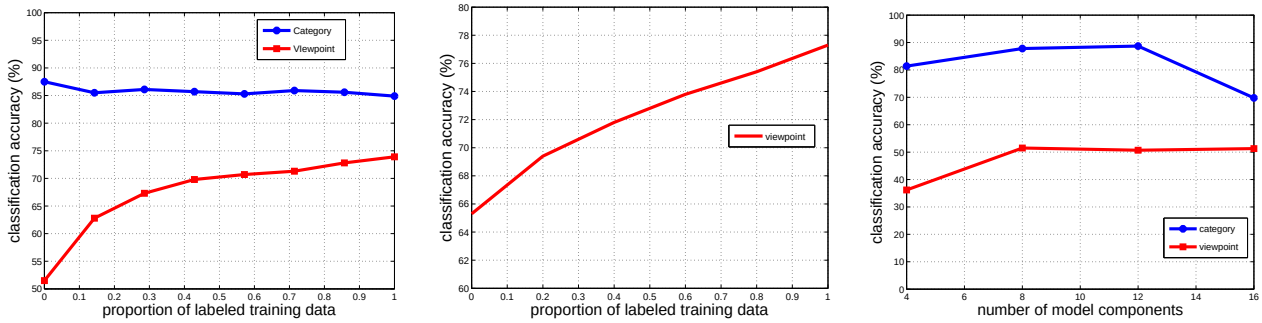
In the semi-supervised case, we are interested in knowing how much partial information from positive training data is "sufficient" to build a reasonable viewpoint model. Figure 4.3 (a, b) illustrate the viewpoint and category classification accuracies with changes in the proportion of training data having discrete viewpoint annotations. Zero proportion means no annotation which corresponds to the unsupervised case, whereas "proportion equals one" is the case of being totally supervised. Note that the accuracy numbers here are evaluated on the whole test set, not the set including only correct category prediction. We notice that even a small proportion (30% in the 3DObject) of annotated data significantly improves the viewpoint classification performance, while the category classification performance remains roughly constant with change of the number of annotated data. (We do not show the curve of category classification on the VOC2006 cars as it is a detection task.)

Unsupervised Case

Evaluation Methodology

The upper half of Table 4.2 compares three model initialization schemes in terms of the viewpoint and category accuracies. We note that our proposed N-cut framework significantly outperformed the aspect ratio criterion by [33] for viewpoint classification. We also compute how far we can reach by computing an "upper bound" performance using the ground truth viewpoint labels of training data in initialization, shown in the third column of the first two databases. We see that the N-cut produces results close to and sometimes even better than the "upper bounds".

We quantitatively evaluate the quality of viewpoint clustering using the following statistics: *purity*, *normalized mutual information*, *rank index*, and *F measure*[70], shown in the



(a) 3DObject Semi-supervised

(b) VOC cars Semi-supervised

(c) 3DObject Unsupervised

Figure 4.3: Semi-supervised/Unsupervised Cases: viewpoint and category classification accuracies as a function of either the proportion of positive training data with viewpoint annotations (semi-supervised) or the number of model components/templates (unsupervised). **(a)**: Semi-supervised model on the 3DObject. **(b)**: Semi-supervised model on the VOC2006 cars. For these semi-supervised cases, the category detection performance is robust and largely independent of the availability of viewpoint labels. Viewpoint classification is robust up to about 30% of labeling. **(c)**: Unsupervised model on the 3DObject dataset.

bottom half of Table 4.2. All measurements of these statistics exhibit consistent behavior as the basic evaluation.

Number of Model Components

The number of components V in the unsupervised model is pre-determined. As a result, we are interested in knowing the impact of this parameter on the viewpoint and category classification performance. Figure 4.3(c) shows both accuracies with V on the 3DObject database. Note that for viewpoint classification, the accuracy undoubtedly breaks down when V is deficient (4) to explain the variety of data in viewpoint (8). It is, however, surprisingly insensitive to V when it gets large. On the other hand, for category classification, the accuracy breaks down when V is large, and insensitive with small V.

4.6 Experimental Evaluation: Continuous Viewpoints

For continuous viewpoint estimation, there is no standard benchmark database available, partly because it is considerably harder to establish groundtruth data to cover arbitrary 3D rotations. [20] uses a selected set of translations and rotations for (continuous) pose estimation. [79] does not use groundtruth but compare results using artificially distorted images. In the case of [74], rotation is limited to in-plane rotation on the ground.

Database	3DObject			VOC2006 cars		
Method	[33]	N-cut	Labels	[33]	N-cut	Labels
Viewpoint	40.2%	**51.5%**	63.4%	47.0%	**65.6%**	65.3%
Category	86.5%	**87.8%**	87.2%	-	-	-
Purity	0.42	**0.53**	0.65	0.58	**0.77**	0.76
NMI	0.43	**0.55**	0.61	0.41	**0.52**	0.50
Rank Index	0.77	**0.83**	0.86	0.71	**0.80**	0.80
F Measure	0.36	**0.45**	0.54	0.61	**0.68**	0.67

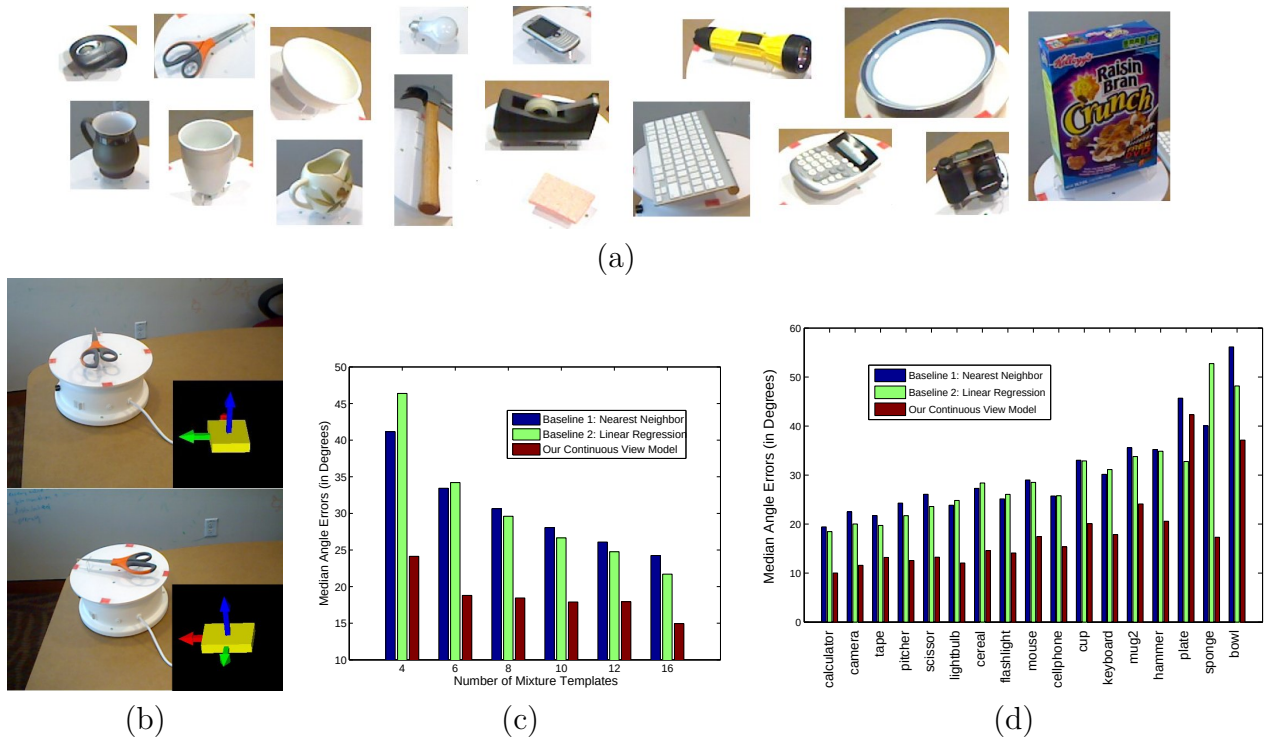
Table 4.2: Unsupervised Case: viewpoint and category classification accuracies as well as four viewpoint clustering measurements[70] on two databases. We show comparison of 3 model initialization schemes ([33], N-cut, and Labels) on the 3DObject and VOC2006 cars. Note that [33] performs poorly in viewpoint classification. The "N-cut", proposed in this paper where the numbers are bolded, produces significantly better results than [33]. The "Labels" case uses the ground truth viewpoint labels to initialize models, which are considered to produce the "upper-bound" results.

