Labeling, Discovering, and Detecting Objects in Images

by

Bryan Christopher Russell

A.B., Computer Science
Dartmouth College (2001)

S.M., Electrical Engineering and Computer Science
Massachusetts Institute of Technology (2003)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Electrical Engineering and Computer Science
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
February 2008

Author .....
Department of Electrical Engineering and Computer Science
January 28, 2007

Certified by .....
William T. Freeman
Professor
Thesis Supervisor

Accepted by .....
Terry P. Orlando
Chairman, Department Committee on Graduate Students

Labeling, Discovering, and Detecting Objects in Images

by

Bryan Christopher Russell

Submitted to the Department of Electrical Engineering and Computer Science
on January 28, 2007, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Electrical Engineering and Computer Science

Abstract

Recognizing the many objects that comprise our visual world is a difficult task. Confounding factors, such as intra-class object variation, clutter, pose, lighting, dealing with never-before seen objects, scale, and lack of visual experience often fool existing recognition systems. In this thesis, we explore three issues that address a few of these factors: the importance of labeled image databases for recognition, the ability to discover object categories from simply looking at many images, and the use of large labeled image databases to efficiently detect objects embedded in scenes. For each of the issues above, we will need to cope with large collections of images.

We begin by introducing *LabelMe*, a large labeled image database collected from users via a web annotation tool. The users of the annotation tool provided information about the identity, location, and extent of objects in images. Through this effort, we have collected about 160,000 images and 200,000 object labels to date. We show that the database spans more object categories and scenes and offers a wider range of appearance variation than most other labeled databases for object recognition. We also provide four useful extensions of the database: (i) resolving synonym ambiguities that arise in the object labels, (ii) recovering object-part relationships, (iii) extracting a depth ordering of the labeled objects in an image, and (iv) providing a semi-automatic process for the fast labeling of images.

We then seek to learn models of objects in the extreme case when no supervision is provided. We draw inspiration from the success of unsupervised topic discovery in text. We apply the Latent Dirichlet Allocation model of Blei et al. to unlabeled images to automatically discover object categories. To achieve this, we employ the visual words representation of images, which is analogous to the words in text. We show that our unsupervised model achieves comparable classification performance to a model trained with supervision on an unseen image set depicting several object classes. We also successfully localize the discovered object classes in images.

While the image representation used for the object discovery process is simple to compute and can distinguish between different object categories, it does not capture explicit spatial information about regions in different parts of the image. We describe a procedure for combining image segmentation with the object discovery process to

provide increased spatial support. We show that this procedure finds object models with improved performance on the categorization and localization task for images depicting scenes containing multiple objects.

We then return to the problem of learning about objects with supervision. We describe a system that makes efficient use of a large labeled database of images for detecting objects embedded in a scene. The system first aligns the components of a scene depicted in an input image to the images in the database. It then makes use of a simple model that combines an object detector with the object knowledge contained in the labels corresponding to the aligned images. The simplicity of the model allows learning for a large number of object classes embedded in many different scenes. We demonstrate improved classification and localization performance over a standard object detector. Furthermore, our system restricts the object search space and therefore greatly increases computational efficiency.

Thesis Supervisor: William T. Freeman
Title: Professor

Acknowledgments

It is unfortunate that there can only be a single author listed on this thesis, as many others have contributed significantly to this work through collaboration, mentorship, support, inspiration, and encouragement. I consider myself fortunate and am grateful to be surrounded by such a community.

First, I would like to thank my advisors, Bill Freeman and Antonio Torralba, for providing an amazing atmosphere to learn and explore. They taught me how to do research, think about hard problems, and effectively work with others. In addition, I admire and am grateful for their creativity, generosity, humbleness, hard work ethic, and ability to balance work and life.

Many thanks go to my thesis readers: Josh Tenenbaum, Alyosha Efros, Bill, and Antonio. I am indebted to them for providing guidance, asking hard questions at the defense, and providing helpful feedback.

I have been blessed with a rich set of collaborators. The late night brainstorming e-mails with Antonio for the LabelMe project was fun. I could always count on Antonio for excellent ideas and feedback. In addition, Kevin Murphy and Bill gave good advice. Antonio provided much enthusiasm and support for this work and deserves equal (if not more) credit.

Collaborating with Josef Sivic, Alyosha Efros, Andrew Zisserman, and Bill on the object discovery work was also great fun. I will always treasure the good-natured humor and banter that comprised each N-way conference call (often the conference call configuration was different due to travel). Above all else, the free-flow of ideas and overall unselfish nature of the group made the entire experience a rewarding one.

Working with Antonio, Ce Liu, Rob Fergus, and Bill on the *Object Recognition by Scene Alignnment* project was a delight. Our spatial proximity in the lab and close friendship made the ideas for this project come easily.

I have been very fortunate to share an office with so many wonderful people: Marshall Tappen, Barun Singh, Mike Siracusa, Ce, Ali Rahimi, Alex Berg, and Clare Poynton. Thanks for all of the wonderful conversations and for being such an integral

part of my graduate school experience. Also, thank you Ce for your kindness, compassion, and for always challenging me to think about the hard problems in computer vision.

Big thanks go to Bryt Bradley and Maysoon Hamdiyyah for having all the answers (even to crossword puzzles!) and ensuring that things could get done.

The students and affiliated researchers in the lab provided a fun source of distraction, whether it was through late night poker games, parties, or sports: Kilian, Neal, Biz, Wanmei, Kevin, Mike, Mario, Kristen, Kinh, Lilla, Ce, Louis-Phillippe, Thomas, Harold, Jacob, Greg, Erik, Rob, John, Kate, Anat, Biliana, Tim, Hyun-Sung, along with many others. Also, thanks to my dear friends and roommmates who helped me get through the thesis crunch: Rodney, Heather, Ashish, Liz, Nisha, Brenda, Vanessa, and Mr. Fish. The UniLu community was an integral part of my life in Cambridge. Thanks for your support and for teaching me about the things that really matter: social justice issues, spirituality, love, and community.

Finally, Kathryn, Mom, Dad, Lisa, and Kevin: your love, support, and guidance has influenced and touched me more deeply than you could ever know.

This research was performed at the Computer Science and Artificial Intelligence Laboratory at MIT and was supported by the National Science Foundation Grant No. 0413232, the National Geospatial-Intelligence Agency NEGI-1582-04-0004, the Office of Naval Research MURI Grant N00014-06-1-0734, the ARDA VACE program, the Canadian NSERC Discovery Grant program, the Canadian Institute For Advanced Research, the EC PASCAL Network of Excellence IST-2002-506778, a grant from BAE Systems, EC Project CLASS, and NSF CAREER award IIS-0546547.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Consider the images in Figure 1-1. What objects are depicted in the images? Where are they located? Do you see an object that you have never seen before? If so, can you find it in other images? Humans can answer these questions with relative ease. Computers cannot do this yet.

Recognizing the many objects that comprise our visual world is a difficult task. The objects depicted in the images undergo various transformations in appearance due to pose, lighting, and scale. Often, the objects are embedded in clutter within the scenes, sometimes with other previously unseen objects. There is also a large number of object categories, with each category having a wide variation in appearance.

A recognition system must address these issues if there is any hope of achieving or exceeding human-level performance. The importance of this is evident when we examine the complex world in which we live. In order to effectively navigate in an environment, one must recognize and know how to interact with the often many objects that live in the environment. Processing and interpreting visual data is one way of achieving this.

This thesis explores three areas that attempt to address a few of the above issues: (i) the role and importance of labeled image databases for recognition, (ii) what can be learned from simply looking at images, and (iii) ways of exploiting large labeled image databases to detect objects embedded in scenes.

Figure 1-1: Illustration of the diverse range of objects and how they live in the world.

1.1 Overview of Methods and Contributions

We give a brief overview of the areas developed in this thesis and highlight our contributions.

1.1.1 Collecting a Large Labeled Database of Images

Visual experience is important if we want computer vision systems to be able to recognize objects. For this, we need to have seen examples and know the identity of many objects classes and how they are embedded in the world. This information can also be important in training and validating recognition systems. The first contribution of this thesis is a large labeled image database, called *LabelMe*, that was collected from online users via a web annotation tool. The users of the annotation tool provided information about the identity, location, and extent of objects in images. Through this effort, we have collected almost 200,000 labels to date. Given the collection of images and labels, we show the contents and quality of the database. We also compare with existing labeled databases that are used for object recognition.

We also provide four useful extensions of the database. The first is a way of resolving synonym ambiguities that arise in the object labels. We accomplish this by mapping the labels to semantic meanings provided by an electronic dictionary. The second is analyzing the co-occurrences of the labels in the images to automatically recover object-part relationships. This can be useful for part-based recognition systems. The third is the use of a simple heuristic to extract a depth ordering of the labeled objects in an image. This can be used to provide ground-truth for occlusions in an image and be useful for systems reasoning about occlusions. The fourth is a semi-automatic process for labeling images by bootstrapping from the existing object labels in the database. The system presents automatically segmented objects for which the user only needs to verify, which greatly speeds up the labeling process.

1.1.2 Discovering Objects and their Location in Images

While it is useful to have many labeled images for training a recognition system, it is also important to consider the ability to reason about unfamiliar objects. The second contribution of this thesis is a study of what can be learned about objects in the extreme case when no supervision is provided. We draw inspiration from the text understanding community, where semantic models were developed to discover latent topics in text. In our case, we wish to automatically discover object categories in a stack of unlabeled images. To achieve this, we employ a representation of images that is analogous to text documents. This allows us to directly apply the semantic models to a set of images. For our analysis, we choose the Latent Dirichlet Allocation (LDA) model [14]. We show the outputs of the model on a set of images depicting primarily a single object against background clutter. We also analyze categorization performance and show localization results of the discovered object categories.

While the image representation used for the object discovery process is simple to compute and can distinguish between a few different object categories, it unfortunately does not capture explicit spatial information about regions in different parts of the image. Also, the representation often fails to distinguish between different semantically meaningful regions. These issues limit the ability to effectively discover

many different objects embedded in a scene.

The third contribution of this thesis is a procedure for combining image segmentation with the object discovery process. The image segmenter parcels an image into visually coherent fragments. While the image segmentation is not perfect, as it often splits objects or includes multiple objects in the same image fragment, we observe that it cleanly separates different object categories much of the time. With this, we get a close approximation to the situation for which object discovery is successful. To visualize our proposed procedure, we describe a method of selecting the best image fragments that represent the discovered object category. We show montages of high scoring image fragments for each discovered category and quantify the goodness of the overall procedure for a few different image sets depicting more complicated scenes.

1.1.3 Detecting Objects in Images through Scene Alignment

After considering the case of learning about objects without supervision, we return to the situation where we have supervision. The fourth contribution of this thesis is a system for detecting objects embedded in a scene. Our system aligns the components of a scene depicted in an input image to a large database of labeled images. The scenes that align best in the image database are used to assist in detecting objects in the input image. The scene alignment offers the advantage that we do not have to model various complex interactions that may exist between objects embedded in a scene. To overcome incorrect scene matches, we develop a model for clustering the database images based on their labels. We subsequently combine the knowledge gleaned from the clusters with trained object detectors to detect objects. We show system outputs and validate on the LabelMe database.

1.2 Thesis Organization

In Chapter 2, we describe the LabelMe database and annotation tool and examine its contents and quality. We also explore extensions of the database and compare it to existing databases used for object recognition. Chapter 3, we consider the problem of

unsupervised object discovery in images. We describe the representation and model used for object discovery. We examine and quantify its outputs. In Chapter 4, we look to overcome the image representational issues by augmenting the object discovery process with image segmentation. We visualize and quantify the discovered categories. We describe the model for supervised object detection in Chapter 5 and show its performance. Finally, in Chapter 6 we conclude.

Chapter 2

LabelMe: a database and web-based tool for image annotation

2.1 Introduction

Thousands of objects occupy the visual world in which we live. Biederman [12] estimates that humans can recognize about 30000 entry-level object categories. Recent work in computer vision has shown impressive results for the detection and recognition of a few different object categories [98, 38, 51]. However, the size and contents of existing datasets, among other factors, limit current methods from scaling to thousands of object categories. Research in object detection and recognition would benefit from large image and video collections with ground truth labels spanning many different object categories in cluttered scenes. For each object present in an image, the labels should provide information about the object's identity, shape, location, and possibly other attributes such as pose.

By analogy with the speech and language communities, history has shown that performance increases dramatically when more labeled training data is made available. One can argue that this is a limitation of current learning techniques, resulting

in the recent interest in Bayesian approaches to learning [26, 83] and multi-task learning [92]. Nevertheless, even if we can learn each class from just a small number of examples, there are still many classes to learn.

Large image datasets with ground truth labels are useful for supervised learning of object categories. Many algorithms have been developed for image datasets where all training examples have the object of interest well-aligned with the other examples [95, 38, 98]. Algorithms that exploit context for object recognition [89, 41] would benefit from datasets with many labeled object classes embedded in complex scenes. Such datasets should contain a wide variety of environments with annotated objects that co-occur in the same images.

When comparing different algorithms for object detection and recognition, labeled data is necessary to quantitatively measure their performance (the issue of comparing object detection algorithms is beyond the scope of this chapter; see [4, 49] for relevant issues). Even algorithms requiring no supervision [77, 67, 26, 83, 82, 65] need this quantitative framework.

Building a large dataset of annotated images with many objects is a costly and lengthy enterprise. Traditionally, datasets are built by a single research group and are tailored to solve a specific problem. Therefore, many currently available datasets only contain a small number of classes, such as faces, pedestrians, and cars. Notable exceptions are the Caltech 101 dataset [27], with 101 object classes (this was recently extended to 256 object classes [35]), the PASCAL collection [24], and the CBCL-streetscenes database [13].

We wish to collect a large dataset of annotated images. To achieve this, we consider web-based data collection methods. Web-based annotation tools provide a way of building large annotated datasets by relying on the collaborative effort of a large population of users [99, 73, 70, 81]. Recently, such efforts have had much success. The Open Mind Initiative [81] aims to collect large datasets from web users so that intelligent algorithms can be developed. More specifically, common sense facts are recorded (e.g. red is a primary color), with over 700K facts recorded to date. This project is seeking to extend their dataset with speech and handwriting data.

Flickr [73] is a commercial effort to provide an online image storage and organization service. Users often provide textual tags to provide a caption of depicted objects in an image. Another way lots of data has been collected is through an online game that is played by many users. The ESP game [99] pairs two random online users who view the same target image. The goal is for them to try to "read each other's mind" and agree on an appropriate name for the target image as quickly as possible. This effort has collected over 10 million image captions since 2003, with the images randomly drawn from the web. While the amount of data collected is impressive, only caption data is acquired. Another game, Peekaboom [100] has been created to provide location information of objects. While location information is provided for a large number of images, often only small discriminant regions are labeled and not entire object outlines.

In this chapter we describe LabelMe, a database and an online annotation tool that allows the sharing of images and annotations. The online tool provides functionalities such as drawing polygons, querying images, and browsing the database. In the first part of the chapter we describe the annotation tool and dataset and provide an evaluation of the quality of the labeling. In the second part of the chapter we present a set of extensions and applications of the dataset. In this section we see that a large collection of labeled data allows us to extract interesting information that was not directly provided during the annotation process. In the third part we compare the LabelMe dataset against other existing datasets commonly used for object detection and recognition. The results in this chapter were developed in collaboration with Antonio Torralba (who contributed equally), Kevin P. Murphy, and William T. Freeman and will appear in a special issue of the International Journal of Computer Vision [69]

2.2 LabelMe

In this section we describe the details of the annotation tool and the results of the online collection effort.

2.2.1 Goals of the LabelMe project

There are a large number of publically available databases of visual objects [92, 4, 50, 63, 25, 27, 28, 35, 18, 52, 48, 17]. We do not have space to review them all here. However, we give a brief summary of the main features that distinguishes the LabelMe dataset from other datasets.

- Designed for object class recognition as opposed to instance recognition. To recognize an object class, one needs multiple images of different instances of the same class, as well as different viewing conditions. Many databases, however, only contain different instances in a canonical pose.

- Designed for learning about objects embedded in a scene. Many databases consist of small cropped images of object instances. These are suitable for training patch-based object detectors (such as sliding window classifiers), but cannot be used for training detectors that exploit contextual cues.

- High quality labeling. Many databases just provide captions, which specify that the object is present somewhere in the image. However, more detailed information, such as bounding boxes, polygons or segmentation masks, is tremendously helpful.

- Many diverse object classes. Many databases only contain a small number of classes, such as faces, pedestrians and cars (a notable exception is the Caltech 101 database, which we compare against in Section 2.4).

- Many diverse images. For many applications, it is useful to vary the scene type (e.g. nature, street, and office scenes), distances (e.g. landscape and close-up shots), degree of clutter, etc.

- Many non-copyrighted images. For the LabelMe database most of the images were taken by the contributing authors of this chapter using a variety of hand-held digital cameras. We also have many video sequences taken with a head-mounted web camera.

- Open and dynamic. The LabelMe database is designed to allow collected labels to be instantly shared via the web and to grow over time.

2.2.2 The LabelMe web-based annotation tool

The goal of the annotation tool is to provide a drawing interface that works on many platforms, is easy to use, and allows instant sharing of the collected data. To achieve this, we designed a Javascript drawing tool, as shown in Figure 2-1. When the user enters the page, an image is displayed. The image comes from a large image database covering a wide range of environments and several hundred object categories. The user may label a new object by clicking control points along the object's boundary. The user finishes by clicking on the starting control point. Upon completion, a popup dialog bubble will appear querying for the object name. The user freely types in the object name and presses enter to close the bubble. This label is recorded on the LabelMe server and is displayed on the presented image. The label is immediately available for download and is viewable by subsequent users who visit the same image.

The user is free to label as many objects depicted in the image as they choose. When they are satisfied with the number of objects labeled in an image, they may proceed to label another image from a desired set or press the *Show Next Image* button to see a randomly chosen image. Often, when a user enters the page, labels will already appear on the image. These are previously entered labels by other users. If there is a mistake in the labeling (either the outline or text label is not correct), the user may either edit the label by renaming the object or delete and redraw along the object's boundary. Users may get credit for the objects that they label by entering a username during their labeling session. This is recorded with the labels that they provide. The resulting labels are stored in the XML file format, which makes the annotations portable and easy to extend.

The annotation tool design choices emphasizes simplicity and ease of use. However, there are many concerns with this annotation collection scheme. One important concern is quality control. Currently quality control is provided by the users themselves, as outlined above. Another issue is the complexity of the polygons provided

Figure 2-1: A screenshot of the labeling tool in use. The user is shown an image along with possibly one or more existing annotations, which are drawn on the image. The user has the option of annotating a new object by clicking along the boundary of the desired object and indicating its identity, or editing an existing annotation. The user may annotate as many objects in the image as they wish.

by the users (i.e. do users provide simple or complex polygon boundaries?). Another issue is what to label. For example, should one label the entire body, just the head, or just the face of a pedestrian? What if it is a crowd of people? Should all of the people be labeled? We leave these decisions up to each user. In this way, we hope the annotations will reflect what various people think are natural ways of segmenting an image. Finally, there is the text label itself. For example, should the object be labeled as a "person", "pedestrian", or "man/woman"? An obvious solution is to provide a drop-down menu of standard object category names. However, we prefer to let people use their own descriptions since these may capture some nuances that will be useful in the future. In Section 2.3.1, we describe how to cope with the text label variability via WordNet [30]. All of the above issues are revisited, addressed, and quantified in the remaining sections.

A Matlab toolbox has been developed to manipulate the dataset and view its contents. Example functionalities that are implemented in the toolbox allow dataset queries, communication with the online tool (this communication can in fact allow one to only download desired parts of the dataset), image manipulations, and other dataset extensions (see Section 2.3).

The images and annotations are organized online into folders, with the folder names providing information about the image contents and location of the depicted scenes/objects. The folders are grouped into two main categories: static pictures and sequences extracted from video. Note that the frames from the video sequences are treated as independent static pictures and that ensuring temporally consistent labeling of video sequences is beyond the scope of this chapter. Most of the images have been taken by the authors using a variety of digital cameras. A small proportion of the images are contributions from users of the database or come from the web. The annotations come from two different sources: the LabelMe online annotation tool and annotation tools developed by other research groups. We indicate the sources of the images and annotations in the folder name and in the XML annotation files. For all statistical analyses that appear in the remaining sections, we will specify which subset of the database subset was used.

2.2.3 Content and evolution of the LabelMe database

We summarize the content of the LabelMe database as of December 21, 2006. The database consists of 111490 polygons, with 44059 polygons annotated using the online tool and 67431 polygons annotated offline. There are 11845 static pictures and 18524 sequence frames with at least one object labeled.

As outlined above, a LabelMe description corresponds to the raw string entered by the user to define each object. Despite the lack of constraint on the descriptions, there is a large degree of consensus. Online labelers entered 2888 different descriptions for the 44059 polygons (there are a total of 4210 different descriptions when considering the entire dataset). Figure 2-2(a) shows a sorted histogram of the number of instances of each object description for all 111490 polygons[1]. Notice that there are many object descriptions with a large number of instances. While there is much agreement among the entered descriptions, object categories are nonetheless fragmented due to plurals, synonyms, and description resolution (e.g. "car", "car occluded", and "car side" all refer to the same category). In section 2.3.1 we will address the issue of unifying the terminology to properly index the dataset according to real object categories.

Figure 2-2(b) shows a histogram of the number of annotated images as a function of the percentage of pixels labeled per image. The graph shows that 11571 pictures have less than 10% of the pixels labeled and around 2690 pictures have more than 90% of labeled pixels. There are 4258 images with at least 50% of the pixels labeled. Figure 2-2(c) shows a histogram of the number of images as a function of the number of objects in the image. There are, on average, 3.3 annotated objects per image over the entire dataset. There are 6876 images with at least 5 objects annotated. Figure 2-3 shows images depicting a range of scene categories, with the labeled objects colored to match the extent of the recorded polygon. For many images, a large number of objects are labeled, often spanning the entire image.

[1] A partial list of the most common descriptions for all 111490 polygons in the LabelMe dataset, with counts in parenthesis: person walking (25330), car (6548), head (5599), tree (4909), window (3823), building (2516), sky (2403), chair (1499), road (1399), bookshelf (1338), trees (1260), sidewalk (1217), cabinet (1183), sign (964), keyboard (949), table (899), mountain (823), car occluded (804), door (741), tree trunk (718), desk (656).

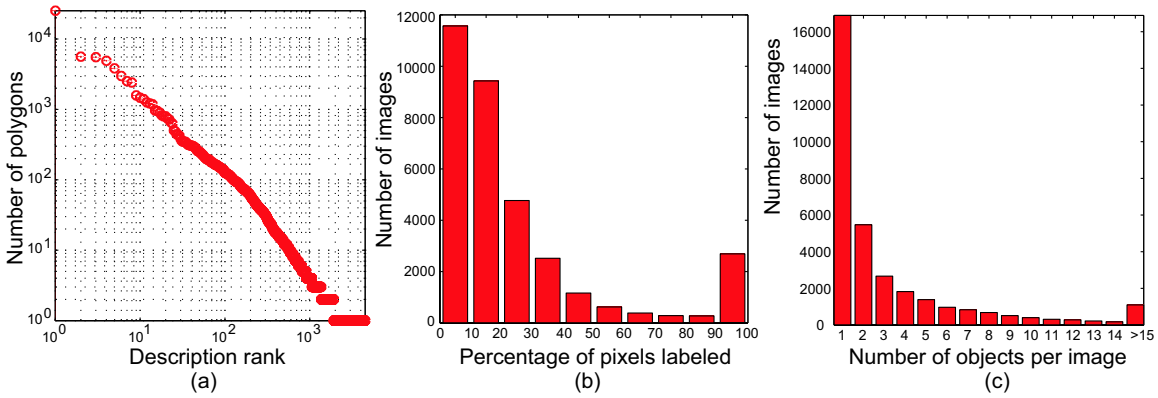


Figure 2-2: Summary of the database content. (a) Sorted histogram of the number of instances of each object description. Notice that there is a large degree of consensus with respect to the entered descriptions. (b) Histogram of the number of annotated images as a function of the area labeled. The first bin shows that 11571 images have less than 10% of the pixels labeled. The last bin shows that there are 2690 pictures with more than 90% of the pixels labeled. (c) Histogram of the number of labeled objects per image.

Figure 2-3: Examples of annotated scenes. These images have more than 80% of their pixels labeled and span multiple scene categories. Notice that many different object classes are labeled per image.

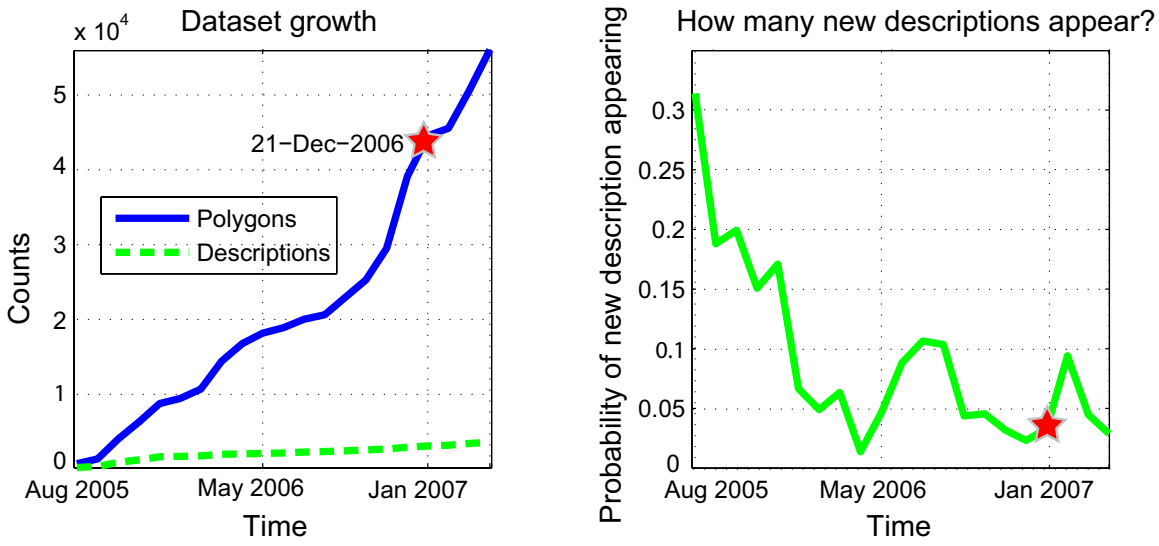


Figure 2-4: Evolution of the online annotation collection over time. Left: total number of polygons (blue, solid line) and descriptions (green, dashed line) in the LabelMe dataset as a function of time. Right: the probability of a new description being entered into the dataset as a function of time. Note that the graph plots the evolution through March 23rd, 2007 but the analysis in this paper corresponds to the state of the dataset as of December 21, 2006, as indicated by the star. Notice that the dataset has steadily increased while the rate of new descriptions entered has decreased.

The web-tool allows the dataset to continuously grow over time. Figure 2-4 depicts the evolution of the dataset since the annotation tool went online. We show the number of new polygons and text descriptions entered as a function of time. For this analysis, we only consider the 44059 polygons entered using the web-based tool. The number of new polygons increased steadily while the number of new descriptions grew at a slower rate. To make the latter observation more explicit, we also show the probability of a new description appearing as a function of time (we analyze the raw text descriptions).

2.2.4 Quality of the polygonal boundaries

Figure 2-5 illustrates the range of variability in the quality of the polygons provided by different users for a few object categories. For the analysis in this section, we only use the 44059 polygons provided online. For each object category, we sort

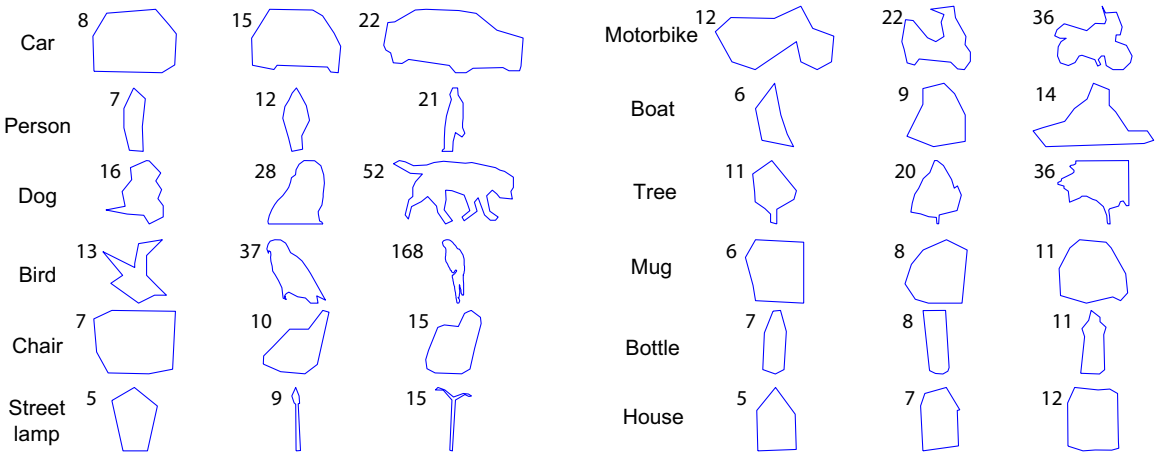


Figure 2-5: Illustration of the quality of the annotations in the dataset. For each object we show three polygons depicting annotations corresponding to the 25th, 50th, and 75th percentile of the number of control points recorded for the object category. Therefore, the middle polygon corresponds to the average complexity of a segmented object class. The number of points recorded for a particular polygon appears near the top-left corner of each polygon. Notice that, in many cases, the object's identity can be deduced from its silhouette, often using a small number of control points.

the polygons according to the number of control points. Figure 2-5 shows polygons corresponding to the 25th, 50th, and 75th percentile with respect to the range of control points clicked for each category. Many objects can already be recognized from their silhouette using a small number of control points. Note that objects can vary with respect to the number of control points to indicate its boundary. For instance, a computer monitor can be perfectly described, in most cases, with just four control points. However, a detailed segmentation of a pedestrian might require 20 control points.

Figure 2-6 shows some examples of cropped images containing a labeled object and the corresponding recorded polygon.

2.2.5 Distributions of object location and size

At first, one would expect objects to be uniformly distributed with respect to size and image location. For this to be true, the images should come from a photographer who randomly points their camera and ignores the scene. However, most of the images in the LabelMe dataset were taken by a human standing on the ground and pointing

Figure 2-6: Image crops of labeled objects and their corresponding silhouette, as given by the recorded polygonal annotation. Notice that, in many cases, the polygons closely follow the object boundary. Also, many diverse object categories are contained in the dataset.

their camera towards interesting parts of a scene. This causes the location and size of the objects to not be uniformly distributed in the images. Figure 2-7 depicts, for a few object categories, a density plot showing where in the image each instance occurs and a histogram of object sizes, relative to the image size. Given how most pictures were taken, many of the cars can be found in the lower half region of the images. Note that for applications where it is important to have uniform prior distribitions of object locations and sizes, we suggest cropping and rescaling each image randomly.

2.3 Extending the dataset

We have shown that the LabelMe dataset contains a large number of annotated images, with many objects labeled per image. The objects are often carefully outlined using polygons instead of bounding boxes. These properties allow us to extract from the dataset additional information that was not provided directly during the labeling process. In this section we provide some examples of interesting extensions of the dataset that can be achieved with minimal user intervention. Code for these applications is available as part of the Matlab toolbox.

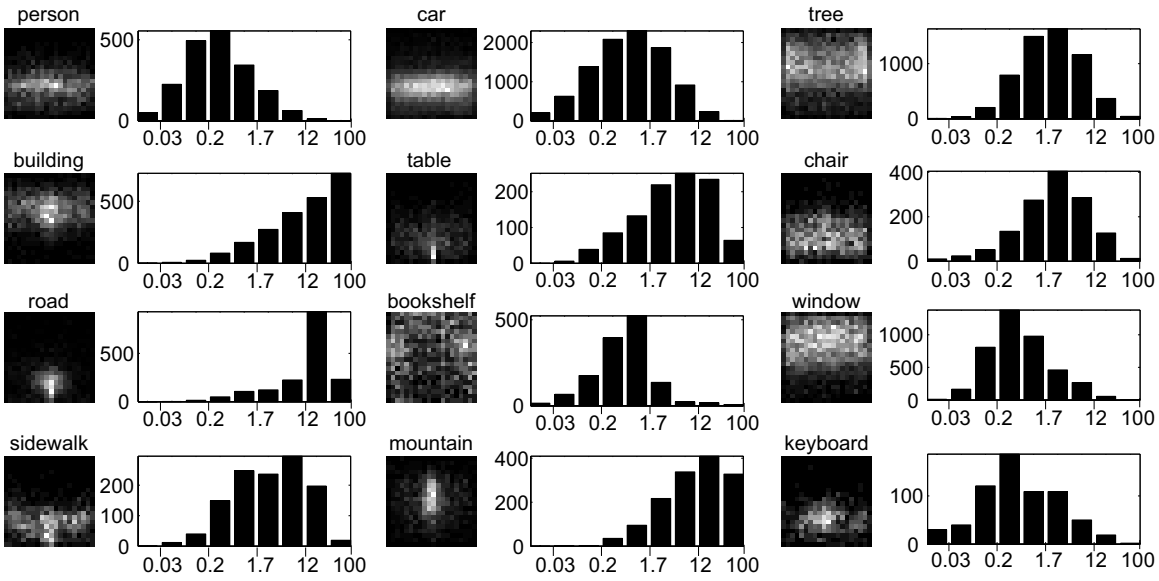


Figure 2-7: Distributions of object location and size for a number of object categories in the LabelMe dataset. The distribution of locations are shown as a 2D histogram of the object centroid location in the different images (coordinates are normalized with respect to the image size). The size histogram illustrates what is the typical size that the object has in the LabelMe dataset. The horizontal axis is in logarithmic units and represents the percentage of the image area occupied by the object.

2.3.1 Enhancing object labels with WordNet

Since the annotation tool does not restrict the text labels for describing an object or region, there can be a large variance of terms that describe the same object category. For example, a user may type any of the following to indicate the "car" object category: "car", "cars", "red car", "car frontal", "automobile", "suv", "taxi", etc. This makes analysis and retrieval of the labeled object categories more difficult since we have to know about synonyms and distinguish between object identity and its attributes. A second related problem is the level of description provided by the users. Users tend to provide basic-level labels for objects (e.g. "car", "person", "tree", "pizza"). While basic-level labels are useful, we would also like to extend the annotations to incorporate superordinate categories, such as "animal", "vehicle", and "furniture".

We use WordNet [30], an electronic dictionary, to extend the LabelMe descriptions. WordNet organizes semantic categories into a tree such that nodes appearing along a

branch are ordered, with superordinate and subordinate categories appearing near the root and leaf nodes, respectively. The tree representation allows disambiguation of different senses of a word (polysemy) and relates different words with similar meanings (synonyms). For each word, WordNet returns multiple possible senses, depending on the location of the word in the tree. For instance, the word "mouse" returns four senses in WordNet, two of which are "computer mouse" and "rodent"[2]. This raises the problem of sense disambiguation. Given a LabelMe description and multiple senses, we need to decide what the correct sense is.

WordNet can be used to automatically select the appropriate sense that should be assigned to each description [43]. However, polysemy can prove challenging for automatic sense assignment. Polysemy can be resolved by analyzing the context (i.e. which other objects are present in the same image). To date, we have not found instances of polysemy in the LabelMe dataset (i.e. each description maps to a single sense). However, we found that automatic sense assignment produced too many errors. To avoid this, we allow for offline manual intervention to decide which senses correspond to each description. Since there are fewer descriptions than polygons (c.f. Figure 2-4), the manual sense disambiguation can be done in a few hours for the entire dataset.

We extended the LabelMe annotations by manually creating associations between the different text descriptions and WordNet tree nodes. For each possible description, we queried WordNet to retrieve a set of senses, as described above. We then chose among the returned senses the one that best matched the description. Despite users entering text without any quality control, 3916 out of the 4210 (93%) unique LabelMe descriptions found a WordNet mapping, which corresponds to 104740 out of the 111490 polygon descriptions. The cost of manually specifying the associations is negligible compared to the cost of entering the polygons and must be updated pe-

[2]The WordNet parents of these terms are (i) *computer mouse*: electronic device; device; instrumentality, instrumentation; artifact, artifact; whole, unit; object, physical object; physical entity; entity and (ii) *rodent*: rodent, gnawer, gnawing animal; placental, placental mammal, eutherian, eutherian mammal; mammal, mammalian; vertebrate, craniate; chordate; animal, animate being, beast, brute, creature, fauna; organism, being; living thing, animate thing; object, physical object; physical entity; entity.

person (27719 polygons)		car (10137 polygons)	
Label	Polygon count	Label	Polygon count
person walking	25330	car	6548
person	942	car occluded	804
person standing	267	car rear	584
person occluded	207	car side	514
person sitting	120	car crop	442
pedestrian	121	car frontal	169
man	117	taxi	8
woman	75	suv	4
child	11	cab	3
girl	9	automobile	2

Table 2.1: Examples of LabelMe descriptions returned when querying for the objects "person" and "car" after extending the labels with WordNet (not all of the descriptions are shown). For each description, the counts represents the number of returned objects that have the corresponding description. Note that some of the descriptions do not contain the query words.

riodically to include the newest descriptions. Note that it may not be necessary to frequently update these associations since the rate of new descriptions entered into LabelMe decreases over time (c.f. Figure 2-4).

We show the benefit of adding WordNet to LabelMe to unify the descriptions provided by the different users. Table 2.1 shows examples of LabelMe descriptions that were returned when querying for "person" and "car" in the WordNet-enhanced framework. Notice that many of the original descriptions did not contain the queried word. Figure 2-8 shows how the number of polygons returned by one query (after extending the annotations with WordNet) are distributed across different LabelMe descriptions. It is interesting to observe that all of the queries seem to follow a similar law (linear in a log-log plot).

Table 2.2 shows the number of returned labels for several object queries before and after applying WordNet. In general, the number of returned labels increases after applying WordNet. For many specific object categories this increase is small, indicating the consistency with which that label is used. For superordinate categories, the number of returned matches increases dramatically. The object labels shown in Table 2.2 are representative of the most frequently occurring labels in the dataset.

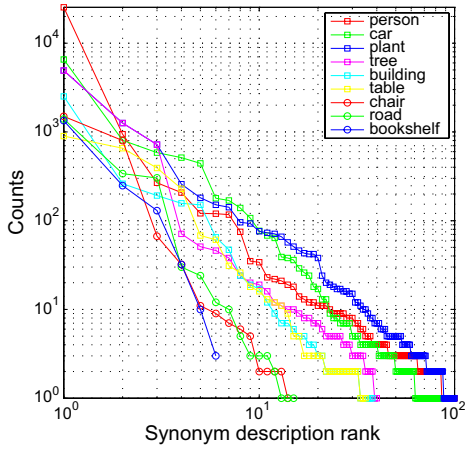


Figure 2-8: How the polygons returned by one query (in the WordNet-enhanced framework) are distributed across different descriptions. The distributions seem to follow a similar law: a linear decay in a log-log plot with the number of polygons for each different description on the vertical axis and the descriptions (sorted by number of polygons) on the horizontal axis. Table 2.1 shows the actual descriptions for the queries "person" and "car".

One important benefit of including the WordNet hierarchy into LabelMe is that we can now query for objects at various levels of the WordNet tree. Figure 2-9 shows examples of queries for superordinate object categories. Very few of these examples were labeled with a description that matches the superordinate category, but nonetheless we can find them.

While WordNet handles most ambiguities in the dataset, errors may still occur when querying for object categories. The main source of error arises when text descriptions get mapped to an incorrect tree node. While this is not very common, it can be easily remedied by changing the text label to be more descriptive. This can also be used to clarify cases of polysemy, which our system does not yet account for.

2.3.2 Object-parts hierarchies

When two polygons have a high degree of overlap, this provides evidence of either (i) an object-part hierarchy or (ii) an occlusion. We investigate the former in this section and the latter in Section 2.3.3.