We believe that a good database with full 3D pose groundtruth is crucial for the advances of pose estimation techniques, and we set to collect a 3D pose database using commercially available IMUs: we use the Microstrain 3DM-GX1 sensors and attach it to a PrimeSense video camera. The Microstrain provides gyro-stabilized full 3D orientation at about 80Hz, and the camera records 640x480 frames at 30Hz. The two streams are aligned manually.

We collect a continuous object pose database covering 17 daily objects with a variety of shape, appearance and scale (Fig 4.4(a)). We put each object on a turning table, let the object turn, while hand-holding the camera/IMU pair and moving it at varying heights and orientations. We typically let each object rotate for 4-5 circles and take about 2K video frames total. In our evaluation experiments, we use all 17 objects and about 1K frames for each object. Frames are evenly and randomly partitioned for training and testing. Object masks are computed from background subtraction and are used to find bounding boxes.

We compare our continuous viewpoint model with two baseline methods. The first one employs a nearest neighbor scheme. Each test example is assigned the same continuous viewpoint label as that of the example's closest mixture template. The second one learns a linear regression model on the responses of all mixture templates to infer viewpoint labels. The comparison of the results is shown in Figure 4.4(c) where prediction errors are measured by the amount of rotation it takes to go from the predicted pose to the groundtruth pose (in degrees). Because the errors can sometimes be very large due to the symmetry in the object shape and appearance, we use the median angular error as the evaluation metric.

Our proposed continuous viewpoint model constantly outperforms both baselines under different numbers of mixture templates. The errors are reduced as the numbers of templates increase which suggests that a sufficient number of canonical viewpoints is needed to cover



(a) (b) (c) (d)

Figure 4.4: Continuous viewpoint classification results on the new continuous viewpoint dataset consisting of 17 daily objects (a), covering a wide range of shape and scale. We place objects on a turning table and attach an IMU to a hand-held camera; groundtruth orientations of objects relative to the camera are estimated from both the IMU readings and turning table rotations (b). our discriminatively trained continuous pose model constantly outperforms two baseline methods (assigning to nearest discrete pose, and a linear regression on top of discrete poses). (c) shows the performance comparisons as the number of discrete viewpoints varies in the mixture model. (d) shows the results for each of the 17 objects, with the number of mixture components set to 8. We observe that viewpoint prediction is challenging (and ill-defined for some of the symmetric objects), and our discriminative approach consistently outperforms the baselines for most objects.

the entire viewpoint hemisphere. A closer examination of the per-category performance is shown in Figure 4.4(d). The errors are in general large for symmetric categories(e.g. plate, bowl) and small for asymmetric ones which meets our intuition. As we see from the examples, the database is challenging: even though the background is simple and so far instance-based, there is a lot of inherent ambiguity in inferring pose from shape, and the improvement in accuracy using our continuous model is substantial.

4.7 Conclusion

In this work we have applied the discriminative template learning framework for joint category and viewpoint classification. Our main contribution is to show that a mixture-of-templates model discriminatively learned in a detection framework capture the characteristics of different views and can be directly used for viewpoint classification. Our results significantly outperform the state-of-the-art on a number of standard 3D object databases. We have also shown that with a good initialization (e.g. Normalized Cuts and discriminative clustering), we are able to produce meaningful viewpoint clusters and promising classification accuracy with a small amount of training labels.

In addition, we have extended the mixture-of-templates approach to the continuous viewpoint case. We use a linear model to capture local appearance variations at each canonical view, and these models are discriminatively trained as in the discrete case. We have been building up a dataset with continuous viewpoint groundtruth, and our model has shown promising performance comparing to a number of baselines, including discrete nearest neighbor and linear regression.

Although our work is still in a preliminary stage, we believe that our results are very important in proving the use of discriminative learning for viewpoint classification. It is no coincidence that our results outperform the state of the art on 3D object databases. Just as in the category case, discriminative learning addresses the classification problem directly and is very powerful in exploring noisy image data. There are many future opportunities in exploring the synergies between object classification and viewpoint estimation.

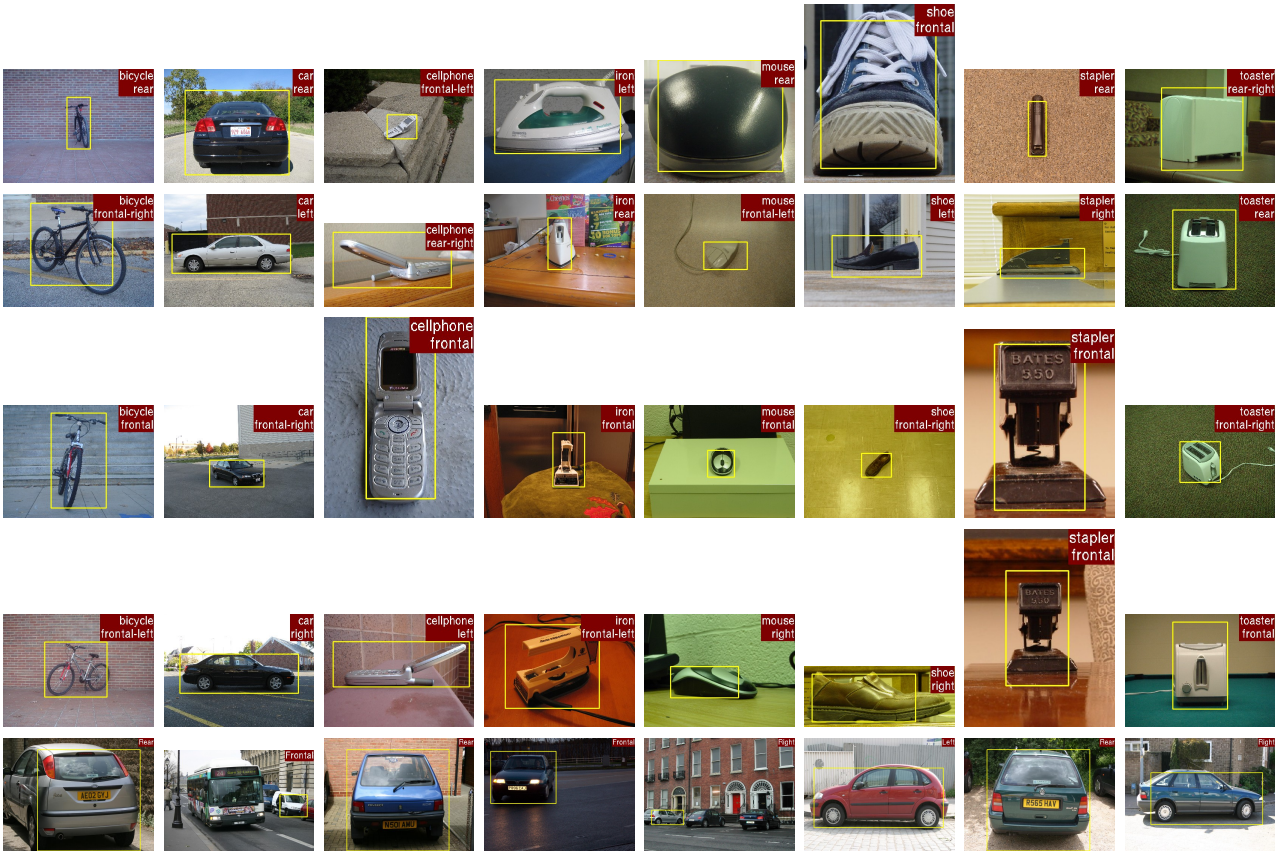


Figure 4.5: Sample viewpoint classification and category detection results. The yellow bounding boxes indicate object detection, and the labels on the upper-right corners show the predicted object category and viewpoint. The top 4 rows show results from the 3D object category database, and the bottom row shows results from the PASCAL VOC 2006 car database.