We propose the following heuristic to discover semantically meaningful object-part

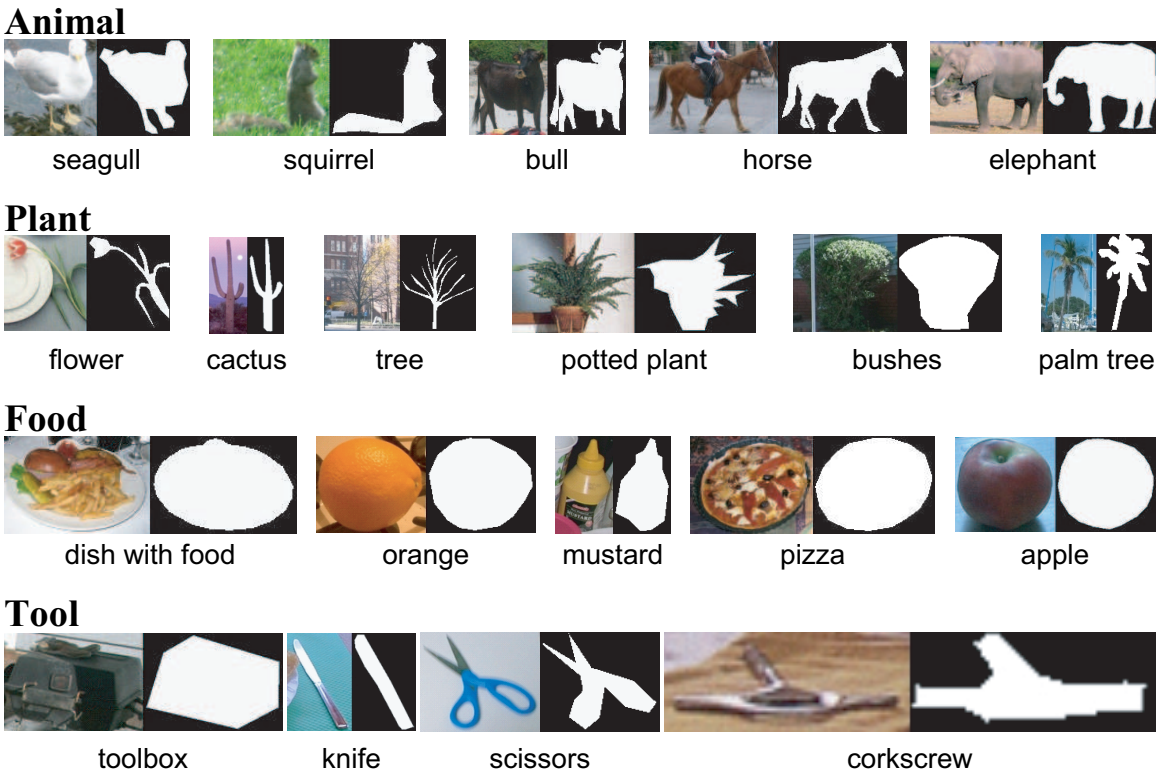


Figure 2-9: Queries for superordinate object categories after incorporating WordNet. Very few of these examples were labeled with a description that matches the superordinate category (the original LabelMe descriptions are shown below each image). Nonetheless, we are able to retrieve these examples.

Category	Original description	WordNet description
person	27019	27719
car	10087	10137
tree	5997	7355
chair	1572	2480
building	2723	3573
road	1687	2156
bookshelf	1588	1763
animal	44	887
plant	339	8892
food	11	277
tool	0	90
furniture	7	6957

Table 2.2: Number of returned labels when querying the original descriptions entered into the labeling tool and the WordNet-enhanced descriptions. In general, the number of returned labels increases after applying WordNet. For entry-level object categories this increase is relatively small, indicating the consistency with which the corresponding description was used. In contrast, the increase is quite large for superordinate object categories. These descriptions are representative of the most frequently occurring descriptions in the dataset.

relationships. Let I_O denote the set of images containing a query object (e.g. car) and $I_P \subseteq I_O$ denote the set of images containing part P (e.g. wheel). Intuitively, for a label to be considered as a part, the label's polygons must consistently have a high degree of overlap with the polygons corresponding to the object of interest when they appear together in the same image. Let the overlap score between an object and part polygons be the ratio of the intersection area to the area of the part polygon. Ratios exceeding a threshold of 0.5 get classified as having high overlap. Let $I_{O,P} \subseteq I_P$ denote the images where object and part polygons have high overlap. The object-part score for a candidate label is $N_{O,P}/(N_P + \alpha)$ where $N_{O,P}$ and N_P are the number of images in $I_{O,P}$ and I_P respectively and α is a concentration parameter, set to 5. We can think of α as providing pseudocounts and allowing us to be robust to small sample sizes.

The above heuristic provides a list of candidate part labels and scores indicating how well they co-occur with a given object label. In general, the scores give good candidate parts and can easily be manually pruned for errors. Figure 2-10 shows

examples of objects and proposed parts using the above heuristic. We can also take into account viewpoint information and find parts, as demonstrated for the car object category. Notice that the object-parts are semantically meaningful.

Once we have discovered candidate parts for a set of objects, we can assign specific part instances to their corresponding object. We do this using the intersection overlap heuristic, as above, and assign parts to objects where the intersection ratio exceeds the 0.5 threshold. For some robustness to occlusion, we compute a depth ordering of the polygons in the image (see Section 2.3.3) and assign the part to the polygon with smallest depth that exceeds the intersection ratio threshold. Figure 2-11 gives some quantitative results on the number of parts per object and the probability with which a particular object-part is labeled.

2.3.3 Depth ordering

Frequently, an image will contain many partially overlapping polygons. This situation arises when users complete an occluded boundary or when labeling large regions containing small occluding objects. In these situations we need to know which polygon is on top in order to assign the image pixels to the correct object label. One solution is to request depth ordering information while an object is being labeled. Instead, we wish to reliably infer the relative depth ordering and avoid user input.

The problem of infering depth ordering for overlaping regions is a simpler problem than segmentation. In this case we only need to infer who owns the region of intersection. We summarize a set of simple rules to decide the relative ordering of two overlapping polygons:

- Some objects are always on the bottom layer since they cannot occlude any objects. For instance, objects that do not own any boundaries (e.g. sky) and objects that are on the lowest layer (e.g. sidewalk and road).

- An object that is completely contained in another one is on top. Otherwise, the object would be invisible and, therefore, not labeled. Exceptions to this rule are transparent or wiry objects.

Figure 2-10: Objects and their parts. Using polygon information alone, we automatically discover object-part relationships. We show example parts for the building, person, mountain, sky, and car object classes, arranged as constellations, with the object appearing in the center of its parts. For the car object class, we also show parts when viewpoint is considered.

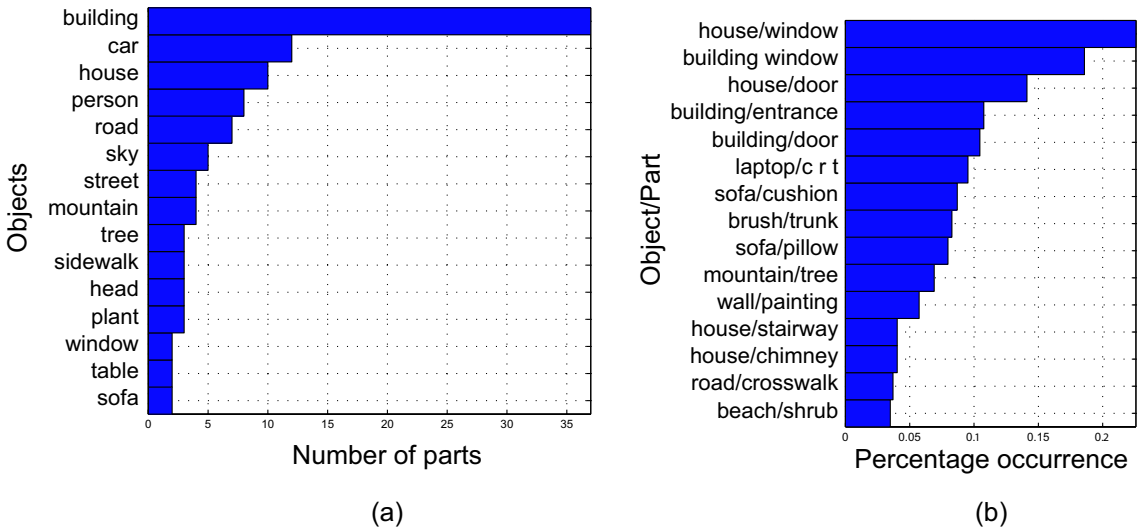


Figure 2-11: Quantitative results showing (a) how many parts an object has and (b) the likelihood that a particular part is labeled when an object is labeled. Note that there are 29 objects with at least one discovered part (only 15 are shown here). We are able to discover a number of objects having parts in the dataset. Also, a part will often be labeled when an object is labeled.

- If two polygons overlap, the polygon that has more control points in the region of intersection is more likely to be on top. To test this rule we hand-labeled 1000 overlapping polygon pairs randomly drawn from the dataset. This rule produced only 25 errors, with 31 polygon pairs having the same number of points within the region of intersection.

- We can also decide who owns the region of intersection by using image features. For instance, we can compute color histograms for each polygon and the region of intersection. Then, we can use histogram intersection [84] to assign the region of intersection to the polygon with the closest color histogram. This strategy achieved 76% correct assignments over the 1000 hand-labeled overlapping polygon pairs. We use this approach only when the previous rule could not be applied (i.e. both polygons have the same number of control points in the region of intersection).

Combining these heuristics resulted in 29 total errors out of the 1000 overlapping polygon pairs. Figure 2-12 shows some examples of overlapping polygons and the

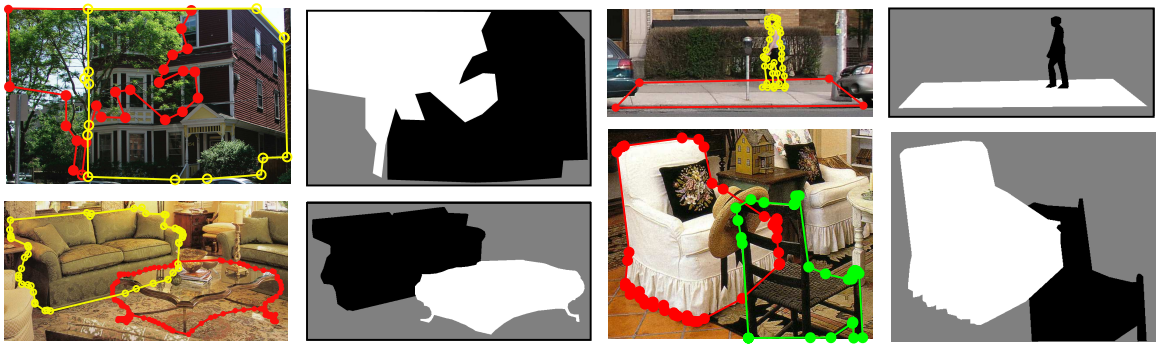


Figure 2-12: Each image pair shows an example of two overlapping polygons and the final depth-ordered segmentation masks. Here, white and black regions indicate near and far layers, respectively. A set of rules (see text) were used to automatically discover the depth ordering of the overlapping polygon pairs. These rules provided correct assignments for 97% of 1000 polygon pairs tested. The bottom right example shows an instance where the heuristic fails. The heuristic sometimes fails for wiry or transparent objects.

final assignments. The example at the bottom right corresponds to an error. In cases in which objects are wiry or transparent, the rule might fail. Figure 2-13 shows the final layers for scenes with multiple overlapping objects.

2.3.4 Semi-automatic labeling

Once there are enough annotations of a particular object class, one could train an algorithm to assist with the labeling. The algorithm would detect and segment additional instances in new images. Now, the user task would be to validate the detection [97]. A successful instance of this idea is the Seville project [3] where an incremental, boosting-based detector was trained. They started by training a coarse detector that was good enough to simplify the collection of additional examples. The user provides feedback to the system by indicating when a bounding box was a correct detection or a false alarm. Then, the detector was trained again with the enlarged dataset. This process was repeated until a satisfactory number of images were labeled.

We can apply a similar procedure to LabelMe to train a coarse detector to be used to label images obtained from online image indexing tools. For instance, if we want more annotated samples of *sailboats*, we can query both LabelMe (18 segmented

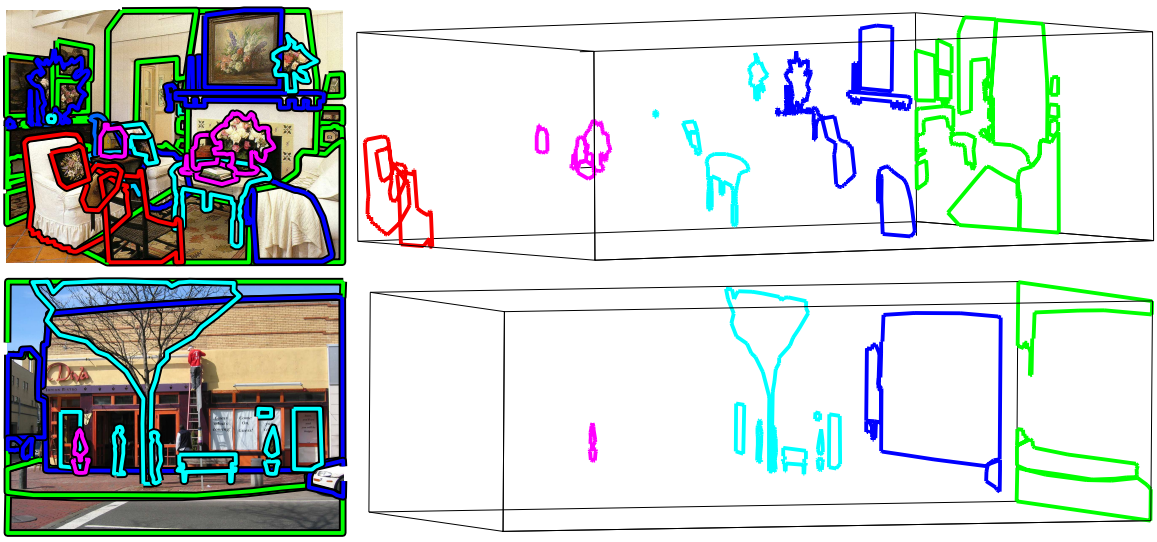
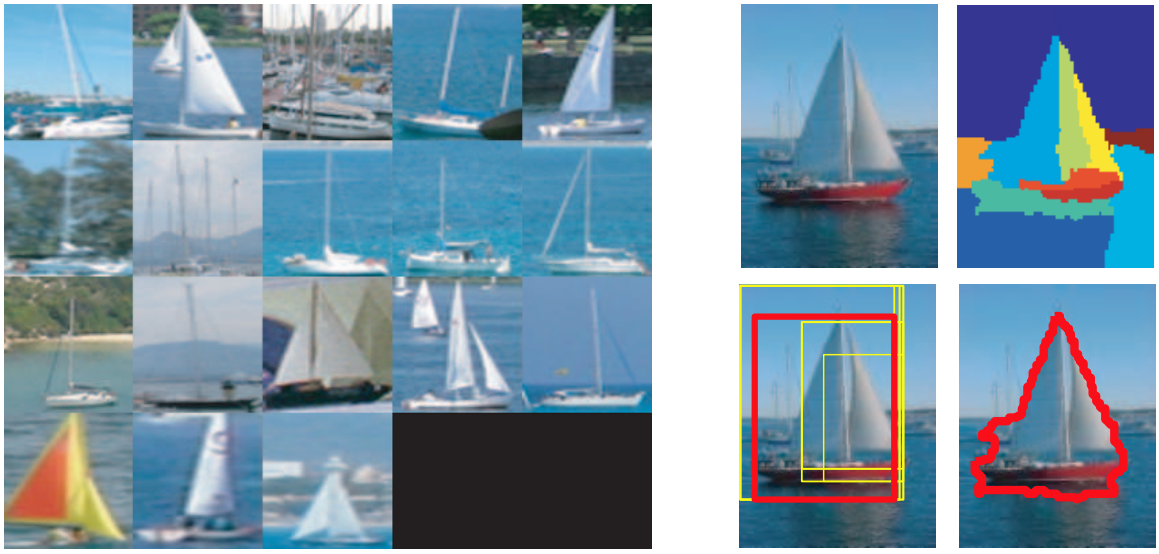


Figure 2-13: Decomposition of a scene into layers given the automatic depth ordering recovery of polygon pairs. Since we only resolve the ambiguity between overlapping polygon pairs, the resulting ordering may not correspond to the real depth ordering of all the objects in the scene.

examples of sailboats were returned) and online image search engines (e.g. Google, Flickr, and Altavista). The online image search engines will return thousands of unlabeled images that are very likely to contain a sailboat as a prominent object. We can use LabelMe to train a detector and then run the detector on the retrieved unlabeled images. The user task will be to select the correct detections in order to expand the amount of labeled data.

Here, we propose a simple object detector. Although objects labeled with bounding boxes have proven to be very useful in computer vision, we would like the output of the automatic object detection procedure to provide polygonal boundaries following the object outline whenever possible.

- Find candidate regions: instead of running the standard sliding window, we propose creating candidate bounding boxes for objects by first segmenting the image to produce 10-20 regions. Bounding boxes are proposed by creating all the bounding boxes that correspond to combinations of these regions. Only the combinations that produce contiguous regions are considered. We also remove all candidate bounding boxes with aspect ratios outside the range defined by



(a) Sailboats from the LabelMe dataset

(b) Detection and segmentation

Figure 2-14: Using LabelMe to automatically detect and segment objects depicted in images returned from a web search. a Sailboats in the LabelMe dataset. These examples are used to train a classifier. b Detection and segmentation of a sailboat in an image downloaded from the web using Google. First, we segment the image (upper left), which produces around 10 segmented regions (upper right). Then we create a list of candidate bounding boxes by combining all of the adjacent regions. Note that we discard bounding boxes whose aspect ratios lie outside the range of the LabelMe sailboat crops. Then we apply a classifier to each bounding box. We depict the bounding boxes with the highest scores (lower left), with the best scoring as a thick bounding box colored in red. The candidate segmentation is the outline of the regions inside the selected bounding box (lower right). After this process, a user may then select the correct detections to augment the dataset

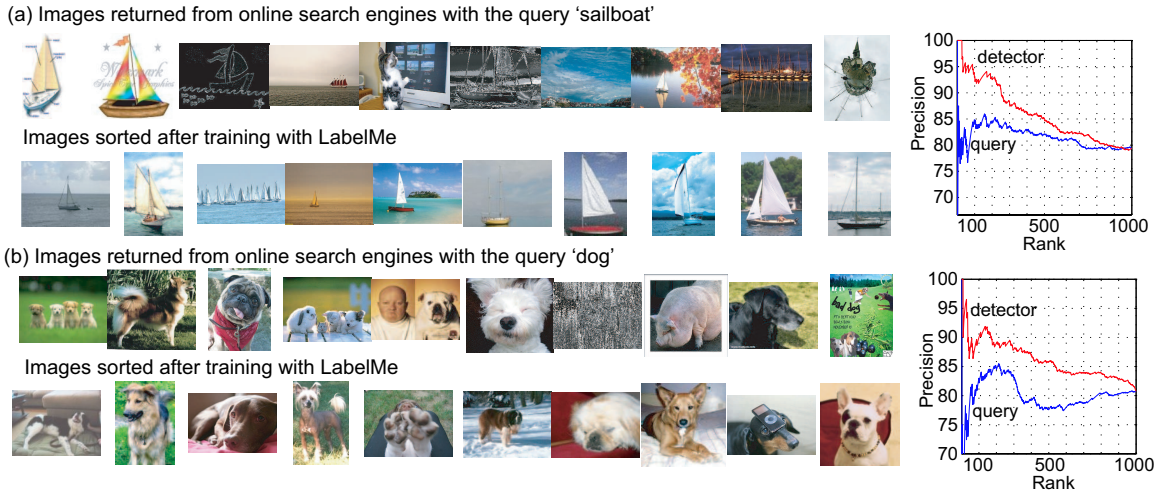


Figure 2-15: Enhancing web-basd image retrieval using labeled image data. Each pair of rows depict sets of sorted images for a desired object category. The first row in the pair is the ordering produced from an online image search using Google, Flickr and Altavista (the results of the three search engines are combined respecting the ranking of each image). The second row shows the images sorted according to the confidence score of the object detector trained with LabelMe. To better show how the performance decreases with rank, each row displays one out of every ten images. Notice that the trained classifier returns better candidate images for the object class. This is quantified in the graphs on the right, which show the precision (percentage correct) as a function of image rank.

the training set. This results in a small set of candidates for each image (around 30 candidates).