Chapter 5

Multi-Components for Object Recognition

5.1 Introduction

Consider the object in the center of Figure 5.1. Although its appearance is very different from any of the surrounding instances, they all belong to the same semantic category "aeroplane". The main causes of intra-class variations in recognition are pose and viewpoint changes, as well as the presence of visually heterogeneous subcategories. For instance, aeroplanes look quite different from side and 45-degree views, and their appearance also changes significantly between the three main subcategories: wide-body passenger jets, fighter jets and propeller aeroplanes. We refer to such visual clusters as *components*.

In this paper, we propose an approach that models each component independently, which we show is easier and more accurate than attempting to characterize all components in a monolithic model. A significant advantage of our approach over monolithic models is that it enables tasks that are finer-grained than bounding box prediction. Objects in the same component are tight in configuration space, and thus inference on the object keypoint locations and segmentation masks becomes feasible. The keypoints and mask of an object can be predicted from those of its most likely component.

Although monolithic models are still common in the literature [23, 85], there have been several influential approaches modeling multiple components of objects [33, 15, 41, 69]. However, each of these methods has it own limitations. Felzenszwalb et al.[33] learn global and part components jointly, but the number of components is pre-defined and not inferred from data. Gu and Ren[41] focus on modeling only viewpoint variations of objects and ignore other sources of intra-class variations such as subcategories. Bourdev et al.[15] use keypoints to align objects but their poselet models typically characterize parts rather than global objects. Malisiewicz et al.[69] is most similar to our work. However, that approach uses only one positive instance for training, which significantly reduces the generalization capacity of each component model and therefore compromises categorization performance. Last but

Figure 5.1: One key challenge of object categorization is intra-class variations induced by pose changes, subcategories, truncation, etc. Since objects belonging to the same category form clusters in the appearance and configuration spaces, it is natural to construct individual models for each cluster and combine them to operate at the category level. We refer to such a cluster as *component*.

not least, all these methods use expensive multi-scale window scanning for object detection, which sets a limit on the number of components as well as on the ability to apply more sophisticated features and more powerful classifiers for better accuracy.

To reduce the computation cost during detection, we adopt a bounding box generation scheme that selectively proposes candidate windows that are more likely to contain objects. Our scheme produces fewer than 500 bounding boxes per image on the VOC2010 dataset, drastically reducing the search space when compared to exhaustive multiscale window scanning, while maintaining high recall of objects over all 20 categories. Furthermore, since this scheme does not rely on category-specific knowledge, the number of candidate windows is independent of the number of categories and thereby scalable to large data.

Although the idea of object candidate selection has been explored in the past, our method differs from previous related work. Both Alexe et al.[2] and Endres and Hoiem[27] propose objects using combined bottom-up and top-down knowledge but their proposals are category independent. Li et al.[59] propose object candidates through a set of ranked figure-ground hypotheses, whereas our proposal is based on generic hierarchical segmentation. Van de Sande et al.[85] also build a segmentation tree by greedily merging initial segments into object

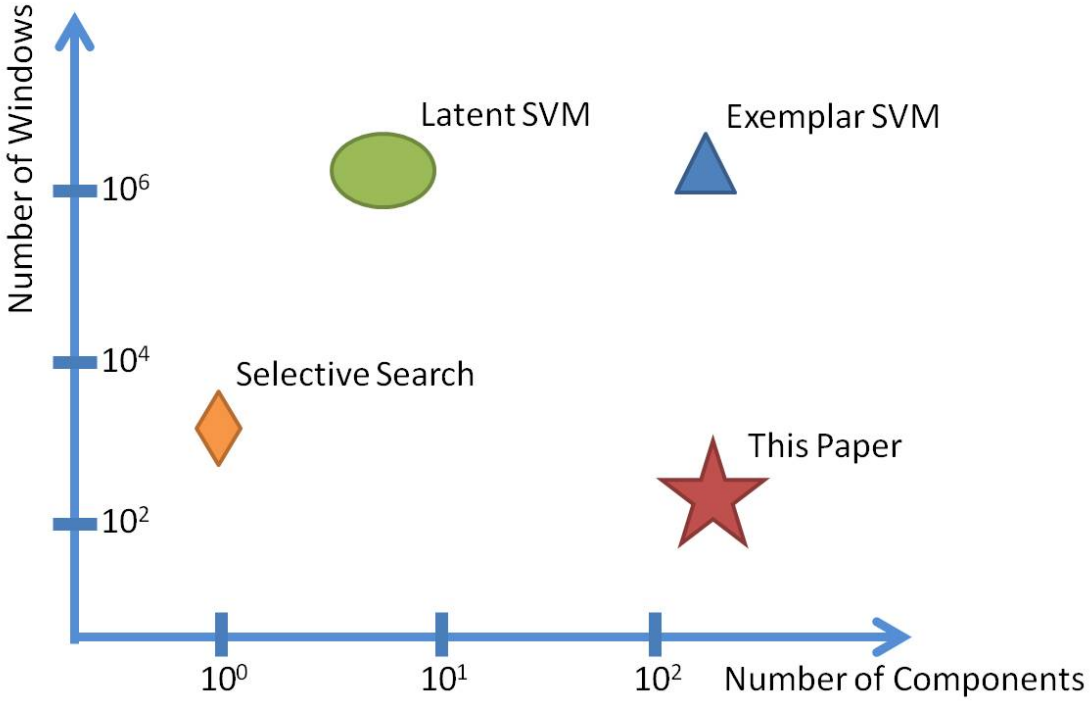


Figure 5.2: Comparison of our approach with related methods ([33] for Latent SVM, [69] for Exemplar SVM, and [85] for Selective Search) in 2D space where the two axes represent the number of components and the number of window candidates. Our approach is distinct from others by combining multi-component models and selective window candidates.

candidates and achieve very high recall on the VOC2007 dataset. In contrast to them, we use high quality contour-based segments instead of over-segmentations as initial segments, and results in fewer (500 with respect to 1500 using their model) candidate proposals per image. Finally, Gu et al.[42] also start with contour-based segments, but their candidate proposal scheme has several limitations, such as broken objects from strong internal contours, too few candidates and low recall of complete objects. Our window generation approach, described in Section 2, overcomes these limitations.

Overall, this paper presents four distinct contributions that, when combined, provide competitive performance on the PASCAL detection challenge while enabling finer-grained tasks than bounding box prediction: (1) global and generic multi-component models characterizing intra-class variation; (2) a category level classifier aggregating responses from multiple components; (3) a bounding box generation mechanism significantly reducing computation complexity during inference; (4) a simple yet effective algorithm allowing prediction of object keypoint locations and masks. Figure 5.2 depicts various methods in a 2D plot that characterizes the number of components of an object model and the number of scanned windows per image, respectively. This work is, to the best of our knowledge, the first one

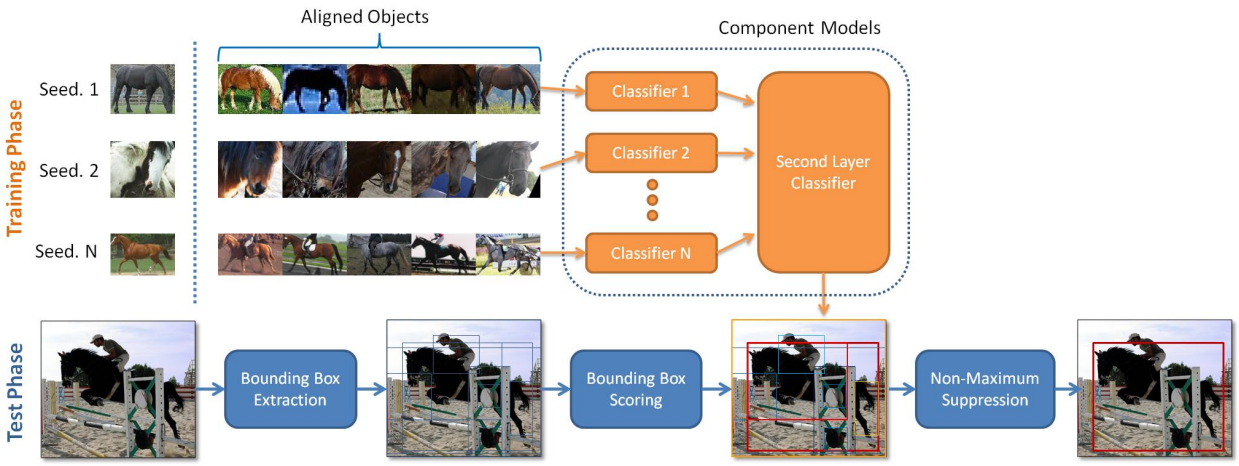


Figure 5.3: Illustration of our detection pipeline. In the training phase (top row), given a seed object, the rest of the objects in the training set are warped to align with that seed based on keypoint and object mask annotations. The top aligned horses are then used as positive training data to learn a single-component classifier. Given N seeds, we have N such classifiers. A second-layer classifier takes the outputs of these component classifiers as input, and produces a final likelihood score. In the test phase (bottom row), a small number of object bounding boxes are proposed based on bottom-up segmentation to avoid exhaustive window scanning. Each candidate box is scored using our learned two-layer classifiers. Finally, a non-maximum suppression is applied to generate detection results.

addressing the combination of multi-component models and selective window candidates.

Figure 5.3 overviews the pipeline of our detection framework. In the training phase, a two-layer model is learned to capture and aggregate the components of an object category from the data. Each first-layer model is a binary classifier trained with a seed object and a list of aligned examples. In the detection phase, a small number of candidate bounding boxes are generated based on segmentation for each image. After these boxes are scored by the two-layer model, a non-maximum suppression is applied to produce final detection results. Compared to monolithic models, this framework provides additional information on the component of the test object, and thus enables inference of object keypoints and mask prediction.

The rest of the paper is organized as follows. In Section 2, we describe our bounding box generation scheme. Sections 3 and 4 show how to find and train a component model. Section 5 describes the fusion mechanism of these component models into a final classifier. We will discuss the experiments in Section 6 and conclude in Section 7.

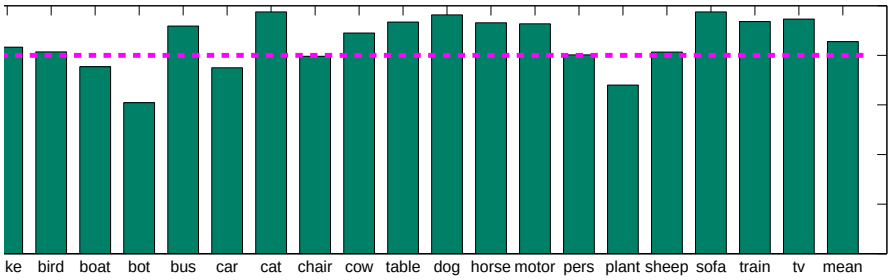


Figure 5.4: Proportion of objects found by our bounding box generation on all 20 PASCAL VOC categories. We obtain a recall of 80% or higher in 16/20 categories.

5.2 Bounding Box Generation

Exhaustive window scanning is computationally expensive when the number of components scales up to hundreds. Therefore, a bounding box selection scheme is necessary to prune out in an early stage windows that do not contain any object. In this section, we present a bottom-up strategy to propose bounding box candidates using segmentation as primary cue.

We start with the publicly available segmentation method of [5]. The output of this algorithm is a hierarchical tree of segments at different levels of contrast. These segments are few in number (about 150) and provide almost full boundary recall when the threshold is low. In order to cope with objects of a large range of sizes, we compute segmentation trees at three resolutions of the input image ($\times 1.0$, $\times 0.6$, $\times 0.3$), and add additional candidate segments by merging pairs or triplets of adjacent segments at the two coarser resolutions. We dump segments of all scales into a segmentation pool for the input image.

Each segment in the pool proposes a bounding box which is the smallest rectangle containing it. As a result, we generate the same number of bounding box candidates as segments in the pool. Some candidates are identical, even though their original segments are different. After removing duplicates, our strategy ends up with fewer than 500 candidate boxes per image on the PASCAL VOC 2010 training set. Figure 5.4 shows for each category the recall of objects whose ground truth bounding boxes overlap more than 50% with at least one of our proposed boxes. In 16/20 categories, we have a recall rate of 80% or higher.

One main advantage of a bottom-up approach is that the resulting bounding boxes encode object sizes and aspect-ratios naturally and with high accuracy, thus avoiding unnecessary redundancies of candidate boxes. Another advantage is that the number of candidate boxes does not scale with the number of target object categories, providing a huge saving of computation.

5.3 Finding Components

Since the intuition behind our component model is that objects belonging to the same component have smaller appearance and configuration distances to each other, a natural strategy of finding components for a category is to perform clustering with e.g. k-means on all training objects of that category. However, in practice, this strategy does not work well. One main reason is that some less common components are difficult to discover because they can easily be absorbed by components of common poses or dominant subcategories due to unbalanced data. Furthermore, objects within a cluster are in many cases not tight enough to build a robust component model because there is no global alignment among them during clustering.

Therefore, we use an alternative two-step strategy to construct each of our components:

- A "seed" object is picked from the training objects which characterizes a component.
- The rest of the training objects of this category are aligned to the seed through a global transformation using keypoint and mask annotations. The top aligned objects constitute the positive set of the component.

We use annotation data from [16] for keypoints and masks of training objects. These annotations are crowdsourced on the Amazon Mechanic Turk platform. For each category, 10 to 20 semantically meaningful keypoints (e.g. head, tail, wing tips, wing bases, and stabilizer tip for aeroplane) are marked on the objects. Invisible keypoints are not labeled and thus excluded in the global transform step. In addition, object masks are labeled using a polygon approximation.

With these additional annotations in hand, we conduct alignment between two objects using a similarity transformation (including rotation, isotropic scaling and translation). Precisely, let I and J be two objects to be aligned, and p_I, p_J, M_I, M_J be their keypoints and masks, respectively. The transformation matrix $\mathcal{T}$ has a close-form solution when the objective is to minimize the Procrustes distance between two sets of keypoints p_I and p_J. The quality of the alignment is measured by the following distance function:

$$d_{quality} = (1 - \lambda) \cdot d_{procrustes} + \lambda \cdot (1 - d_{overlap})$$

where $d_{procrustes} = \sum_{i \in I} (\mathcal{T}(p_J^i) - p_I^i)^2$ is the Procrustes distance, and

$$d_{overlap} = \frac{Area(M_I \cap \mathcal{T}(M_J))}{Area(M_I \cup \mathcal{T}(M_J))}$$

is the intersection-over-union score between the seed mask and the transformed object mask. The parameter λ controls the relative weight between the two terms. In our experiments, we observe that even $\lambda = 1$ gives reasonable alignment results. Finally, we sort all aligned objects for a given seed using the quality distances defined above, and pick top 32 of them as positive instances of the component model. In the PASCAL VOC 2010 data, 32 is an empirical choice