- Compute features: resize each candidate region to a normalized size (96 × 96 pixels). Then, represent each candidate region with a set of features (e.g. bag of words [67], edge fragments [64], multiscale-oriented filters [61]). For the experiments presented here, we used the Gist features [61] (code available online) to represent each region.

- Perform classification: train a support vector machine classifier [96] with a Gaussian kernel using the available LabelMe data and apply the classifier to each of the candidate bounding boxes extracted from each image. The output of the classifier will be a score for the bounding boxes. We then choose the bounding box with the maximum score and the segmentation corresponding to the segments that are inside the selected bounding box.

For the experiments presented here, we queried four object categories: sailboats, dogs, bottles, and motorbikes. Using LabelMe, we collected 18 sailboat, 41 dog, 154 bottle, and 49 motorbike images. We used these images to train four classifiers. Then, we downloaded 4000 images for each class from the web using Google, Flickr and Altavista. Not all of the images contained instances of the queried objects. It has been shown that image features can be used to improve the quality of the ranking returned by online queries [32, 11]. We used the detector trained with LabelMe to sort the images returned by the online query tools.

Figure 2-15 shows the results and compares the images sorted according to the ranking given by the output of the online search engines and the ranking provided by the score of the classifier. For each image we have two measures: (i) the rank in which the image was returned and (ii) the score of the classifier corresponding to the maximum score of all the candidate bounding boxes in the image. In order to measure performance, we provided ground truth for the first 1000 images downloaded from the web (for sailboats and dogs). The precision-recall graphs show that the score provided by the classifier provides a better measure of probability of presence

Figure 2-16: Examples of automatically generated segmentations and bounding boxes for sailboats, motorbikes, bottles, and dogs.

of the queried object than the ranking in which the images are returned by the online tools. However, for the automatic labeling application, good quality labeling demands very good performance on the object localization task. For instance, in current object detection evaluations [25], an object is considered correctly detected when the area of overlap between the ground truth bounding box and the detected bounding box is above 50% of the object size. However, this degree of overlap will not be considered satisfactory for labeling. Correct labeling requires above 90% overlap to be satisfactory.

After running the detectors on the 4000 images of each class collected from the web, we were able to select 162 sailboats, 64 dogs, 40 bottles, and 40 motorbikes that produced good annotations. This is shown in Figure 2-16. The user had the choice to validate the segmentation or just the bounding box. The selection process is very efficient. Therefore, semi-automatic labeling may offer an interesting way of efficiently labeling images.

However, there are several drawbacks to this approach. First, we are interested in labeling full scenes with many objects, making the selection process less efficient. Second, in order for detection to work with a reasonable level of accuracy with current methods, the object needs to occupy a large portion of the image or be salient. Third, the annotated objects will be biased toward being easy to segment or detected. Note that despite semi-automatic labeling not being desirable for creating challenging benchmarks for evaluating object recognition algorithms, it can still be useful for training. There are also a number of applications that will benefit from having access

Dataset	# categories	# images	# annotations	Annotation type
LabelMe	183	30369	111490	Polygons
Caltech-101 [28]	101	8765	8765	Polygons
MSRC [102]	23	591	1751	Region masks
CBCL-Streetscenes [13]	9	3547	27666	Polygons
Pascal2006 [25]	10	5304	5455	Bounding boxes

Table 2.3: Summary of datasets used for object detection and recognition research. For the LabelMe dataset, we provide the number of object classes with at least 30 annotated examples. All the other numbers provide the total counts.

to large amounts of labeled data, including image indexing tools (e.g. Flickr) and photorealistic computer graphics [80]. Therefore, creating semi-automatic algorithms to assist image labeling at the object level is an interesting area of application on its own.

2.4 Comparison with existing datasets for object detection and recognition

We compare the LabelMe dataset against four annotated datasets currently used for object detection and recognition: Caltech-101 [28], MSRC [102], CBCL-Streetscenes [13], and PASCAL2006 [25]. Table 2.3 summarizes these datasets. The Caltech-101 and CBCL-streetscenes provide location information for each object via polygonal boundaries. PASCAL2006 provides bounding boxes and MSRC provides segmentation masks.

For the following analysis with the LabelMe dataset, we only include images that have at least one object annotated and object classes with at least 30 annotated examples, resulting in a total of 183 object categories. We have also excluded, for the analysis of the LabelMe dataset, contributed annotations and sequences.

Figure 2-17(a) shows, for each dataset, the number of object categories and, on average, how many objects appear in an image. Notice that currently the LabelMe dataset contains more object categories than the existing datasets. Also, observe that the CBCL-Streetscenes and LabelMe datasets often have multiple annotations per

image, indicating that the images correspond to scenes and contain multiple objects. This is in contrast with the other datasets, which prominently feature a small number of objects per image.

Figure 2-17(b) is a scatter plot where each point corresponds to an object category and shows the number of instances of each category and the average size, relative to the image. Notice that the LabelMe dataset has a large number of points, which are scattered across the entire plot while the other datasets have points clustered in a small region. This indicates the range of the LabelMe dataset: some object categories have a large number of examples (close to 10K examples) and occupy a small percentage of the image size. Contrast this with the other datasets where there are not as many examples per category and the objects tend to occupy a large portion of the image. Figure 2-17(c) shows the number of labeled instances per object category for the five datasets, sorted in decreasing order by the number of labeled instances. Notice that the line corresponding to the LabelMe dataset is higher than the other datasets, indicating the breadth and depth of the dataset.

We also wish to quantify the quality of the polygonal annotations. Figure 2-17(d) shows the number of polygonal annotations as a function of the number of control points. The LabelMe dataset has a wide range of control points and the number of annotations with many control points is large, indicating the quality of the dataset. The PASCAL2006 and MSRC datasets are not included in this analysis since their annotations consist of bounding boxes and region masks, respectively.

2.5 Conclusion

We described a web-based image annotation tool that was used to label the identity of objects and where they occur in images. We collected a large number of high quality annotations, spanning many different object categories, for a large set of images, many of which are high resolution. We presented quantitative results of the dataset contents showing the quality, breadth, and depth of the dataset. We showed how to enhance and improve the quality of the dataset through the application of WordNet, heuristics

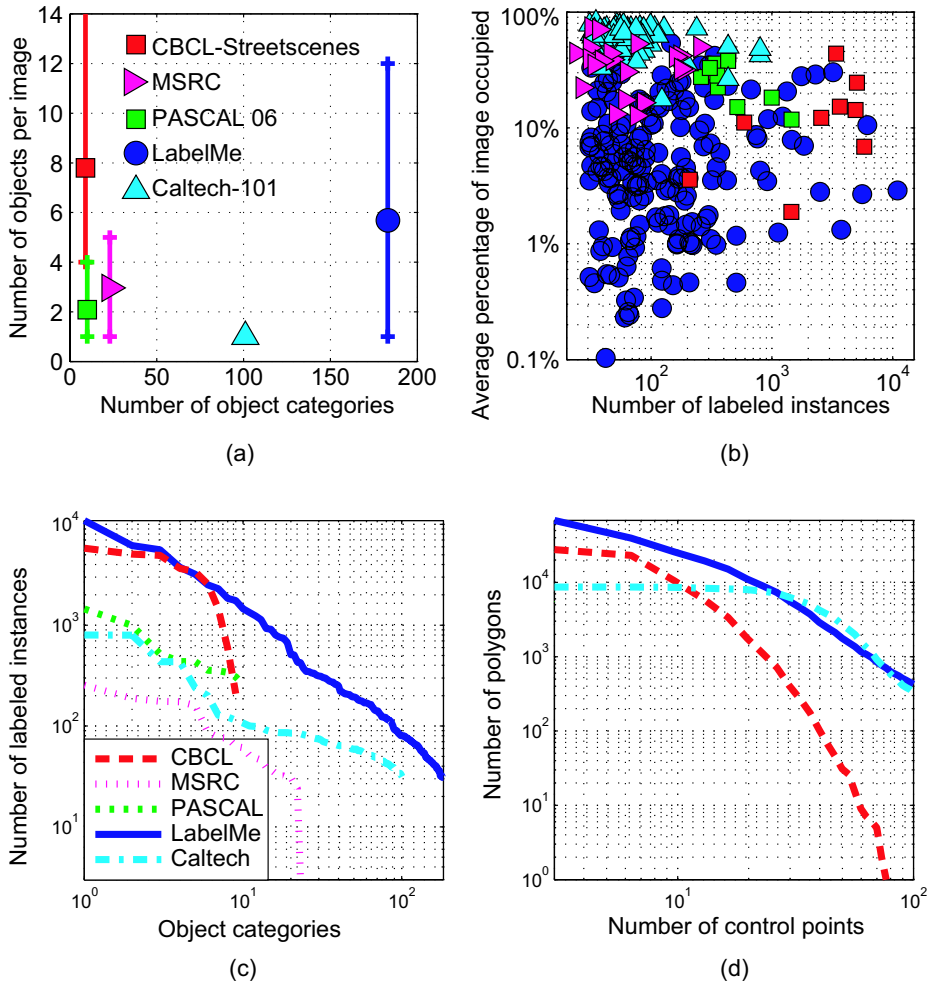


Figure 2-17: Comparison of five datasets used for object detection and recognition: Caltech-101 [26], MSRC [102], CBCL-Streetscenes [13], PASCAL2006 [25], and LabelMe. (a) Number of object categories versus number of annotated objects per image. (b) Scatter plot of number of object category instances versus average annotation size relative to the image size, with each point corresponding to an object category. (c) Number of labeled instances per object category, sorted in decreasing order based on the number of labeled instances. Notice that the LabelMe dataset contains a large number of object categories, often with many instances per category, and has annotations that vary in size and number per image. This is in contrast to datasets prominently featuring one object category per image, making LabelMe a rich dataset and useful for tasks involving scene understanding. (d) Depiction of annotation quality, where the number of polygonal annotations are plotted as a function of the number of control points (we do not show the PASCAL2006 and MSRC datasets since their annotations correspond to bounding boxes and region masks, respectively).

to recover object parts and depth ordering, and training of an object detector using the collected labels to increase the dataset size from images returned by online search engines. We finally compared against other existing state of the art datasets used for object detection and recognition.

Our goal is not to provide a new benchmark for computer vision. The goal of the LabelMe project is to provide a dynamic dataset that will lead to new research in the areas of object recognition and computer graphics, such as object recognition in context and photorealistic rendering.

Chapter 3

Discovering objects and their location in images

3.1 Introduction

Common approaches to object recognition involve some form of supervision, for which the labeled dataset developed in Chapter 2 is very helpful. For example, in face detection, the location or boundary outline is provided [72, 98]. Recent work on multi-class object recognition involves specifying the object identity for cropped images containing a single object [33, 101] or providing keywords that name the objects contained in a given image [7, 6].

While such supervision is valuable, there are two main drawbacks: (i) To effectively learn about all of the intra-class variation that an object class contains and to avoid overfitting, we need a large amount of labelled training data, as seen in Chapter 2. (ii) Providing annotations may introduce unforeseen biases. Furthermore, when we sit down to provide labeled data, we encounter the question, What is an object anyway?

Keeping in mind that the number of available images is several orders of magnitude larger than those with annotations, the above observations lead us to ask the question, *Is it possible to learn visual object classes simply from looking at images?* In an attempt to answer this question, we will apply probabilistic models that have been

used in the natural language processing community to discover interesting topics in a corpus of text documents. To apply these models, we will need a way to convert an image to a text document[1]. We will develop the notion of a *visual word* by quantizing local appearance descriptors [20, 78] that are found at interest points in an image and adjusted for affine invariance [56, 58, 71]. In this chapter, we primarily investigate the Latent Dirichlet Allocation (LDA) model [15] and compare with the very similar probabilistic Latent Semantic Analysis (pLSA) model [39, 40] (see [76, 77] for more details and results for pLSA).

For tractability reasons, both of these models make the simplifying assumption that words in a given document are *exchangable*, that is, the spatial relationship between words are ignored. This is also known as the *bag-of-words* assumption. While this assumption seems to throw away a lot of useful information, the two models cope with the loss of information in the text domain because certain words are highly discriminitive and because of the redundancy of natural language.

For the visual domain, at first glance it seems that we cannot effectively exploit regularity as in the natural language domain since spatial relationships are just as crucial as local features themselves to detecting and recognizing objects. However, the local appearance descriptors that we use have been used for matching regions in two separate images [53] and have been shown to be quite discriminitive and able to encode complex visual stimuli. Furthermore, the affine-invariant regions detected by the interest point operators overlap significantly, as shown in Figure 3-10(b) and hence provide much redundancy. The overlap also preserves spatial information since a reshuffling of the pixels will yield different interest points and descriptors.

In addition to discovering topics, we also show how to localize objects in a given image using the detected visual words. While these visual words provide surprising top-down segementations, we attempt to improve the results by introducing an additional vocabulary, which we call *doublets*, that is formed from spatially neighboring word pairs. We show that doublets provide a somewhat cleaner segmentation of the objects in an image.

[1]This need of *converting* images to text documents will compel us to refer to images as documents.

Several researchers have proposed mining large visual datasets to cluster multiple *instances* of objects. Examples include discovering main characters [34] and other prominent objects and scenes [79] in movies or mining famous people in collections of news photographs [10]. While these are related tasks, to our knowledge, we are among the first to convincingly answer whether it is possible to learn visual object classes simply from looking at images. There has been other recent work that is similar to ours [65, 87, 5].

In Section 3.2, we describe the process of extracting visual words from an image. We also describe the doublets representation. In Section 3.3, we give an overview of various models used to recover the semantics of a corpus of text documents. In particular, we will describe the unigram, mixture of unigrams, pLSA, and LDA models. We will also describe the inference procedure for LDA and compare with the similar pLSA. In Section 3.4, we show outputs after applying the topic discovery models to images. We look at visual words that are highly representative of the discovered topics, cluster images that are all best described by a learned topic, classify unseen images given the learned topics, and show localization of the discovered objects in images. In Section 3.5, we conclude. The results in this chapter were developed in collaboration with Josef Sivic, Alyosha Efros, Andrew Zisserman, and William Freeman and appeared at the 2005 International Conference on Computer Vision [77] and also as an MIT AI Lab technical report [76].

3.2 Obtaining Visual Words

In this section, we will outline how to obtain visual words so that an image can be converted to a document. We will describe two types of visual words: vector-quantized SIFT features [53] obtained through two different interest point detectors and pairs of words called *doublets* that loosely encode spatial relationships. With these visual words, we can then apply the topic discovery models to the data.

3.2.1 Vector-Quantized SIFT Features

We seek a vocabulary of visual words that will be insensitive to changes in viewpoint and illumination. To achieve this, we use vector quantized SIFT descriptors [53] computed on affine covariant regions [56, 58, 71]. Affine covariance gives tolerance to viewpoint changes. SIFT descriptors, which histograms the response of local oriented filters applied at various scales over a nonoverlapping spatial grid, gives some tolerance to illumination changes. Vector quantizing these descriptors gives tolerance to morphology within an object category. Others have used similar descriptors for object classification [20, 62], but in a supervised setting.

Two types of affine covariant regions are computed for each image. The first is constructed by elliptical shape adaptation about an interest point. The method is described in [58, 71]. The second is constructed using the maximally stable procedure of Matas *et al.* [56] where areas are selected from an intensity watershed image segmentation. For both of these we use the binaries provided at [2]. Both types of regions are represented by ellipses. These are computed at twice the originally detected region size in order for the image appearance to be more discriminating.

Each ellipse is mapped to a circle by appropriate scaling along its principal axes and a SIFT descriptor is computed. There is no rotation of the patch, i.e. the descriptors are rotation variant (alternatively, the SIFT descriptor could be computed relative to the the dominant gradient orientation within a patch, making the descriptor rotation invariant [53]; since this invariance gives slightly worse performance, we do not carry out this procedure in this work). The SIFT descriptors are then vector quantized into *visual words* for the vocabulary. The vector quantization is carried out by k-means clustering, which finds around 300K regions. The training regions are extracted from a random subset (about one third of each category) of images of airplanes, cars, faces, motorbikes and backgrounds (see Expt. (2) in section 3.4). About 1K clusters are used for each of the Shape Adapted and Maximally Stable regions, and the resulting total vocabulary has 2,237 words. The number of clusters, k, is clearly an important parameter. The intention is to choose the size of k to determine

words which give some intra-class generalization. This vocabulary is used for all the experiments throughout this thesis.

3.2.2 Doublet Visual Words

If we histogram visual words, we lose the spatial relationship between different regions of the image. We seek to increase the spatial specificity of object description while at the same time allowing for configurational changes. We thus augment our vocabulary of words with *doublets* – pairs of visual words that co-occur within a local spatial neighborhood (c.f. we call visual words *singlets*). In this new representation, for a vocabulary of size V, the codebook has on the order of VxV entries (we remove permutations). For ease the computational burden, we only form doublets for 100 singlet codebook entries. We choose the 100 singlet words that best represent a discovered topic, which will be described in Section 3.3.2.

We find doublets in images by considering each visual word. If a given visual word belongs to the top set of 100 singlets, we look for another visual word to form a pair. To avoid trivial doublets (those with both visual words in the same location), we discard those visual words with ellipses having significant overlap. We then form doublets from all pairs of the remaining words that are within five nearest neighbors of each other.

We prefer pairs of visual words with ellipses of similar sizes. We find these by multiplying the distance between the two visual word centroids by the ratio of the larger to smaller major ellipse axis of the two visual word ellipses. Figure 3-1 illustrates the geometry and formation of the doublets. Figures 3-10(d),(e) show examples of doublets on a real image.

3.3 Models for Discovering Latent Topics

We have shown how to convert images to look like text documents through the use of the visual words representation. We now wish to analyze the images and discover object categories. In this section, we describe models that have been used in the text

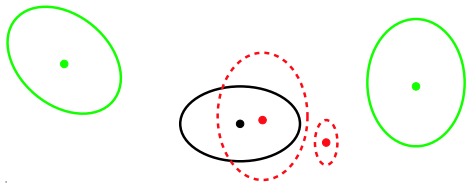


Figure 3-1: The doublet formation process. We augment the visual word representation by counting the co-occurrence of pairs of codebook entries. We find doublet pairs by considering each visual word in an image. For a target visual word (depicted in the center in black), we consider nearby visual words. We discard those visual words that significantly overlap with the target visual word. Moreover, we give preference to visual words that are the same size as the target visual word. In this case, the two dotted red ellipses are discarded and the green ones (of the same size) are paired.

understanding community to discover latent topics.

3.3.1 Unigram and Mixture of Unigrams Models

Histogramming the words that appear in a document provides a summary of its contents. We can also summarize an entire corpus of documents in this manner. This is known as a *unigram* model. By histogramming an entire corpus of N documents, each containing M_i words, the unigram model recovers the parameters of the probability distribution $p(w_{ij}|\phi)$ over the words w_{ij} in a document. If the histogram is normalized, then the distribution is assumed to be a multinomial. This model is depicted in Figure 3-2(a) as a graphical model [46]. We use plate notation, where nodes inside a plate are duplicated by the counts in the top corner of the plate.

While the unigram model provides a summary of the corpus' contents, its descriptive ability is quite limited. The reason is that a corpus will contain documents covering a range of topics. Arranging words into documents provides some structure to separate the different topics. This is because an author of a document will not write about all possible topics, but only a few. By histogramming all of the words in a document corpus, the words corresponding to the latent topics are all jumbled together. We therefore lose the ability to the words that best describe the topics.