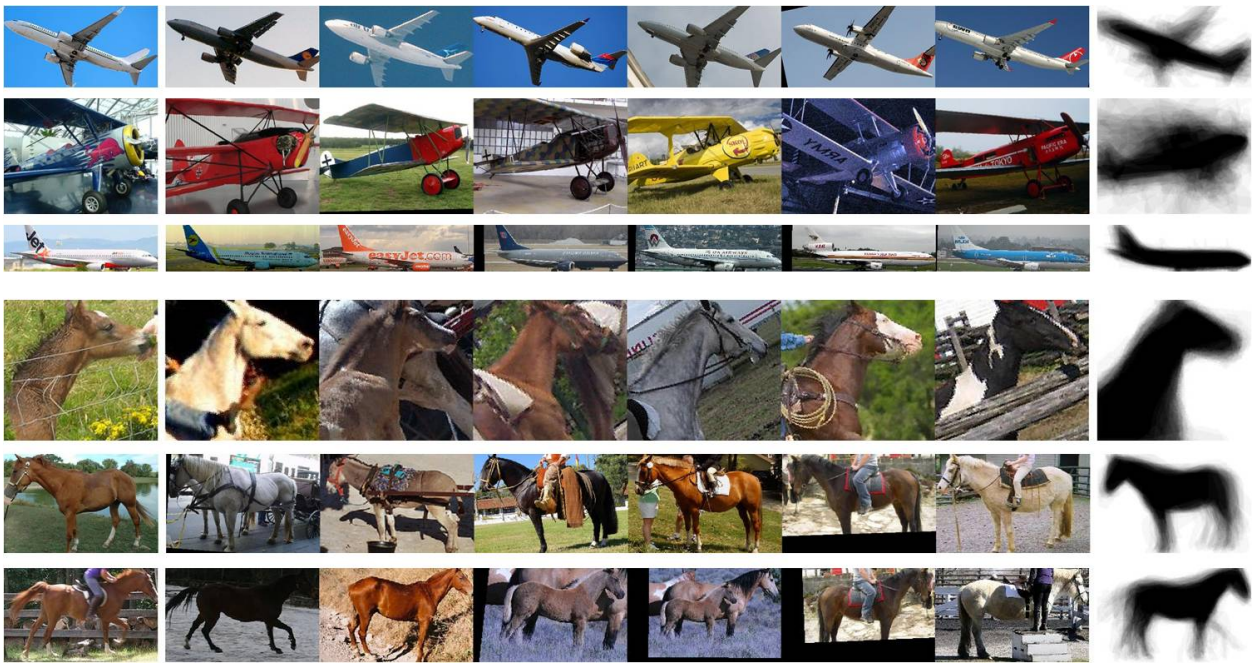


Figure 5.5: Visualization of some of our components for aeroplane (top) and horse (bottom) categories. Each row corresponds to a component. The left-most images are seeds; the middle six images are top aligned objects for each seed; and the right-most images are averaged mask of top 32 aligned objects. Note that, by design, objects within each component are tight in configuration space.

that is small enough to exclude degraded aligned objects for most components of categories, and big enough to make the model generalizable.

With one component model set up, we can easily extend this two-step scheme and construct a set of component models by picking multiple distinct seeds from the training data. Our strategy prevents uncommon components from being absorbed by common ones. Objects within each component are tight in configuration space after global alignment and hence we can train strong classifiers. Figure 5.5 shows our alignment results on the aeroplane and horse categories. Each component model is a binary classifier and we will describe the training framework in the next section.

5.4 Training Components

Each of our component models is a binary classifier attempting to distinguish a particular component of category objects from the rest of the world. We take our top aligned objects as the positive set of the component. After that, we conduct following two steps to complete the training process:

- The negative set of the component is the set of all bounding box candidates extracted from negative training images.
- An Intersection Kernel Support Vector Machine (IKSVM[66, 97]) is learned based on the positive and negative sets and the model is bootstrapped once by data-mining hard negatives. The SVM scores are then converted to probabilities through a sigmoid function whose parameters are also learned from data.

We use four types of features to describe bounding boxes, all using a spatial pyramid representation. A single component model is learned based on one type of feature. The second layer classifier, which we will describe in the next Section, aggregates outputs of multiple components as well as multiple feature models.

Spatial Pyramid of Vector Quantized SIFT

We implement a spatial pyramid of vector quantized SIFT[56] in the standard way: interest points are extracted from an image in a grid basis. A three scale opponent-SIFT[84] descriptors are computed and concatenated to form a vector at each interest point. These vectors are then quantized to codewords based on a class-specific codebook. The size of our codebook is 400. Next, for a given bounding box, we divide it into 2×2 cells and count the frequencies of the codewords within each cell where interest points lie. The final descriptor of the bounding box is the concatenated histograms of codewords within each cell.

Spatial Pyramid of Poselet Activations

The implementation of this feature is similar to that of the vector quantized SIFT, except that we replace the SIFT-based codewords by poselet activations. Poselets[14] have been shown to be powerful in describing shapes for characteristic parts of objects. Compared to SIFT features, poselet activations are more sparse, informative, and discriminative. This feature fires on average twice per image per poselet on the VOC dataset, and provides both strength and rectangular support of the activation. Each poselet model is trained with a highly discriminative classifier. We use pre-trained models of [15]. A typical number of poselets per category is 100 to 200. Given these properties, we apply a "soft count" strategy to aggregate poselet activations within an image cell. Denote $H(C, w)$ as the bin value for

poselet index i in the histogram of image cell C.

$$H(C,i) = \sum_{a \in A} S(a) \times \frac{Area(B(a) \cap B(C))}{Area(B(a))} \times \mathbf{1}(I(a) = i)$$

where $I(a)$,$S(a)$ and $B(a)$ are the index, strength, and support of the activation a, and $B(C)$ is the support of C. Note that we soft count the strength of an activation by the fraction of overlap between the support of the activation and the image cell. It proves essential to smooth out activation noise caused by shifts of activation support.

Spatial Pyramid of Local Coordinate Coding with HOG

The work in [63] demonstrates a state-of-the-art image classification system using spatial pyramid of local coordinate coding (LCC) on local descriptors. Here we implement a similar feature representation. We sample HOG descriptors[23] on a dense grid with step size of four pixels. Then each HOG descriptor is coded using local coordinate coding [101] with codebook size of 8192. Finally, the coding results are max-pooled with a spatial pyramid representation with 1×1 and 3×3 blocks.

Object-centric Spatial Pooling

The work in [82] demonstrates that object-centric spatial pooling (OCP) is more effective than traditional spatial pyramid matching(SPM) based pooling for bounding box classification. Given a bounding box, OCP pools separately the foreground and background regions to form a bounding box representation. In contrast to traditional SPM that only performs pooling on foreground, OCP includes pooling on background and is able to provide more accurate localization. This is because the learned object detector with OCP will prevent the leakage of parts of a object into background.

For each candidate bounding box, we generate its feature representation using object-centric pooling. The way to generate the feature is the same as in Section 5.4 except that the SPM pooling was replaced by OCP.

5.5 Second-Layer Classifier and Non-Maximum Suppression

In order to complete the detection pipeline, we need to make use of our learned component models to make predictions whether a category object exists in a given bounding box location in an image. Our strategy is to learn a second-layer classifier which takes the outputs of our component models as an input vector, and output a single probability score per category.

During this second-layer classifier training, we find that the selection of the training data has a big impact on the performance of detection. First, the performance is improved if we

Training Data		Number of Duplicates		
All	Only Hard Instances	1	3	5
.302	.420	.382	.407	.420

Table 5.1: Design choices of second-layer classifier on the aeroplane category of VOC 2010 val dataset using the spatial pyramid of poselet activation features. We notice that including only active bounding boxes for training and having more near-duplicates as positive data both have positive impacts on the detection performance.

only include those objects on which at least one component classifier fires. One advantage of this choice is outlier removal, since those hard objects that either have a peculiar pose, an unrepeatable occlusion pattern, or have incorrect category labels, will not be included in the training data and thereby makes the second-layer classifier easier to learn. Another advantage is enabling component selection, since bad component models usually fire everywhere and can be easily captured in the hard negative objects. As a result, the learned classifier will assign high weights to good, discriminative components and low or zero weights to bad ones. Second, a few near-duplicates of positive training data also improves the overall performance. We apply a multiple-instance-learning [3] framework to pick best near-duplicates for each training data.

Table 5.1 compares the detection performance given different choices of our second-layer classifier on the aeroplane category.

Finally, we apply non-maximum suppression on the resulting bounding boxes to generate detection results. The bounding boxes are sorted by their detection scores, and we greedily select the highest scoring ones while removing those that are sufficiently covered by a previously selected bounding box. We use 30% as the coverage threshold based on cross-validation results.

5.6 Experiments

Object Detection on PASCAL VOC

We use the standard PASCAL VOC [29] platform to benchmark our detection framework. Each of our component models is trained on VOC 2010 train data, and evaluated on 2010 val. We use all but no more than 500 objects from training data as the set of seed objects. In addition, each seed and its subsequent aligned objects are mirrored to produce a left-right symmetric model. These design choices end up with 400 to 1000 components, depending on the category.

Table 5.2 illustrates the power of individual features and their combinations. The mean average precisions(mAP) of all categories on VOC 2010 val are shown. Note that features play different roles in different object categories. Feature combination significantly improves performance on all categories.

	aer	bik	bir	boa	bot	bus	car	cat	cha	cow	din	dog	hor	mot	per	pot	she	sof	tra	tvm
S	.373	.348	.092	.061	.171	.411	.297	.251	.043	.133	.093	.152	.314	.322	.177	.061	.217	.169	.178	.300
P	.420	.361	.167	.108	.171	.585	.321	.266	.117	.218	.193	.272	.325	.433	.255	.150	.308	.260	.350	.422
L	.352	.443	.139	.094	.194	.537	.375	.356	.137	.292	.191	.273	.378	.490	.194	.170	.317	.230	.361	.372
O	.529	.294	.108	.103	.081	.469	.248	.451	.036	.102	.186	.284	.201	.386	.220	.048	.193	.198	.320	.374
SP	.454	.415	.174	.112	.216	.548	.356	.335	.111	.217	.179	.252	.392	.430	.289	.138	.337	.277	.348	.428
SPL	.457	.469	.215	.113	.242	.602	.421	.397	.153	.352	.242	.350	.466	.532	.289	.183	.414	.310	.412	.478
SPLO	.568	.434	.248	.164	.234	.635	.384	.568	.134	.298	.302	.430	.425	.514	.332	.163	.411	.381	.472	.482

Table 5.2: Detection results on VOC 2010 val: In order to better understand the power of each individual feature and their combinations, we run control experiments on the validation set of VOC 2010 and compare performance using different types of features. Note that features contribute differently to different categories, and feature combination is essential for improved performance. S,P,L,O stands for SIFT VQ'ed, poselet VQ'ed, LCC, and Object-centric features, respectively.

	aer	bik	bir	boa	bot	bus	car	cat	cha	cow	din	dog	hor	mot	per	pot	she	sof	tra	tvm
[15]	.332	.519	.085	.082	.348	.390	.488	.222	-	.206	-	.185	.482	.441	.485	.091	.280	.130	.225	.330
[33]	.524	.543	.130	.156	.351	.542	.491	.318	.155	.262	.135	.215	.454	.516	.475	.091	.351	.194	.466	.380
[85]	.582	.419	.192	.140	.143	.448	.367	.488	.129	.281	.287	.394	.441	.525	.258	.141	.388	.342	.431	.426
NLPR	.533	.553	.192	.210	.300	.544	.467	.412	.200	.315	.207	.303	.486	.553	.465	.102	.344	.265	.503	.403
[103]	.542	.485	.157	.192	.292	.555	.435	.417	.169	.285	.267	.309	.483	.550	.417	.097	.358	.308	.472	.408
NUS	.491	.524	.178	.120	.306	.535	.328	.373	.177	.306	.277	.295	.519	.563	.442	.096	.148	.279	.495	.384
Ours	.537	.429	.181	.165	.235	.481	.421	.454	.067	.234	.277	.352	.407	.490	.320	.116	.346	.287	.433	.392

Table 5.3: Detection Results on VOC 2010 test: This table compares our full system with other leading approaches on VOC 2010 test data. Our performance is highly competitive.

Table 5.3 compares the results of our full system with leading approaches on the VOC 2010 test set. Our results are highly competitive to the leading performance on this benchmark.

Multi-Component vs. Monolithic vs. Per Exemplar Models

In addition to knowing where our approach stands in the detection field, we are also interested in knowing how much we benefit from our multi-component architecture. Table 5.4 illustrates control experiments that compare our multi-component model with a monolithic model using the same feature set (SIFT and poselet activations) and positive instances (the positive set of the monolithic model is the set of all aligned objects in component models). We conclude from the table that the multi-component model handles the intra-class variations better than the monolithic model and therefore yields improved results for all categories.

On the other hand, our performance is significantly higher than [69] which uses a single object per component as positive set. This illustrates the generalization power of our model through object alignment with seeds for each component.

Figure 5.6: Visualization of our detection results on PASCAL VOC dataset. Red bounding boxes indicate detection. Figures are the left show correct detection, while figures on the right show some failure cases. Many are due to heavy occlusion/trancation, poor localization, and confusion between similar categories.

(a) aeroplane (b) bicycle (c) bottle

(d) bus (e) cat (f) chair

(g) diningtable (h) dog (i) horse

(j) person (k) sofa (l) train

Figure 5.7: Our multi-component models enable fine-grained visual recognition applications such as keypoint prediction and segmentation, as shown in the figures above. Each detected object in the test image (shown in the top-right of each category) is associated with a "matching" component that assigns the highest detection score to the object. The two figures on the left of each category depict the seed object (with its keypoints marked in blue) and the average mask of the "matching" component. Inference on keypoint locations and mask of the test object is obtained through a transformation between the bounding box of the seed and that of the test object, as well as bottom-up segmentation cues. The two figures on the right of each category show our results, where estimated keypoint locations are marked in pink, and segmentation mask in red.

	aer	bik	bir	boa	bot	bus	car	cat	cha	cow	din	dog	hor	mot	per	pot	she	sof	tra	tvm	avg
MN	.248	.268	.059	.109	.092	.381	.375	.228	.097	.163	.236	.147	.252	.260	.177	.104	.197	.211	.210	.358	.209
MC	.334	.370	.150	.150	.226	.431	.493	.328	.115	.358	.178	.163	.436	.382	.298	.116	.333	.235	.302	.396	.290
[69]	.208	.480	.077	.143	.131	.397	.411	.052	.116	.186	.111	.031	.447	.394	.169	.112	.226	.170	.369	.300	.227

Table 5.4: Detection Results on VOC 2007 test: This table compares the results on VOC 2007 test set between our multi-component(MC) model and the monolithic(MN) model using the same feature set and training data. Note the improvement of multi-component model over monolithic model on almost every category. In addition, our model also outperforms [69] that trains each component model using only one positive instance.

Keypoints and Mask Transfer via Component Models

Like [42, 69], our multi-component models provide more information than just bounding boxes in the object detection framework. The output scores of the first-layer component classifiers also imply which component is most similar in appearance and configuration to a detected object. We refer to the one assigning the highest score to a detection as the "matching" component of that detection. See Figure 5.7 for some examples. Since training objects are tight within a component, a projection of the keypoints and mask of the seed object onto the test image based on the transformation between the seed and the detection windows provides reasonable estimates of the keypoint locations and the mask of the detected object. This is a straight-forward yet useful application which is difficult to obtain from most previous related work.

In fact, we can even do better in object mask prediction through two modifications. First, it is more robust to replace the mask of the seed by the average mask of that component. Second, we use bottom-up segmentation of the detected image to refine our results. Suppose that the final object mask is a selected union of bottom-up segments, our objective is to maximize the intersection-over-union score of the final object mask to the transformed average mask of the matching component. Although the problem is non-linear and exact solution is hard to obtain, we use a greedy selection method to approximate the result. We start with the best single segment among the bottom-up segmentation pool. Then at each step, pick a segment from the pool and merge it with current mask so that it maximizes the intersection-over-union score. The cycle continues until the score starts to decrease. In practice, this approximation works very well, and Figure 5.7 shows our mask prediction results.

5.7 Conclusion

This paper presents a novel multi-component category model for object detection. Each component model characterizes a particular variation of object category, and objects within a component are tight in appearance and configuration. Therefore, a component model is not only easy to learn, but also highly discriminative. A second layer classifier is learned

to fuse the outputs of component models into a final classification score, and a selective bounding box proposal is applied to significantly speed up the detection process. We also show the power of our multi-component model beyond detection - object keypoint and mask prediction, and achieve promising results.

Bibliography

[1] F. Aiolli and A. Sperduti. "Multiclass Classification with Multi-prototype Support Vector Machines". In: *Journal of Machine Learning Research* 6 (2005), pp. 817–850.

[2] B. Alexe, T. Deselaers, and V. Ferrari. "What is an Object?" In: *Computer Vision and Pattern Recognition*. 2010.

[3] S. Andrews, I. Tsochantaridis, and T. Hofmann. "Support Vector Machines for Multiple-instance Learning". In: *Neural Information Processing Systems*. 2003.

[4] P. Arbeláez and L. Cohen. "Constrained Image Segmentation from Hierarchical Boundaries". In: *Computer Vision and Pattern Recognition*. 2008.

[5] P. Arbeláez et al. "Contour Detection and Hierarchical Image Segmentation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33.5 (May 2011), pp. 898–916.

[6] P. Arbeláez et al. "From Contours to Regions: An Empirical Evaluation". In: *Computer Vision and Pattern Recognition*. 2009.

[7] M. Arie-Nachmison and R. Basri. "Constructing Implicit 3D Shape Models for Pose Estimation". In: *International Conference on Computer Vision*. 2009.