We can augment the unigram model by assuming that a document's author writes about only one topic. This model is called the *mixture of unigrams* model and is

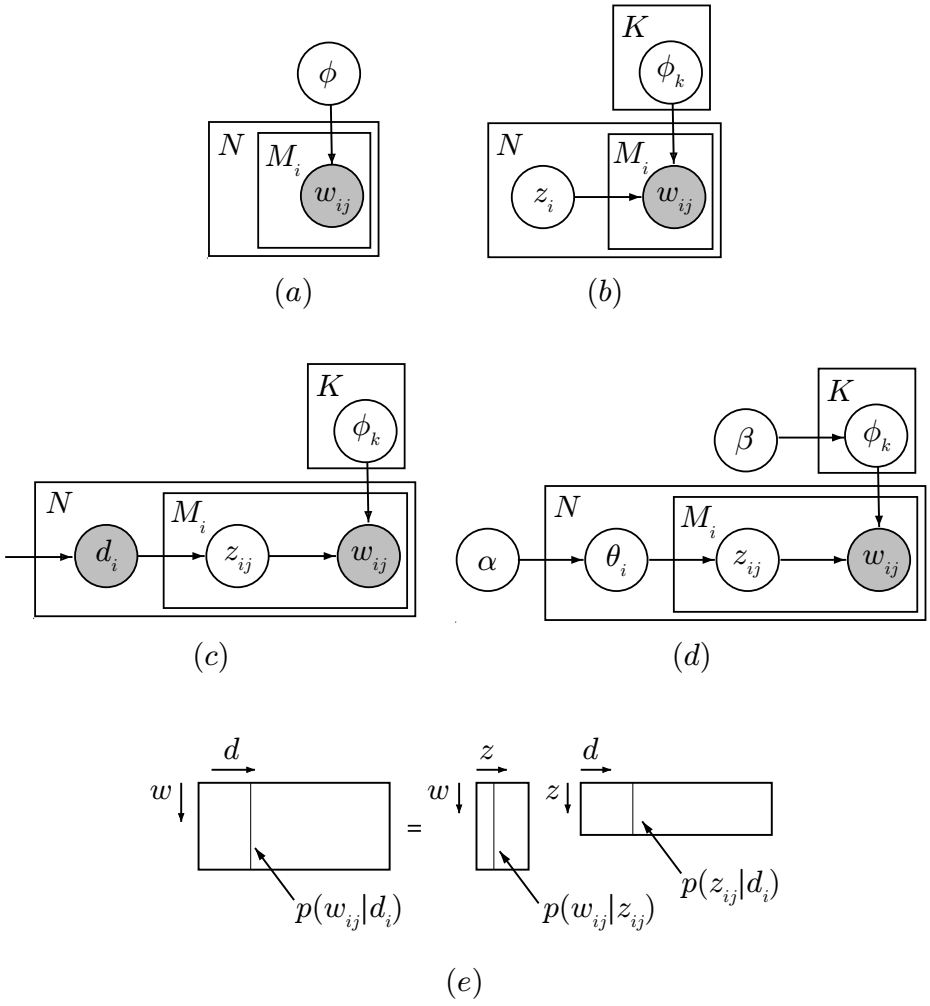


Figure 3-2: (a) Unigram model. There is only one topic for the entire corpus, which is described by a single set of mixing weights over the vocabulary. (b) Mixture of unigrams model. This model assumes that there are multiple global topics, each with their own mixing weights over the vocabulary, and that each document can be described by a single topic. The words are distributed conditioned on the topic. However, this proves to be too limiting for describing a corpus of documents. (c) pLSA model of Hofmann [39]. In this model, each document may be described by multiple topics, with each document having separate mixing weights for the different topics. (d) LDA model of [14]. This model is a generative model and incorporates the same augmented features for text as pLSA. However, this model has several nice advantages over pLSA, which we describe in the text. Both pLSA and LDA can be viewed as performing a low-rank matrix factorization over the latent topics, which we depict in (e). More specifically, words in a document are histogrammed as a column vector ($p(w_{ij}|d_i)$), with the documents in the corpus arranged as a matrix (shown on the left). pLSA and LDA can both recover two matrices: the set of mixing weights over the vocabulary for the topics ($p(w_{ij}|z_{ij})$) and the set of mixing weights over the topics for each document ($p(z_{ij}|d_i)$).

depicted in Figure 3-2(b). We assume that there are K topics, each having its own mixing weights for the words that comprise the topic. Each document is assigned to one of the latent topics, with its topic assignment indicated by z_i. Learning and inference for this model is straightforward as it is similar to a mixture model.

3.3.2 The pLSA and LDA Models

While the mixture of unigrams model proves successful for documents comprising a single topic, it runs into problems on documents containing multiple topics. The reason for this is similar to why a unigram model over a corpus fails to learn about topics: since words in a document are constrained to belong to a single topic, the words belonging to different topics are jumbled together. Since multiple-topic documents are common and arise naturally (such as this thesis), this problem must be addressed.

We augment the mixture of unigrams model to allow each word in a document to belong to a different topic. In this way, we allow the documents to be composed of multiple topics. We consider two models that achieve this: probabilistic Latent Semantic Analysis (pLSA) [39] and Latent Dirichlet Allocation (LDA) [14].

The pLSA model is shown in Figure 3-2(c). The pLSA model recovers two probability distributions: $p(z_{ij}|d_i)$ for document d_i and $p(w_{ij}|z_{ij}, \phi)$. Both of these distributions are assumed to be multinomial. The first distribution allows separate topic mixing weights for each document, while the second distribution is shared across all documents.

The LDA model is depicted in Figure 3-2(d). The LDA model is a proper generative model, where:

$$\theta_i|\alpha \sim Dirichlet(\alpha) \tag{3.1}$$

$$\phi_k|\beta \sim Dirichlet(\beta) \tag{3.2}$$

$$z_{ij}|\theta_i \sim \theta_i \tag{3.3}$$

$$w_{ij}|z_{ij} = k, \phi \sim \phi_k \tag{3.4}$$

As with pLSA, LDA assumes that each document has its own distribution over topics $p(z_{ij}|\theta_i)$ and topic distributions $p(w_{ij}|z_{ij}, \phi)$ shared across all documents.

The LDA generative model also results from application of De Finetti's theorem [21]. Since documents are represented by histograms of words, we toss out the ordering of the words. Since a random permutation of the words in a document results in the same histogram, the words in the document are said to be *exchangeable*. This is also known as the *bag of words* assumption. Moreoever, an infinite sequence of random variables (in this case words) is said to be *infinitely exchangeable* if the joint distribution of all finite subsequences are exchangeable. De Finetti's theorem states that an infinitely exchangeable infinite sequence of random variables is independently and identically distributed conditioned on a random parameter, which in turn is drawn from another distribution. In the finite case, this means:

$$p(w_i, z_i|\alpha) = \int p(\theta_i|\alpha) \prod_{j=1}^{M_i} p(z_{ij}|\theta_i)\ p(w_{ij}|z_{ij})\ d\theta_i \tag{3.5}$$

Notice that this is proportional to the generative model in Figure 3-2(d).

Notice that the data likelihood term for pLSA is:

$$p(w_{ij}|d_i) \propto \sum_{k=1}^{K} p(w_{ij}|z_{ij} = k)\ p(z_{ij} = k|d_i) \tag{3.6}$$

and for LDA:

$$p(w_{ij}) \propto \sum_{k=1}^{K} p(w_{ij}|z_{ij} = k, \phi_k)\ p(z_{ij} = k|\theta_i) \tag{3.7}$$

Both models sum over the latent topic assignments z_{ij}. Hence, we can view pLSA and LDA as performing a constrained low-rank matrix factorization, as depicted in Figure 3-2(e). The document histograms are stacked as columns in a matrix, as depicted on the left. pLSA and LDA recover matrices corresponding to mixing weights over the vocabulary words for the topics ($p(w_{ij}|z_{ij})$) and mixing weights over the topics for each document ($p(z_{ij}|d_i)$. This view of the models relates them to low rank factorization methods for recovering the semantic structure of text, as pioneered by Latent Semantic Analysis (LSA) [22].

While the pLSA and LDA are powerful models for text, one limitation is the requirement to choose the number of topics K to discover. While this is usually not a problem, it does require a priori knowledge of the data. However, a Bayesian extension of the LDA model, the Hierarchical Dirichlet Process [85], or minimum complexity methods [8] can be used to automatically infer the number of topics implied by a corpus. We do not investigate this problem for the remainder of this thesis.

3.3.3 Inference for pLSA and LDA

We are interested in recovering the mixing weight over words for the latent topics that appear in the corpus. For pLSA, this means that we wish to recover both $p(w_{ij}|z_{ij})$ and $p(z_{ij}|d_i)$ that maximizes $p(w|d)$. We can do this via expectation maximization (EM), as outlined in [39]. In our implementation, we observed convergence in 40–100 iterations, with each iteration taking approximately 2.3 seconds. This was tested on 4K images with 7 fitted topics and $\sim$300 non-zero words counts per image using a Matlab implementation on a 2GHz PC. For more details, see [77].

For LDA, we marginalize over the topic distributions for each document. Bayesian inference can be achieved by variational expectation maximization [14], expectation propagation [59], or Gibbs sampling [36]. For this work, we utilize the Gibbs sampler, which we now describe.

We wish to find the set of mixing weights over words that maximizes the following posterior distribution:

$$\phi^* \in \underset{\phi}{\text{argmax}}\ p(\phi|w,\alpha,\beta) = \underset{\phi}{\text{argmax}} \sum_z p(\phi|w,z,\alpha,\beta)\ p(z|w,\alpha,\beta) \quad (3.8)$$

$$\approx \underset{\phi}{\text{argmax}} \sum_{s=1}^{S} p(\phi|w,z^s,\alpha,\beta) \quad (3.9)$$

$$\approx \underset{\phi}{\text{argmax}} \max_{s=\{1,\ldots,S} p(\phi|w,z^s,\alpha,\beta) \quad (3.10)$$

where $z^s \sim p(z|w,\alpha,\beta)$ are samples drawn from a posterior distribution over z. The samples z are drawn in succession for each word, conditioned on topic assignments for the remaining words:

$$z_{ij} \sim p(z_{ij} = k|w_{ij} = v, w_{\setminus(ij)}, z_{\setminus(ij)}, \alpha, \beta) \quad (3.11)$$

$$\propto \left(N_{ik}^{\setminus(ij)} + \alpha_k\right)\left(N_{kv}^{\setminus(ij)} + \beta_v\right) \quad (3.12)$$

where $\setminus(ij)$ indicates that we exclude example (ij) from consideration, $N_{ik}^{\setminus(ij)}$ is the number of words assigned to topic k in document i (excluding the topic assignment of z_{ij}), and $N_{kv}^{\setminus(ij)}$ is the number times vocabulary word v is assigned to topic k. Notice that computationally, we simply need to keep track of these two counts and update them for each new sample.

We recover the parameters ϕ as:

$$\phi_k \propto N_{kv} + \beta_v \quad (3.13)$$

where N_{kv} is the total number of times vocabulary word v is assigned to topic k.

To infer the latent mixing weight over topics for a given document:

$$\theta_i \propto N_{ik} + \alpha_k \quad (3.14)$$

where N_{ik} is the number of time topic k was assigned in document i.

Finally, we can compute the posterior probability of a topic assignment given the

learned parameters ϕ and inferred weights θ_i:

$$p(z_{ij} = k | w_{ij} = v, \theta_i, \phi_k) \propto \theta_{ik} \ \phi_{kv} \tag{3.15}$$

This distribution will be handy later when we attempt to localize a discovered object category in an image.

For our implementation, we set the hyperparameters to be symmetric as in [36], where $\alpha_k = 0.5$ and $\beta_v = 0.5$ for all topics k and vocabulary words v. The hyperparamters essentially perform Laplacian smoothing by controlling the mixing of the multinomial weights (lower values give less mixing). This prevents degeneracy for the case where there are no observed words for a given topic. We run the Gibbs sampler 10 times with 100 iterations (every z_{ij} is sampled in each iteration) for each run. For each run, we randomly assign the initial topic settings. We compute the latent parameters based on the sample z_s that gives the highest posterior probability $p(z|w, \alpha, \beta)$ across the 10 runs. Empirically, the sampler reaches a plateau of the posterior after approximately 50 iterations, with each iteration taking a few seconds on a 2GHz PC running a Matlab implementation.

3.3.4 Comparison of pLSA and LDA

While both pLSA and LDA allow for topic assignments to individual words, there are nonetheless noteworthy differences between the two models. First, the pLSA model indexes the documents in the *training* set. This causes pLSA to not be a generative model and has the undesirable effect of not being able to directly model unseen documents. The LDA model does not suffer from this. To ameliorate this, a folding-in heuristic must be used where the learned weights over words are held fixed and the topic weights for the new documents are inferred. Second, while the LDA model marginalizes over the mixing weights for topics in documents, pLSA requires direct computation of the mixing weights for both the topics and words. This causes overfitting when the two models are applied to unseen documents, as observed in [14, 59]. pLSA uses a heuristics, such as tempering, to smooth the learned parameters.

See [14] for a more detailed analysis of these effects.

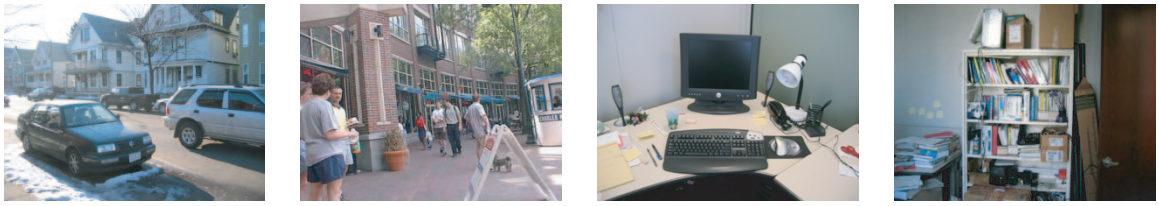
3.4 Experiments

In this section, we analyze the outputs of the LDA topic discovery model when applied to the visual words representation of images. For our analysis, we explore four interesting tasks: (i) understanding the outputs of the LDA model by looking at the visual words that best describe the discovered visual topics, (ii) clustering the input images based on the likelihood of their membership to the topics, (iii) classification of unseen images that are outside of the input image set using the visual topics, and (iv) localization of the discovered visual topics within images.

We will see that the visual topics will often correspond to object categories or types of texture. To ensure that we understand the topics that are discovered, we consider image sets for which we know the desired visual topics. We consider the Caltech 101 dataset [27, 33] and the MIT Objects and Scenes dataset [91]. The Caltech dataset contains 101 different object categories with one category prominently featured in a given image. This dataset contains large intra-category variation. The MIT dataset is more difficult as it contains multiple object categories in a given image. Example images from both of these datasets and a description of the preprocessing on these images are shown in Figure 3-3. We will show results on these two datasets and compare with the output from pLSA and a baseline k-means method.

Baseline Method – K-means (KM) To understand the contributions of the topic discovery model to the system performance, we also implemented an algorithm that finds clusters based on the word frequency vector for each image (i.e. we compute a vector w_i of length V that is the normalized histogram count of the visual words for a given image). This is very similar to the mixture of unigrams model (c.f. Section 3.3.1) except that we assume $p(w_i|z_i)$ is Gaussian distributed where z_i is the topic assignment for the document. The standard k-means procedure is used to determine K clusters from the word frequency vectors by hard-assigning each document to exactly one

Examples from Caltech 101



Examples from MIT Objects and Scenes

Figure 3-3: We pool images from five object categories from the Caltech 101 dataset (top row). The categories and number of images used are: faces, 435; motorbikes, 800; airplanes, 800; cars rear, 1155; background (indoor and outdoor scenes around the Caltech campus), 900. We chose to experiment with these categories since they offer the greatest number of examples per category, thereby increasing the chance of success. All images have been converted to grayscale before processing. No other alterations were made with the exception of removing a white border around a number of motorbike images since this was providing an artifactual cue. The MIT Objects and Scenes dataset (bottom row) contains 2,873 images and indoor and outdoor scenes, with annotations consisting of polygonal outlines. Again, these images were converted to grayscale before processing.

cluster based on the Euclidean distance of the feature vector to the cluster center.

3.4.1 Topic Discovery

We carry out two experiments of increasing complexity and recover the visual words that correspond to the discovered topics. For each experiment, we specify the number of topics to discover and then fit an LDA model to the data. To visualize the visual words that correspond to each discovered topic, we use the recovered ϕ multinomial parameters, as given in Equation 3.13, to display those words with high probability for a given topic.

Expt. (1) Images of four object categories with cluttered backgrounds. We use images from the faces, motorbikes, airplanes, and cars rear object categories

from the Caltech 101 dataset. All four of these categories have cluttered backgrounds and vary significantly in scale (especially in the case of car rear). Figure 3-4 shows the two top visual words for each of the discovered topics when $K = 4$. Notice that the top words for each discovered topic correspond to semantically meaningful regions for the four object categories (e.g. eyes for faces, wheel parts for motorbikes, nose tips for airplanes, and license plates for cars rear). It appears that we are discovering the four object categories!

Expt. (2) Images of four object categories plus "background" category. We add to Expt. (1) the set of background images from the Caltech 101 dataset. We increase the number of topics to account for the newly added background images. We visualize the background image set by fitting an LDA model with $K = 3$ to the set of background images. We show the most likely visual words (based on ϕ) for each discovered topic in Figure 3-5. These visual words illustrate the type of scenes contained in the dataset.

3.4.2 Clustering Images

Since it appears that we are discovering topics corresponding to object categories, we would like to assign each image to one of the discovered topics. We do this by using the recovered θ multinomial parameters, as given in Equation 3.14, to assign each document to the topic with the highest probability. To evaluate the goodness of these assignments, we compute a confusion matrix, where each column corresponds to an object category and whose entries indicate the distribution of the images to the different topics. For example, if there are C object categories in the image set and K topics are discovered, then the resulting confusion matrix will be of dimension KxC with columns summing to one. An ideal confusion matrix, up to some permutation of the rows, is the identity matrix. For a given confusion matrix, we show an intensity plot of the matrix for the best permutation of the rows and report the average along the diagonal and the number of images that appear off of the diagonal.

Figure 3-6 shows confusion matrices and Table 3.1 shows summary statistics for

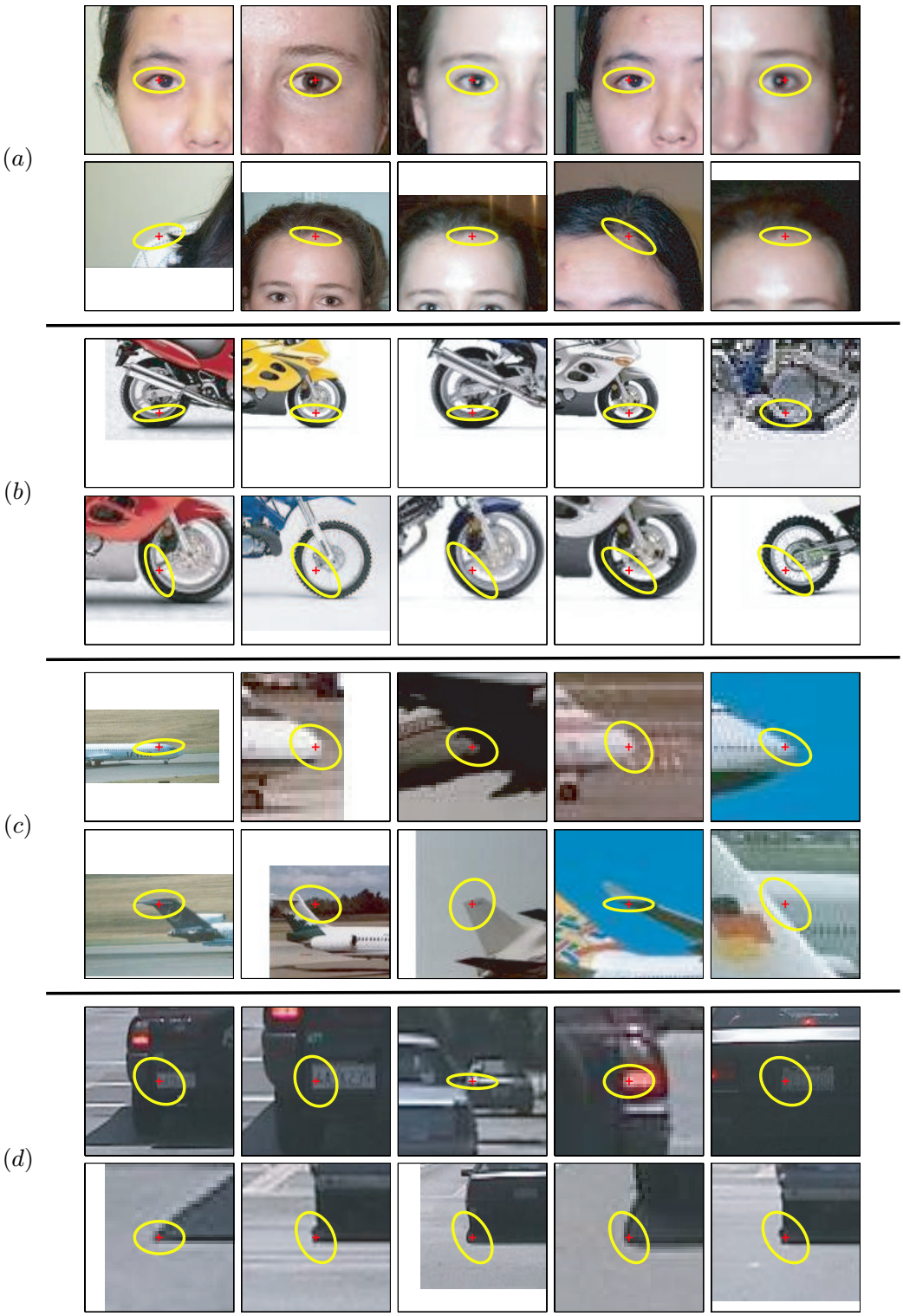


Figure 3-4: The two most likely words (shown by 5 examples in a row) for four learned topics in Expt. (1): (a) Faces, (b) Motorbikes, (c) Airplanes, (d) Cars.

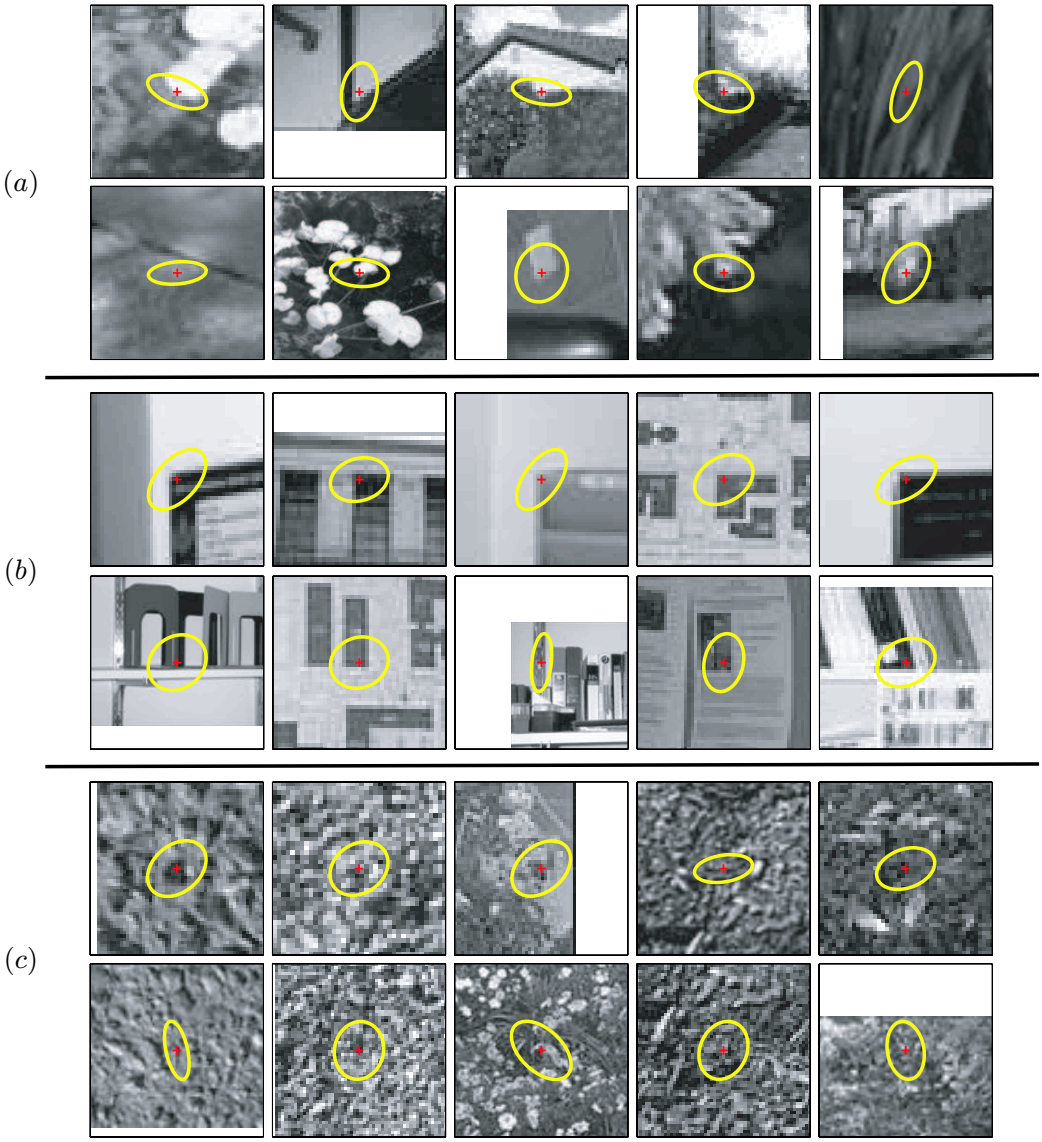


Figure 3-5: The most likely words (shown by 5 examples in a row) for the three background topics learned in Expt. (2): (a) mainly local feature-like structure (b) mainly corners and edges coming from the office/building scenes, (c) mainly textured regions like grass and trees.

Expt. (1) as the number of topics K is increased from 4 to 7. For $K = 4$, we get an almost perfect assignment score for the four categories. As the number of topics is increased, certain object categories seem to cluster into subtopics. This seems to happen mostly with the motorbike and car rear object categories. Upon looking at the clustered images, we find that these subtopics correspond to motorbikes containing background versus no background and the car rear set containing multiple examples of two cars from a video sequence. These discovered subtopics agree well with the structure of the data.

Figure 3-7 and Table 3.1 show confusion matrices and summary statistics for Expt. (2) as K is increased from 5 to 7. Notice that for both this and the earlier experiments, both LDA and pLSA models significantly outperform the baseline k-means method and that there is little difference in the LDA and pLSA performance. For LDA, there is some confusion between the background and the desired object categories. For example, for $K = 5$ a significant number of background images are incorrectly assigned as faces. Another example is for $K = 7$ where some motorbikes and cars are incorrectly assigned to background topics. This seems to occur because, as noted above, many images contain a mixture of visual words corresponding to foreground and background. Examples include faces occurring in office scenes and cars appearing in road scenes.

To remedy this situation, we provide a small amount of supervision by learning an LDA model for half of the background images (450 images in total) with $K = 3$ and then, while holding the recovered ϕ parameters fixed, learn another LDA model with $K = 7$ on the remaining images. This has the effect of decorrelating the foreground and background visual words since the background category will be described by the first set of parameters and the remaining parameters will explain the four object categories.

Figure 3-7 and Table 3.1 show the improvement in the confusion matrix and summary statistics. Again, there is little difference in performance between this semi-supervised version of LDA and an equivalent version of pLSA (described in [77]).

Expt.	Categories	T	LDA		pLSA		KM baseline	
			%	#	%	#	%	#
(1)	4	4	97	86	98	70	72	908
(2)	4 + bg	5	78	931	78	931	56	1820
(2)*	4 + bg	6	84	656	76	1072	–	–
(2)*	4 + bg	7	78	1007	83	768	–	–
(2)*	4 + bg-fxd	7	90	330	93	238	–	–

Table 3.1: Confusion matrix summary statistics of the experiments comparing LDA, pLSA, and the k-means baseline method. The '%' column is the average along the diagonal of the confusion matrix corresponding to a given experiment and the '#' column is the number of misclassified images. For the case of (2)*, the two/three background topics are allocated to one category. See the text and Figures 3-6 and 3-7 for more details. Notice that both LDA and pLSA have comparable results and outperforms the baseline k-means method.

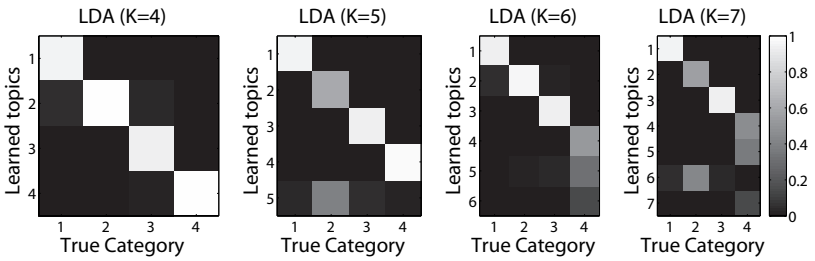


Figure 3-6: LDA confusion tables for Expt. (1) and increasing number of topics discovered (K=4,5,6,7). Brightness indicates number, with the ideal being bright down the diagonal. The rows/columns correspond from top/left to bottom/right as faces, motorbikes, airplanes, and cars rear. Notice that we learn interesting "subtopics" in the car and motorbike categories. For the motorbike case, these subtopics correspond to motorbikes with or without background. For the cars, there appears to be a couple of particular cars with many repeated examples in the dataset, with topics assigned to these cases.

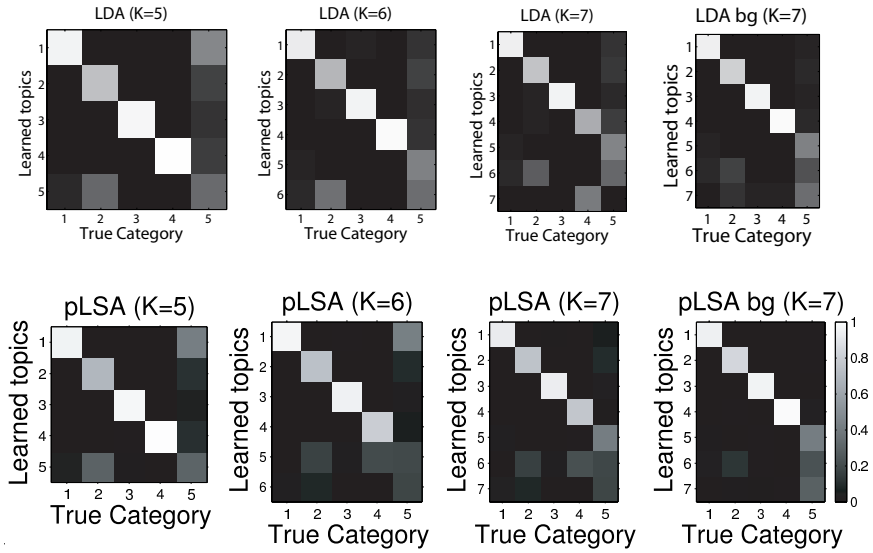


Figure 3-7: Confusion tables for Expt. (2) for LDA (top row) and for pLSA (bottom row). The number of discovered topics is increased from $K = 5$ to $K = 7$, with the last column depicting $K = 7$ and fixed background parameters. Brightness indicates number, with the ideal being bright down the diagonal. The rows/columns correspond from top/left to bottom/right as faces, motorbikes, airplanes, cars rear, and the set of background images. Notice how we remove some of the confusion between the background images and that we discover background "subtopics" when we increase the K.

	LDA			
True Class →	Faces	Motorbikes	Airplanes	Cars rear
Topic 1 - Faces	97.70	0.50	3.75	1.00
Topic 2 - Motorbikes	1.38	96.50	2.25	0.00
Topic 3 - Airplanes	0.00	0.75	93.00	0.00
Topic 4 - Cars rear	0.92	2.25	1.00	99.00

Table 3.2: Confusion matrices for unseen test images in Expt. (3) for LDA. The test images comprise examples from four object categories (there are no background images). Notice that there is little confusion between the different categories.

	pLSA			
True Class →	Faces	Motorbikes	Airplanes	Cars rear
Topic 1 - Faces	99.54	0.25	1.75	0.75
Topic 2 - Motorbikes	0.00	96.50	0.25	0.00
Topic 3 - Airplanes	0.00	1.50	97.50	0.00
Topic 4 - Cars rear	0.46	1.75	0.50	99.25

Table 3.3: Confusion matrices for unseen test images in Expt. (3) for pLSA. The test images comprise examples from four object categories (there are no background images). Notice that there is little confusion between the different categories.

3.4.3 Classifying New Images

After fitting an LDA model to a corpus of images, we can use the learned ϕ parameters to classify unseen images. We describe two experiments to investigate this.

Expt. (3) Training images of four object categories plus "background" category. We compare our performance with the supervised method of Fergus *et al.* [33]. We train an LDA model with $K = 7$ on half of the images of the four object categories plus all of the background images. We then test on the remaining images belonging to the object categories and assign them to the learned object topics (we ignore the background topics when performing the assignments). The resulting confusion matrix is shown in Tables 3.2 and 3.3. Notice that there is little confusion for LDA and pLSA.

Expt. (4) Binary classification of category against background. We investigate binary classification of unseen images containing objects versus unseen back-

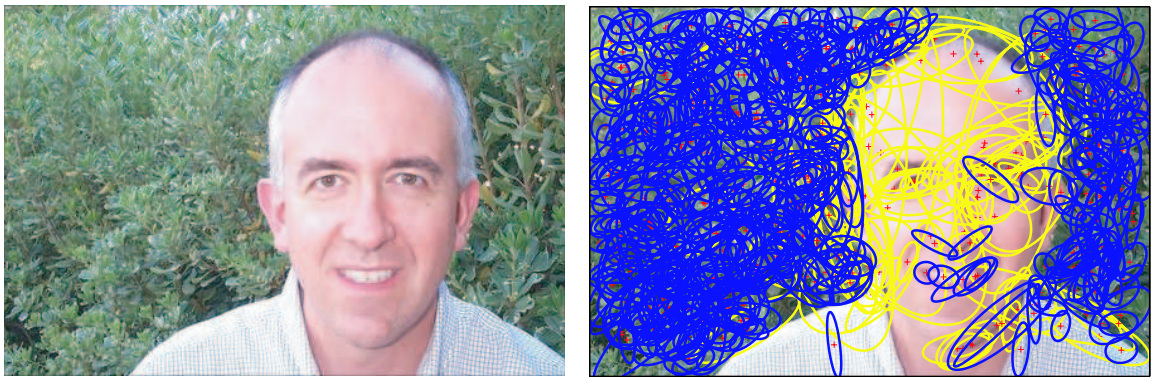
Object categ.	LDA (a)	LDA (b)	pLSA (a)	pLSA (b)	Fergus *et al.* [33]
Faces	17.3	7.8	5.3	3.3	3.6
Motorbikes	17.5	9.9	15.4	8.0	6.7
Airplanes	4.8	2.5	3.4	1.6	7.0
Cars rear*	21.3 / 11.6	18.2 / 8.5	21.4 / 11.9	16.7 / 7.0	9.7

Table 3.4: Equal error rates for the classification task of object versus background for LDA, pLSA, and [33]. The object category test images were classified against (a) the set of 500 testing background images (this test was performed in [33]) and (b) the set of testing background images and testing images of all other object categories. The small confusion between the different categories explains the improvement in performance in column (b). (*) For the cars rear category, we classify against a separate set of background images consisting of road scenes (as in [33]). For training, we experimented with using 400/900 background images respectively.

ground images and compare with [33]. For training, we use the same set of images containing objects as in Expt. (3) but use only 400 (out of 900) background images. We then fit an LDA model with $K = 7$ on this set of images. The test set (the remaining images containing objects and the 500 background images) is used to produce ROC curves for each discovered object topic. We report the equal error rates (the point where the probability of detecting the object category equals one minus the false alarm rate) for LDA, pLSA, and [33] in Table 3.4. For the cars rear category, to be comparable with [33], for testing we classify against a separate set of road scenes. Also, for the cars rear category, we experimented with training on all of the background images. Notice that LDA and pLSA obtains comparable performance with [33] even though we do not supply any training labels for the images.

3.4.4 Obtaining a Weak Segmentation

We now consider scoring each visual word that appears in a particular image. We can do this with the posterior probability given in Equation 3.15. Figure 3-8 shows two images whose visual words have probability greater than 0.88 of belonging to a particular topic (we used the model trained in Expt. (2) with $K = 7$ and the parameters of 3 background topics pre-learned and held fixed). For both examples, it appears that the topic labels assigned to the visual words align nicely with the



(a) (b)

(c) (d)

Topic	Figure 3-8(a),(b) $P(z\|\theta)$	# regions	Figure 3-8(c),(d) $P(z\|\theta)$	# regions
1 Faces (yellow)	0.25	81	0.00	0
2 Motorbikes (green)	0.02	0	0.00	0
3 Airplanes (black)	0.00	0	0.01	0
4 Cars rear (red)	0.00	0	0.66	84
5 Backg I (magenta)	0.02	0	0.32	14
6 Backg II (cyan)	0.00	0	0.01	0
7 Backg III (blue)	0.70	522	0.00	0

(e)

Figure 3-8: Image as a mixture of visual topics (Expt. (2)) using 7 learned topics with fixed background parameters). (a),(c) Original images. (b),(d) Images as mixtures of visual topics. Visual words with topic posterior given in Equation 3.15 greater than 0.88 are shown. In total, there are 926 and 391 elliptical regions in the top and bottom row images respectively. A key to the colors is given in (e) along with the number of visual words assigned to each topic and the probability of each topic occuring in the two images. Notice that the displayed visual words agree well with the object categories.

different regions of the image, producing a weak segmentation of the image.

To evaluate the goodness of the segmentations, we use a set of ground-truth bounding boxes for the set of face images [24] and compute a score that measures the percentage of overlap between the ellipses used for segmentation and the bounding box:

$$\rho_i = \frac{B_i \cap E_i}{B_i \cup E_i} \tag{3.16}$$

where B_i is the set of pixels inside the bounding box for image i and E_i is the set of pixels inside all of the ellipses used for segmenting the face in image i. We compute a score ρ over all of the face images by taking the average of the ρ_i. For all of displayed segmentations in this chapter that has the set of face images in the training corpus, we set the threshold for diplaying a word (e.g. 0.88 above) to be the one that empirically maximizes this score. For the segmentation experiment above, we get $\rho = 0.50$.

It seems quite remarkable that we are able to get a segmentation with the bag-of-words assumption since all spatial structure between the words is thrown away. However, it is important to note that the words themselves and the overlap that occurs between neighboring words seem to retain much of the spatial structure of the desired objects (i.e. we would probably get a completely different result if the pixels were randomly shuffled).

Notice that Equation 3.15 depends not only on the probability of a visual word occuring in a particular topic, but also on the probability of that topic occuring in the given image. This allows us to overcome the phenomena of polysemy. In the text domain, polysemy occurs when a word has two different meanings (e.g. 'bank' as in (i) a money keeping institution, or (ii) a river side). Figure 3-9 shows an example of a visual word with relatively high probability in two different topics. For these examples, the words are assigned to the correct topics for the segmentation task.

Expt. (5) Image segmentation for faces. We now investigate how doublets can improve image segmentation. To illustrate this clearly, we experiment with only the face and background images. We learn a set of parameters for the background

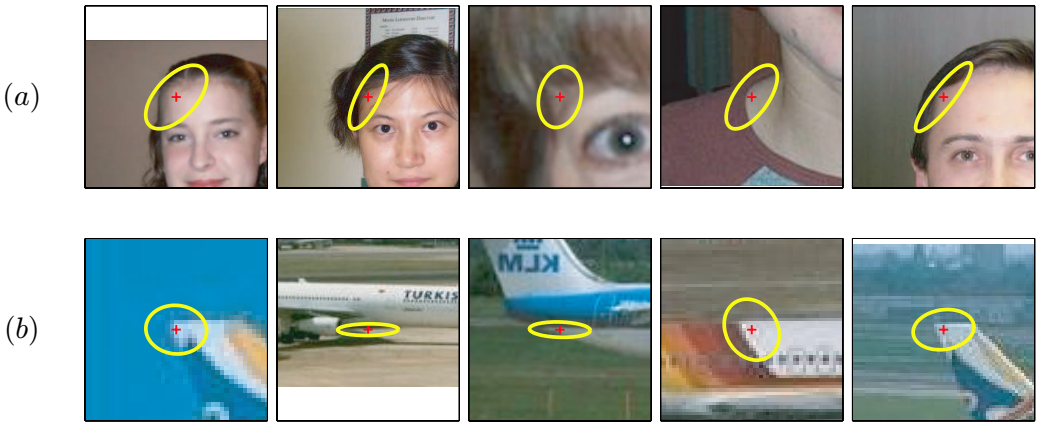


Figure 3-9: Example of a polysemic visual word (this word has high probability in two different topics). Each row depicts examples of the visual word occurring in two different topics. For the segmentation task, we are able to overcome polysemy since the probability that a given visual word in an image belongs to a topic, as given in Equation 3.15, depends on the likelihood of the topic occurring in the image in addition to the likelihood of the visual word occurring in the topic. For each of these examples, the word was assigned to the correct topic and hence induced the correct segmentation.

class by fitting an LDA model with $K = 3$ on half of the background images (400 images in total). We then fit an LDA model with $K = 4$ on the faces and remaining background images with the learned background parameters held fixed. A doublet vocabulary is then formed from the top 100 visual words for the face topic using the inferred ϕ parameters. We then combine the original vocabulary (cf singlets) and the new doublet vocabulary and repeat the above steps using the new vocabulary to fit an LDA model for $K = 4$ with fixed background parameters. We initially fit an LDA model on the singlet vocabulary to reduce the size of the doublet vocabulary. Figure 3-10 shows examples of doublets and compares the resulting segmentation using singlets versus doublets. Notice that we get a somewhat cleaner segmentation with doublets. Furthermore, the percentage of overlap increases from $\rho = 0.51$ for singlets to $\rho = 0.56$ for doublets.