[8] D. Ballard. "Generalizing the Hough Transform to Detect Arbitrary Shapes". In: *Pattern Recognition* 13.2 (1981), pp. 111–122.

[9] D. Batra, R. Sukthankar, and T. Chen. "Learning Class-specific Affinities for Image Labelling". In: *Computer Vision and Pattern Recognition*. 2008.

[10] A. Berg, T. Berg, and J. Malik. "Shape Matching and Object Recognition Using Low Distortion Correspondence". In: *Computer Vision and Pattern Recognition*. 2005.

[11] A. Berg and J. Malik. "Geometric Blur and Template Matching". In: *Computer Vision and Pattern Recognition*. 2001.

[12] I. Biederman, R. Mezzanotte, and J. Rabinowitz. "Scene Perception: Detecting and Judging Objects Undergoing Relational Violations". In: *Cognitive Psychology* 14 (1982), pp. 143–177.

[13] O. Boiman, E. Shechtman, and M. Irani. "In Defense of Nearest-neighbor Based Image Classification". In: *Computer Vision and Pattern Recognition*. 2008.

[14] L. Bourdev and J. Malik. "Poselets: Body Part Detectors Trained Using 3D Human Pose Annotations". In: *International Conference on Computer Vision*. 2009.

[15] L. Bourdev et al. "Detecting People Using Mutually Consistent Poselet Activations". In: *European Conference on Computer Vision*. 2010.

[16] T. Brox et al. "Object Segmentation by Alignment of Poselet Activations to Image Contours". In: *Computer Vision and Pattern Recognition*. 2011.

[17] H. Bulthoff and S. Edelman. "Psychophysical Support for a Two-dimensional View Interpolation Theory of Object Recognition". In: *Proceedings of the National Academy of Sciences* 89.1 (1992), pp. 60–64.

[18] C. Chang and C. Lin. *LIBSVM: a Library for Support Vector Machines*. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`. 2001.

[19] H. Chiu, L. Kaelbling, and T. Lozano-Perez. "Virtual-training for Multi-view Object Class Recognition". In: *Computer Vision and Pattern Recognition*. 2007.

[20] A. Collet et al. "Object Recognition and Full Pose Registration from a Single Image for Robotic Manipulation". In: *International Conference on Robotics and Automation*. 2009.

[21] D. Comaniciu and P Meer. "Mean Shift: A Robust Approach toward Feature Space Analysis". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24.5 (2002), pp. 603–619.

[22] C. Cyr and B. Kimia. "A Similarity-based Aspect-graph Approach to 3D Object Recognition". In: *International Journal of Computer Vision* 57.1 (2004), pp. 5–22.

[23] N. Dalal and B. Triggs. "Histograms of Oriented Gradients for Human Detection". In: *Computer Vision and Pattern Recognition*. 2005.

[24] D. DeMenthon and L. Davis. "Model-based Object Pose in 25 Lines of Code". In: *International Journal of Computer Vision* 15 (1995), pp. 123–141.

[25] R. Detry, N. Pugeault, and J. Piater. "A Probabilistic Framework for 3D Visual Object Representation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31.10 (2009), pp. 1790–1803.

[26] R. Duda and P. Hart. "Use of the Hough Transformation to Detect Lines and Curves in Pictures". In: *Communication ACM* 15.1 (1972), pp. 11–15. ISSN: 0001-0782.

[27] I. Endres and D. Hoiem. "Category Independent Object Proposals". In: *European Conference on Computer Vision*. 2010.

[28] M. Everingham et al. *The PASCAL Visual Object Classes Challenge 2006 (VOC2006) Results*. http://www.pascal-network.org/challenges/VOC/voc2006/results.pdf.

[29] M. Everingham et al. "The Pascal Visual Object Classes (VOC) Challenge". In: *International Journal of Computer Vision* 88.2 (June 2010), pp. 303–338.

[30] R. Fan et al. "Liblinear: A Library for Large Linear Classification". In: *Journal of Machine Learning Research* 9 (2008), pp. 1871–1874.

[31] L. Fei-Fei, F. Fergus, and P. Perona. "Learning Generative Visual Models from Few Training Examples: an Incremental Bayesian Approach Testing on 101 Object Categories". In: *Workshop on Generative-Model Based Vision, Computer Vision and Pattern Recognition*. 2004.

[32] P. Felzenszwalb, D. McAllester, and D. Ramanan. "A Discriminatively Trained, Multiscale, Deformable Part Model". In: *Computer Vision and Pattern Recognition*. 2008.

[33] P. Felzenszwalb et al. "Object Detection with Discriminatively Trained Part Based Models". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.9 (September 2010), pp. 1627–1645.

[34] V. Ferrari, F. Jurie, and C. Schmid. "Accurate Object Detection with Deformable Shape Models Learnt from Images". In: *Computer Vision and Pattern Recognition*. 2007.

[35] V. Ferrari, T. Tuytelaars, and L. Van Gool. "Object Detection by Contour Segment Networks". In: *European Conference on Computer Vision*. 2006.

[36] C. Fowlkes, D. Martin, and J. Malik. "The Berkeley Segmentation Dataset and Benchmark (BSDS)". In: URL: `http://www.cs.berkeley.edu/projects/vision/grouping/segbench/`.

[37] A. Frome, Y. Singer, and J. Malik. "Image Retrieval and Classification Using Local Distance Functions". In: *Neural Informaion Processing Systems*. 2006.

[38] A. Frome et al. "Learning Globally-consistent Local Distance Functions for Shape-based Image Retrieval and Classification". In: *International Conference on Computer Vision*. 2007.

[39] S. Gould et al. "Multi-class Segmentation with Relative Location Prior". In: *International Journal of Computer Vision* 80.3 (December 2008), pp. 300–316.

[40] G. Griffin, A. Holub, and P. Perona. *Caltech-256 Object Category Dataset*. Tech. rep. 7694. California Institute of Technology, 2007.

[41] C. Gu and X. Ren. "Discriminative Mixture-of-templates for Viewpoint Classification". In: *European Conference on Computer Vision*. 2010.

[42] C. Gu et al. "Recognition Using Regions". In: *Computer Vision and Pattern Recognition*. 2009.

[43] X. He, R. Zemel, and M. Carreira Perpinan. "Multiscale Conditional Random Fields for Image Labeling". In: *Computer Vision and Pattern Recognition*. 2004.

[44] G. Heitz and D. Koller. "Learning Spatial Context: Using Stuff to Find Things". In: *European Conference on Computer Vision*. 2008.

[45] J. Henderson and A. Hollingworth. "High-level Scene Perception". In: *Annual Review of Psychology* 50.243 (1999), p. 71.

[46] D. Hoiem, A. Efros, and M. Hebert. "Geometric Context from a Single Image". In: *International Conference on Computer Vision*. 2005.

[47] D. Hoiem, C. Rother, and J. Winn. "3D LayoutCRF for Multi-view Object Class Recognition and Segmentation". In: *Computer Vision and Pattern Recognition*. 2007.

[48] J. Koenderink and A. van Doorn. "The Internal Representation of Solid Shape with Respect to Vision". In: *Biological Cybernetics* 32 (1979), pp. 211–6.

[49] P. Kohli, L. Ladicky, and P. Torr. "Robust Higher Order Potentials for Enforcing Label Consistency". In: *Computer Vision and Pattern Recognition*. 2008.

[50] M. Kumar, P. Torr, and A. Zisserman. "Obj Cut". In: *Computer Vision and Pattern Recognition*. 2005.

[51] S. Kumar and M. Hebert. "A Hierarchical Field Framework for Unified Context-based Classification". In: *International Conference on Computer Vision*. 2005.

[52] A. Kushal, C. Schmid, and J. Ponce. "Flexible Object Models for Category-level 3D Object Recognition". In: *Computer Vision and Pattern Recognition*. 2004.

[53] C. Lampert. "Partitioning of Image Datasets Using Discriminative Context Information". In: *Computer Vision and Pattern Recognition*. 2008.

[54] C. Lampert, M. Blaschko, and T. Hofmann. "Beyond Sliding Windows: Object Localization by Efficient Subwindow Search". In: *Computer Vision and Pattern Recognition*. 2008.

[55] S. Lavallee and R. Szeliski. "Recovering the Position and Orientation of Free-form Objects from Image Contours Using 3D Distance Maps". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 17.4 (1995), pp. 378–390.

[56] S. Lazebnik, C. Schmid, and J. Ponce. "Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories". In: *Computer Vision and Pattern Recognition*. 2006.

[57] B. Leibe, A. Leonardis, and B. Schiele. "Combined Object Categorization and Segmentation with an Implicit Shape Model". In: *European Conference on Computer Vision, Workshop on Statistical Learning in Computer Vision*. 2004.

[58] T. Leung and J. Malik. "Representing and Recognizing the Visual Appearance of Materials Using Three-dimensional Textons". In: *International Journal of Computer Vision* 43.1 (June 2001), pp. 29–44.

[59] F. Li, J. Carreira, and C. Sminchisescu. "Object Recognition as Ranking Holistic Figure-ground Hypotheses". In: *Computer Vision and Pattern Recognition*. 2010.

[60] F. Li and P. Perona. "A Bayesian Hierarchical Model for Learning Natural Scene Categories". In: *Computer Vision and Pattern Recognition*.

[61] J. Liebelt, C. Schmid, and K. Schertler. "Viewpoint-independent Object Class Detection Using 3D Feature Maps". In: *Computer Vision and Pattern Recognition*. 2008.

[62] J. Lim et al. "Context as Region Ancestry". In: *International Conference on Computer Vision*. 2009.

[63] Y. Lin et al. "Large-scale Image Classification: Fast Feature Extraction and SVM Training". In: *Computer Vision and Pattern Recognition*. 2011.

[64] D. Lowe. "Distinctive Image Features from Scale-invariant Keypoints". In: *International Journal of Computer Vision* 60.2 (2004), pp. 91–110.

[65] M. Maire et al. "Using Contours to Detect and Localize Junctions in Natural Images". In: *Computer Vision and Pattern Recognition*. 2008.

[66] S. Maji, A. Berg, and J. Malik. "Classification Using Intersection Kernel Support Vector Machine is Efficient". In: *Computer Vision and Pattern Recognition*. 2008.

[67] S. Maji and J. Malik. "Object Detection Using a Max-margin Hough Transform". In: *Computer Vision and Pattern Recognition*. 2009.

[68] T. Malisiewicz and A. Efros. "Recognition by Association via Learning Per-exemplar Distances". In: *Computer Vision and Pattern Recognition*. 2008.

[69] T. Malisiewicz, A. Gupta, and A. Efros. "Ensemble of Exemplar-SVMs for Object Detection and Beyond". In: *International Conference on Computer Vision*. 2011.

[70] C. Manning, P. Raghavan, and H. Schutze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[71] K. Murphy, A. Torralba, and W. Freeman. "Using the Forest to See the Trees: A Graphical Model Relating Features, Objects, and Scenes". In: *Neural Information Processing Systems*. 2003.

[72] A. Oliva and A. Torralba. "Modeling the Shape of the Scene: A Holistic Representation of the Spatial Envelope". In: *International Journal of Computer Vision* 42.3 (2001), pp. 145–175.

[73] A. Opelt, A. Pinz, and A. Zisserman. "A Boundary-fragment-model for Object Detection". In: *European Conference on Computer Vision*. 2006.

[74] M. Ozuysal, V. Lepetit, and P. Fua. "Pose Estimation for Category Specific Multiview Object Localization". In: *Computer Vision and Pattern Recognition*. 2009.

[75] S. Palmer. "The Effects of Contextual Scenes on the Identification of Objects". In: *Memory and Cognition* 3 (1975), pp. 519–526.

[76] C. Pantofaru, C. Schmid, and M. Hebert. "Object Recognition by Integrating Multiple Image Segmentations". In: *European Conference on Computer Vision*. 2008.

[77] D. Parikh, L. Zitnick, and T. Chen. "From Appearance to Context-based Recognition: Dense Labeling in Small Images". In: *Computer Vision and Pattern Recognition*. 2008.

[78] A. Rabinovich et al. "Objects in Context". In: *Internation Conference on Computer Vision*. 2007.

[79] B. Rosenhahn, T. Brox, and J. Weickert. "Three-dimensional Shape Knowledge for Joint Image Segmentation and Pose Tracking". In: *International Journal of Computer Vision* 73.3 (2007), pp. 243–262.

[80] F. Rothganger et al. "3D Object Modeling and Recognition Using Local Affine-invariant Image Descriptors and Multi-view Spatial Constraints". In: *International Journal of Computer Vision* 66.3 (2006), pp. 231–259.

[81] H. Rowley, S. Baluja, and T. Kanade. "Neural Network-based Face Detection". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20 (1998), pp. 23–38.

[82] O. Russakovsky et al. *Object-centric Spatial Pooling for Image Classification*. Tech. rep. 2011-TR112. NEC Labs America, 2011.

[83] B. Russell et al. "Using Multiple Segmentations to Discover Objects and Their Extent in Image Collections". In: *Computer Vision and Pattern Recognition*. 2006.

[84] K. van de Sande, T. Gevers, and C. Snoek. "Evaluating Color Descriptors for Object and Scene Recognition". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.9 (2010), pp. 1582–1596.

[85] K. van de Sande et al. "Segmentation as Selective Search for Object Recognition". In: *International Conference on Computer Vision*. 2011.

[86] S. Savarese and L. Fei-Fei. "3D Generic Object Categorization, Localization and Pose Estimation". In: *International Conference on Computer Vision*. 2007.

[87] C. Schmid and R. Mohr. "Local Grayvalue Invariants for Image Retrieval". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19.5 (1997), pp. 530–535.

[88] J. Shi and J. Malik. "Normalized Cuts and Image Segmentation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.8 (2000), pp. 888–905.

[89] J. Shotton et al. "Semantic Texton Forests for Image Categorization and Segmentation". In: *Computer Vision and Pattern Recognition*. 2008.

[90] J. Shotton et al. "TextonBoost: Joint Appearance, Shape and Context Modeling for Multi-class Object Recognition and Segmentation". In: *European Conference on Computer Vision*. 2006.

[91] T. Strat and M. Fischler. "Context-based Vision: Recognizing Objects Using Information from Both 2-D and 3-D Imagery". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 13.10 (October 1991), pp. 1050–1065.

[92] H. Su et al. "Learning a Dense Multi-view Representation for Detection, Viewpoint Classification and Synthesis of Object Categories". In: *International Conference on Computer Vision*. 2009.

[93] E. Sudderth et al. "Depth from Familiar Objects: A Hierarchical Model for 3D Scenes". In: *Computer Vision and Pattern Recognition*. 2006.

[94] M. Sun et al. "A Multi-view Probabilistic Model for 3D Object Classes". In: *Computer Vision and Pattern Recognition*. 2009.

[95] A. Thomas et al. "Towards Multi-view Object Class Detection". In: *Computer Vision and Pattern Recognition*. 2006.

[96] S. Todorovic and N. Ahuja. "Learning Subcategory Relevances for Category Recognition". In: *Computer Vision and Pattern Recognition*. 2008.

[97] A. Vedaldi and B. Fulkerson. *VLFeat: An Open and Portable Library of Computer Vision Algorithms*. `http://www.vlfeat.org/`. 2008.

[98] A. Vedaldi et al. "Multiple Kernels for Object Detection". In: *Neural Information Processing Systems*. 2009.

[99] J. Verbeek and B. Triggs. "Region Classification with Markov Field Aspect Models". In: *Computer Vision and Pattern Recognition*. 2007.

[100] P. Viola and M. Jones. "Robust Real-time Face Detection". In: *International Journal of Computer Vision* 57.2 (2004), pp. 137–154.

[101] K. Yu, T. Zhang, and Y. Gong. "Nonlinear Learning Using Local Coordinate Coding". In: *Neural Informaion Processing Systems*. 2009.

[102] H. Zhang et al. "SVM-KNN: Discriminative Nearest Neighbor Classification for Visual Category Recognition". In: *Computer Vision and Pattern Recognition*. 2006.

[103] L. Zhu et al. "Latent Hierarchical Structural Learning for Object Detection". In: *Computer Vision and Pattern Recognition*. 2010.