If we perform Expt. (2) and form doublets from the top 100 words in all of the non-background topics using pre-learned fixed parameters for the background, then the overlap score decreases to $\rho = 0.51$. This suggests that there is some benefit in topic-specific doublets.

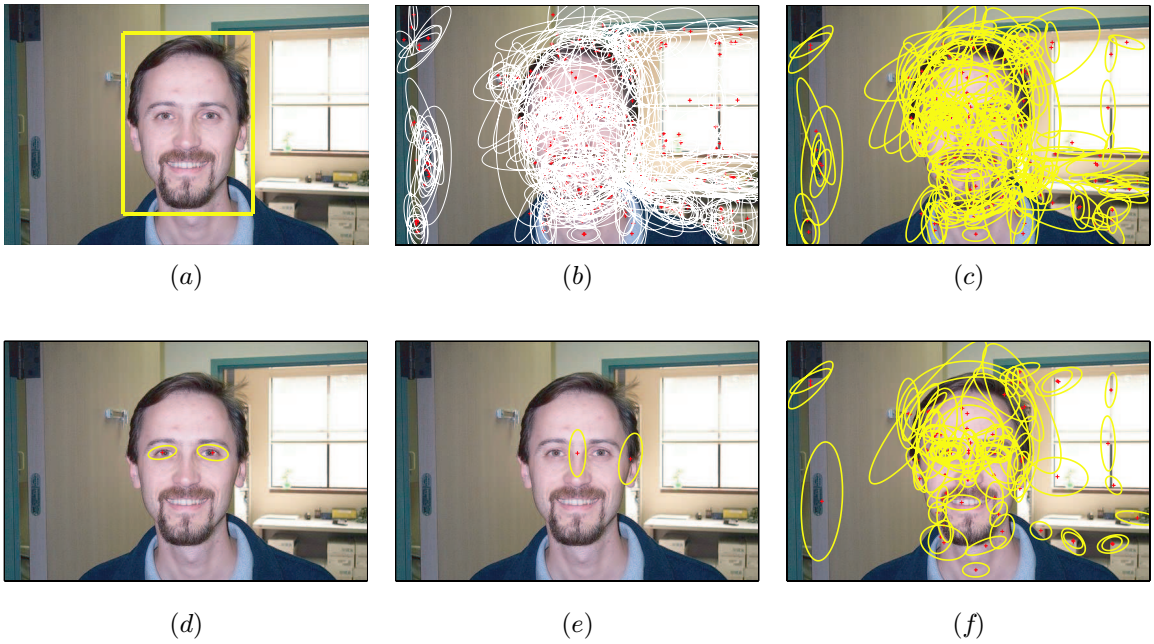


Figure 3-10: **Improving object segmentation.** (a) The original frame with ground truth bounding box. (b) All 416 detected elliptical regions superimposed on the image. (c) Resulting segmentation after fitting an LDA model to the "singlet" visual words in Expt. (5). (d) and (e) show examples of 'doublets'. (f) Segmentation obtained using doublets. Notice that we get a cleaner segmentation using doublets. For (c) and (f) respectively, the threshold that we use for displaying the singlets is 0.89 and doublets is 0.70.

Notice the level of supervision to achieve this segmentation: the images are an unordered mix of faces and backgrounds. It is not necessary to label which is which, yet both the face objects *and their segmentation* are learned. The supervision provided is the number of topics K to learn and the separate set of background images to pre-learn the background topics.

Expt. (6) Image Segmentation on the MIT Dataset. We consider the entire MIT Objects and Scenes dataset. We fit an LDA model with $K = 10$ and form doublets using the top 40 visual words from each topic according to the recovered ϕ parameters. A second LDA model is fitted using the combined singlet and doublet vocabulary. Figures 3-11 and 3-12 show examples of segmentations induced by 4 of the 10 learned topics. These topics, more so than the rest, have a clear semantic interpretation, and cover objects such as computers, buildings, trees, and bookshelves.

Notice that the results clearly demonstrate that: (i) images can be accessed by the multiple objects they contain (in contrast to GIST [61], for example, which classifies an entire image); (ii) the topics induce segmentations of multiple instances of objects in each image.

3.5 Conclusions

We demonstrated that it is possible to learn visual object classes simply by looking. To do this, we described the Latent Dirichlet Allocation model and showed how to convert images to the text domain by means of vector-quantized SIFT features and doublets, which form a vocabulary of visual words. For a given set of unlabelled images, we discovered meaningful object parts and clustered the images into the appropriate object classes with high reliability (cf. Table 3.1). We were able to improve the results for those objects that caused confusion by training on side data, then fixing the learned topic parameters, and finally learning the remaining parameters on the rest of the dataset.

We reproduced the experiments of [33] using a set of training images to fit an LDA model and then using the learned parameters to classify a set of unseen images. For the training phase, we did not provide any class labels; we only specified the number of topics to discover. We obtained very competitive performance without much supervision.

Finally, we showed that we are able to localize objects remarkably well, in spite of the bag-of-words assumption, using those visual words with the highest posterior probability. Since this posterior depends on the likelihood of an object class occurring in an image in addition to the likelihood of a visual word occurring in an object class, we showed that this allows us to overcome polysemy. We improved the segmentation results by incorporating the doublet visual words.

Figure 3-11: Example segmentations induced by four (out of 10) discovered topics on the MIT dataset. Examples from the first 20 most probable images for each topic are shown. For each topic the top row shows the original images and the bottom row shows visual words (doublets) belonging to that particular topic in that image. Note that we can give semantic interpretation to these topics: (a) covers computers; (b) covers building regions; (c) covers bookshelves; (d) covers trees and grass.

Figure 3-12: Example segmentations on the MIT dataset for the 10 topic decomposition. Left: the original image. Middle: all detected regions superimposed. Right: the topic induced segmentation. The topics depicted are from Figure 3-11. The color key is: a-cyan, b-red, c-magenta, d-green. Notice that each image is segmented into several 'topics'.

Chapter 4

Using Multiple Segmentations to Discover Objects and their Extent in Image Collections

4.1 Introduction

In Chapter 3 we posed the question, given a Gargantuan number of images, "Is it possible to learn visual object classes simply from looking at images?". That is, if our data set contains many instances of visually similar object classes, can we *discover* these object classes? In this chapter we extend this question to "Is it possible to learn visual object classes *and their segmentations* simply from looking at images?"

To automatically discover objects in an image collection, two very challenging issues must be addressed: (i) how to recognize visually similar objects; and (ii) how to segment them from their background. But, in a sense, both object recognition and image segmentation can be thought of as parts of one large *grouping problem* within the space of an entire dataset. Given a stack of all images in the dataset, groups representing similar objects can be seen as volumes in that stack. Projecting such

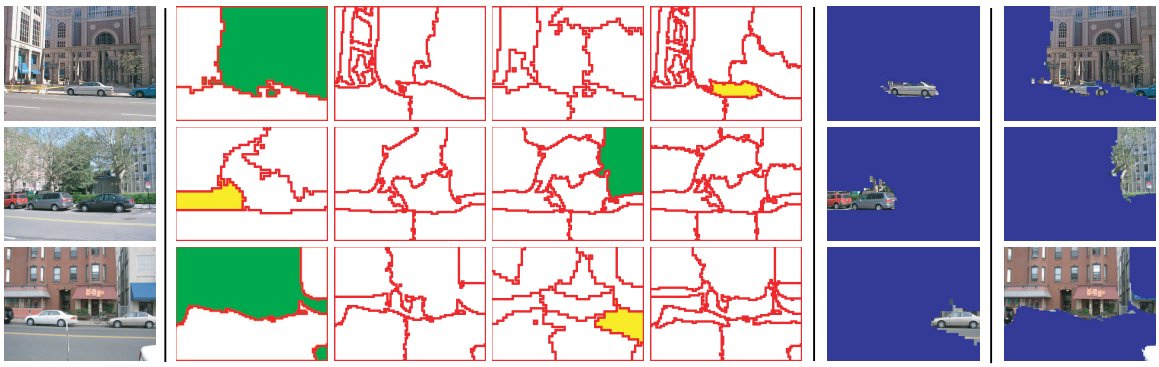


Figure 4-1: **Problem summary**. Given a set of input images (first column), we wish to discover object categories and infer their spatial extent (e.g. cars and buildings: final two columns). We compute multiple segmentations per image (a subset is depicted in the second through fifth columns; all of the segmentations for the first row are shown in Figure 4-3). The task is to sift the good segments from the bad ones for each discovered object category. Here, the segments chosen by our method are shown in green (buildings) and yellow (cars).

volumes onto a particular image gives segmentation; projecting onto the image index gives recognition. Our aim here is to couple object-based matching/recognition and image-based segmentation into a general grouping framework.

To be concrete, the problem that we wish to solve is the following: given a large dataset of images (containing multiple instances of several object classes), retrieve segmented instances grouped into object classes. The hope is that this will recover commonly occurring object classes in the dataset (e.g. cars, buildings). Our approach is to first obtain multiple segmentations of each image, and to make the assumption that each object instance is correctly segmented by at least one segmentation. The problem is then reduced to finding coherent groups of correctly segmented objects within this large "soup" of candidate segments, i.e. one of grouping in the space of candidate image segments. Our approach is illustrated in figure 4-1.

In Section 4.2 we give motivation for combining segmentation with the topic discovery models of Chapter 3. In Section 4.3, we provide reasons for producing multiple segmentations on images for our task. In Section 4.4, we describe the overall algorithm. We also describe a way of choosing good segments that are representative of the discovered topics. In Section 4.5, we provide qualitative and quantitative results for our approach. In Section 4.6 we conclude. The results in this chapter were devel-

oped in collaboration with Alyosha Efros, Josef Sivic, William Freeman, and Andrew Zisserman and appeared at the 2006 Conference on Computer Vision and Pattern Recognition [67].

4.2 Grouping visual words

One major issue noticed by several groups [65, 77], is that the "visual words" are not always as descriptive as their text counterparts. While some visual words do capture high-level object parts, (e.g. wheels, eyes, airplane wingtips), many others end up encoding simple oriented bars and corners and might more appropriately be called "visual phonemes" or even "visual letters".

Moreover, there is a proportion of visual synonyms – several words describing the same object or object part, and, more problematically, visual polysemy – the same word describing several different objects or object parts. The problem of visual polysemy becomes apparent when we consider how an image is represented in the "bag of words" document model. All visual words in an image are placed into a single histogram, losing all spatial and neighborhood relationships. Suppose a car is described by ten visual words. Does the presence of these ten words in an image imply that it contains a car? Not necessarily, since these ten words did not have to occur together spatially, but anywhere in the image. Of course, if the object and its background are highly correlated (e.g. cars and roads or cows and grass), then modeling the entire image can actually help recognition. However, this is unlikely to scale as we look at a large number of object classes. Therefore, what we need is a way to group visual words spatially [32, 83] to make them more descriptive.

4.3 Multiple segmentation approach

In this chapter we propose to use image segmentation as a way to utilize visual grouping cues to produce groups of related visual words. In theory, the idea sounds simple: compute a segmentation of each image so that each segment corresponds to

a coherent object. Then cluster similar segments together using the "bag of words" representation. However, image segmentation is not a solved problem. It is naive to expect a segmentation algorithm to partition an image into its constituent objects – in the general case, you need to have solved the recognition problem already! In practice, some approaches, like Mean-shift [19], perform only a low-level over-segmentation of the image (superpixels). Others, like Normalized Cuts [75] attempt to find a global solution, but often without success (however, see Duygulu *et al.* [23] for a clever joint use of segments and textual annotations).

Recently, Hoiem *et al.* [42] have proposed a surprisingly effective way of utilizing image segmentation without suffering from its shortcomings. For each image, they compute *multiple* segmentations by varying the parameters of the segmenting algorithm. Each of the resulting segmentations is still assumed to be wrong – but the hope is that *some* segments in *some* of the segmentations will be correct. For example, consider the images in figures 4-1 and 4-3. None of the segmentations are entirely correct, but most objects get segmented correctly at least once. This idea of maintaining multiple segmentations until further evidence can be used to disambiguate is similar to the approach of Borenstein *et al.* [16].

The problem now becomes one of going through a large "soup" of (overlapping) segments and trying to discover the good ones. But note that, in a large image dataset with many examples of the same object, the good segments (i.e. the ones containing the object) will all be represented by a similar set of visual words. The bad segments, on the other hand, will be described by a random mixture of object-words and background-words. To paraphrase Leo Tolstoy [88]: *all good segments are alike, each bad segment is bad in its own way.* This is the main insight of the paper: segments corresponding to objects will be exactly the ones represented by coherent groups (topics), whereas segments overlapping object boundaries will need to be explained by a mixture of several groups (topics). We exploit this insight in the object discovery algorithm described next.

Given a large, unlabeled collection of images:

1. For each image in the collection, compute multiple candidate segmentations, e.g. using Normalized Cuts [75] (section 4.4.1).
2. For each segment in each segmentation, compute a histogram of "visual words" [78] (section 3.2).
3. Perform topic discovery on the set of *all segments* in the image collection (using Latent Dirichlet Allocation [14]), treating each segment as a document (section 3.3.2).
4. For each discovered topic, sort *all segments* by how well they are explained by this topic (section 4.4.2).

Figure 4-2: Algorithm overview.

4.4 The Algorithm

Given a large, unlabeled collection of images, our goal is to automatically discover object categories with the objects segmented out from the background. Our algorithm is summarized in figure 4-2.

The result is a set of discovered topics, where the top-ranked discovered segments correspond to the objects within that topic. The rest of the section will describe the novel steps of the algorithm in detail.

4.4.1 Generating multiple segmentations

Our aim is to produce sufficient segmentations of each input image to have a high chance of obtaining a few "good" segments that will contain potential objects. There are approaches in the literature for sampling likely segmentations [94] and multiscale segmentations [74]. But since we are not relying on the full segmentation to be correct, the particular choice of a segmentation algorithm is not that critical. Indeed, the fact that segmentation algorithms are not particularly stable as one perturbs their parameters is exactly what we use to obtain a variety of different segmentations.

We have chosen the Normalized Cuts framework [75], because it aims to produce a global segmentation with large segments that have a chance to be objects. The affinity metric we use is the intervening contour cue based on the texture-suppressing

boundary detector of Martin *et al.* [55]. To produce multiple segmentations, we varied two parameters of the algorithm: the number of segments K and the size of the input image. We typically set $K = 3, 5, 7, 9$ segments and applied these settings at 2 image scales: 50- and 100-pixels across (for the LabelMe dataset, we also used $K = 11, 13$ and for the MSRC dataset we added a third scale at 150-pixels across). This results in up to 12 different segmentations per image, for a total of up to 96 (overlapping) segments. Figure 4-3 shows the set of resulting segmentations for sample images.

4.4.2 Sorting the soup of segments

We wish to find good segments within each topic. We sort the segments by the similarity of the visual word distribution (normalized histogram) within each segment to the learned multinomial weights ϕ_k for a given topic k. Let ϕ_s be the multinomial parameter describing the visual word distribution within a segment. We sort the segments based on the Kullback-Leibler (KL) divergence $D(p(w|s, \phi_s)||p(w|z, \phi_k))$ between the two distributions over visual words.

Figure 4-3 shows discovered objects segmented out of the image. We also show the generated multiple segmentations and have weighted each segment based on their KL divergence score. Notice that often there is a tight segmentation of the discovered objects.

4.5 Results

In this section, we show qualitative results on several datasets and report quantitative results on two tasks: (i) the retrieval task, where we wish to evaluate whether or not the top ranked images for a particular topic contain the discovered object; and (ii) the segmentation task, where we wish to evaluate the quality of object segmentation and the proportion of well-segmented highly-ranked objects.

Image datasets: We investigated three datasets: Caltech [33], MSRC [102], and LabelMe. A summary of the object categories and number of images used appears

in Table 4.1. We tested on progressively more difficult datasets. For the Caltech set, we used four object categories – the 'Caltech four' of [33] – each containing a single instance appearing in flat or cluttered background, and a set of background images. The MSRC set contains 23 object and scene categories. Many of the objects in this set are prominently featured and located close to the center of the image. There are also many images containing multiple objects, with some that are occluded. The LabelMe dataset is a more difficult collection of scene images where the objects are not controlled and appear in their natural habitat. For this set, we queried for images containing cars, trees, and buildings. The query resulted in 1554 images, containing many other additional objects.

Figures 4-4 through 4-7 show montages of segments for each topic, sorted by their KL divergence score. Note that for each discovered category, the objects are reasonably segmented and are consistent. The depicted segments each come from a different image to avoid showing multiple segments of the same object.

To assess the contributions of the different steps of the algorithm, we evaluate: (a) the proposed algorithm (of Figure 4-2), (b) swapping the LDA model for the simpler pLSA model to evaluate the contribution of the Dirichlet prior over the multinomial weights, (c) using only a single segmentation for each image (in conjunction with the LDA model) to evaluate the contribution of computing multiple segmentations for each image, (d) our previous method [77], where we use no segmentation at all and each image is treated as a separate document, with the object extent determined by the union of visual words having high posterior probability (greater than 0.5) for a particular topic. For all tests, each method was run 10 times and the run with the highest likelihood was used.

Image retrieval performance is evaluated on the MSRC database, where labels indicating object presence/absence are available. The evaluation is performed for four objects: 'bicycles', 'cars', 'signs' and 'windows'. For the proposed method (a), top ranked images for corresponding topics are shown in Figure 4-6. Precision-recall curves were computed and the average precision is reported in Table 4.2 for the tested methods.

For 'bicycles' and 'windows', the proposed method performs on par or better than the other methods. Method (d), where no segmentation is used, performs best on 'cars' because it is learning about other objects in the scene that overlap significantly with the target object (e.g. roads). These other objects predict well the presence and location of the target object for the tested dataset. This effect may also explain why method (c), which uses a coarse segmentation, performs better on 'signs'. Method (b) performs significantly worse than the other methods. We believe this is due to pLSA overfitting the data, because of the lack of a Dirichlet prior on the document-topic coefficients [14]. In our earlier work [77], we did not observe a significant difference in performance between pLSA and LDA. This might be due to the smaller number of topics and documents used. Our earlier work had only about 4K documents and 4-7 topics, whereas in this work we have about 200K documents and 25 topics.

The segmentation accuracy is evaluated on the LabelMe dataset, where ground truth object segmentation was labelled for each tested method on the top twenty returned images for topics covering four objects: 'buildings', 'cars', 'roads' and 'sky'. Let R and GT be respectively the set of pixels in the retrieved object segment and the ground truth segmentation of the object. The performance score ρ measures the area correctly segmented by the retrieved object segment. It is the ratio of the intersection of GT and R to the union of GT and R, i.e. $\rho = \frac{GT \cap R}{GT \cup R}$. If more than one ground truth segmentation intersects R, then we use the one that results in the highest score. The score is then averaged over the top 20 retrieved object segments. The results are summarized in table 4.3.

Our method scores about the same or better than the other methods on 'roads' and 'sky' objects. Methods (b) and (c) perform better on 'building' and 'car' objects respectively. Note that this comparison takes into account only the top 20 segments for each method and does not measure the number of top-ranked high quality segments. For the 'car' object, we have closely inspected the results of methods (a) and (c). While the quality of segmentations is worse in the top 20 returned images, the proposed method (a) outperforms single segmentation LDA (c) over the top 500 returned images (the proposed method returns about 15% more high quality seg-

Dataset	# of images	# of categories
Caltech [33]	4,090	4 + background
MSRC [102]	4,325	23 object and scene categories
LabelMe	1,554	cars, buildings, trees
Google Maps [1]	4,096	satellite imagery of Boston

Table 4.1: Summary of datasets used in this paper.

Method	bicycles	cars	signs	windows
(a) Mult. seg. LDA	0.69	0.77	0.43	0.74
(b) Mult. seg. pLSA	0.67	0.28	0.34	0.57
(c) Sing. seg. LDA	0.67	0.73	0.46	0.72
(d) No seg. LDA	0.64	0.85	0.40	0.74
(e) Chance	0.06	0.12	0.04	0.15

Table 4.2: Average precisions for the tested methods on several objects from the MSRC dataset.

Method	buildings	cars	roads	sky
(a) Mult. seg. LDA	0.53	0.21	0.41	0.77
(b) Mult. seg. pLSA	0.59	0.09	0.16	0.77
(c) Sing. seg. LDA	0.55	0.29	0.32	0.65
(d) No. seg. LDA	0.47	0.16	0.14	0.16

Table 4.3: Segmentation score for the tested methods on several objects with ground truth labels from the LabelMe dataset. See text for a description of the segmentation score.

ments). This suggests that using multiple segmentations generates more high quality segments in the dataset.

4.6 Conclusion

By combining multiple candidate segmentations with probabilistic document analysis methods, we have developed an algorithm that finds and segments visual topics within an unlabeled collection of images. The discovered topics relate closely to object classes within the training set, such as cars, bicycles, faces, signs, trees, and windows. (In comparison with the recent results of Winn *et al.* [103], we should note that ours are obtained completely automatically from a large corpus of unlabeled images,

whereas theirs are computed from a small set of single-object-category images.) These results show the power of classical segmentation methods augmented with the power of modern document analysis methods.

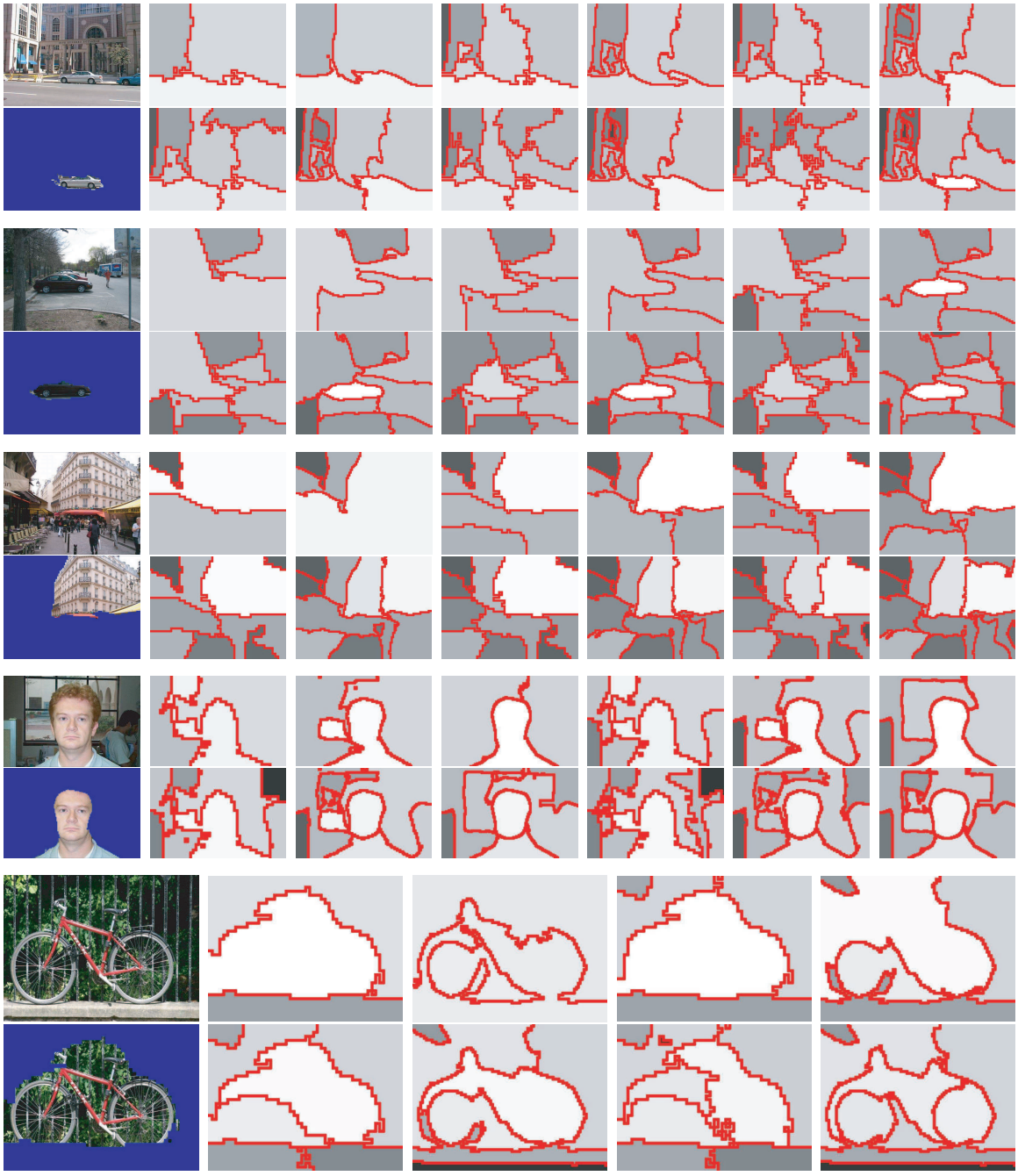


Figure 4-3: How multiple candidate segmentations are used for object discovery. The top left image of every pair of rows is the input image, which is segmented using Ncuts at different parameter settings into 12 different sets of candidate regions. The explanatory power of each candidate region is evaluated as described in the text; we illustrate the resulting rank by the brightness of each region. The image data of the top-ranked candidate region is shown in the bottom left, confirming that the top-ranked regions usually correspond to objects.

Figure 4-4: Top segments for 4 topics (out of 10) discovered in the Caltech dataset. Note how the discovered segments, learned from a collection of unlabelled images, correspond to motorbikes, faces, and cars.

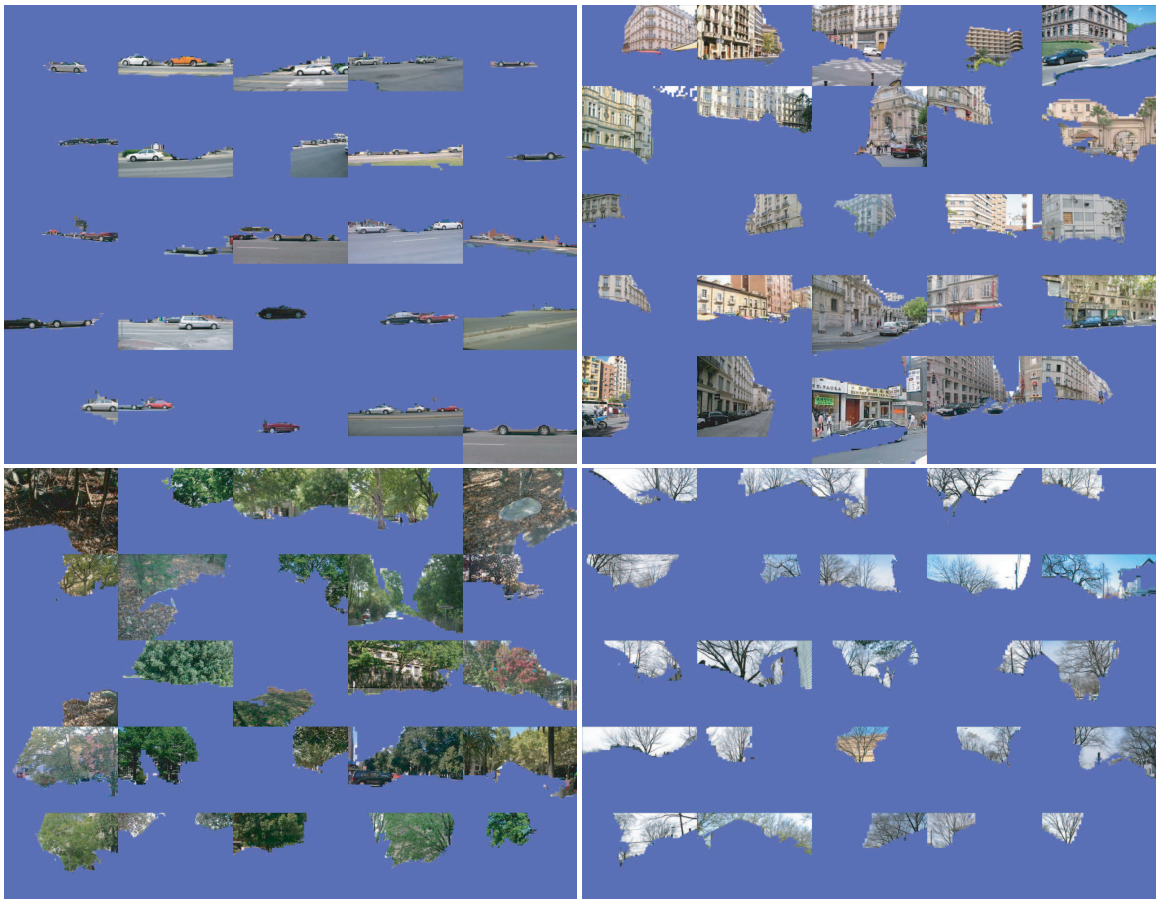


Figure 4-5: Top segments for 4 (out 20) topics discovered in the LabelMe dataset. Note how the discovered segments, learned from a collection of unlabeled images, correspond to cars, buildings, and two types of trees.

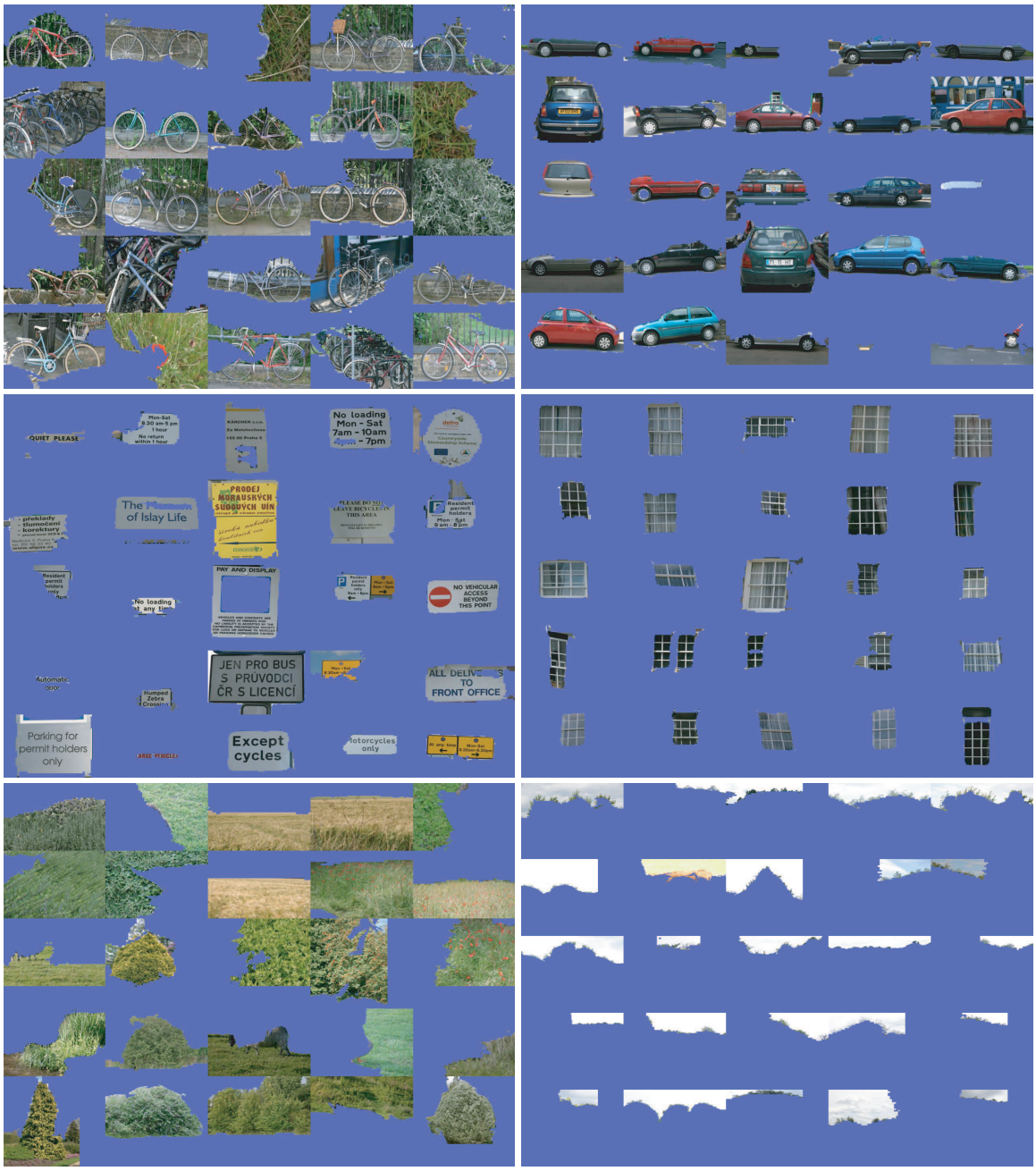


Figure 4-6: Top 21 segments for 6 topics (out of 25) discovered in the MSRC dataset. Note how the discovered segments, learned from a collection of unlabeled images, correspond to cars, bicycles, signs, windows, grass, and sky categories, respectively.

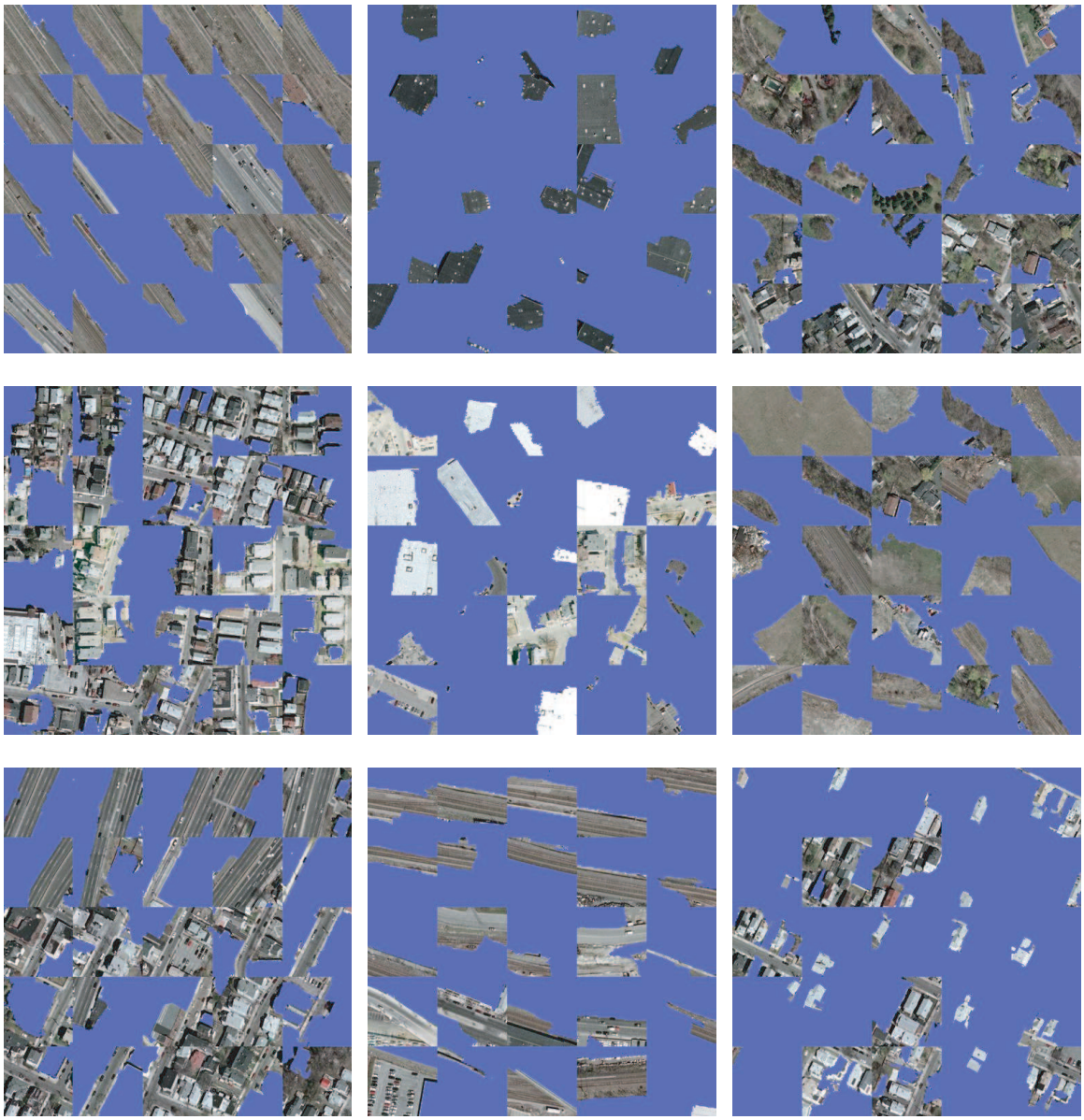


Figure 4-7: Top 25 segments for 9 topics (out of 25) discovered on a set of city satellite images extracted from Google Maps [1]. Notice how we discover different city structures, such as roads, buildings, and houses.

Chapter 5

Object Recognition by Scene Alignment

5.1 Introduction

The recognition of objects in a scene often consists of matching representations of image regions to an object model while rejecting background regions. Recent examples of this approach include aligning pictorial cues [31], shape correspondence [9], and modeling the constellation of parts [33]. Other models, exploiting knowledge of the scene context in which the objects reside, have proven successful in boosting object recognition performance [89, 93, 83, 41, 66]. These methods model the relationship between scenes and objects and allow information transfer across the two.

Here, we exploit scene context using a different approach: we formulate the object detection problem as one of aligning elements of the entire scene to a large database of labeled images. The background, instead of being treated as a set of outliers, is used to guide the detection process. Our approach relies on the observation that when we have a large enough database of labeled images, we can find with high probability some images in the database that are very close to the query image in appearance, scene contents, and spatial arrangement [37, 90]. Since the images in the database are partially labeled, we can transfer the knowledge of the labeling to the query image. Figure 5-1 illustrates this idea. With these assumptions, the problem

of object detection in scenes becomes a problem of aligning scenes. The main issues are: (1) Can we find a big enough dataset to span the required large number of scene configurations? (2) Given an input image, how do we find a set of images that aligns well with the query image? (3) How do we transfer the knowledge about objects contained in the labels?

(a) Input image (b) Images with similar scene configuration (c) Output image with object labels transferred

Figure 5-1: Overview of our system. Given an input image, we search for images having a similar scene configuration in a large labeled database. The knowledge contained in the object labels for the best matching images is then transfered onto the input image to detect objects. Additional information, such as depth-ordering relationships between the objects, can also be transferred.

The LabelMe dataset (c.f. Chapter 2) is well-suited for this task, having a large number of images and labels spanning hundreds of object categories. Recent studies using non-parametric methods for computer vision and graphics [90, 37] show that when a large number of images are available, simple indexing techniques can be used to retrieve images with object arrangements similar to those of a query image.

The core part of our system is the transfer of labels from the images that best match the query image. We assume that there are commonalities amongst the labeled objects in the retrieved images and we cluster them to form candidate scenes. These scene clusters give hints as to what objects are depicted in the query image and their likely location. We describe a relatively simple generative model for determining which scene cluster best matches the query image and use this to detect objects.

The remaining sections are organized as follows: In Section 5.2, we describe our representation for scenes and objects. We formulate a model that integrates the information in the object labels with object detectors in Section 5.3. In Section 5.4,

we extend this model to allow clustering of the retrieved images based on the object labels. We show experimental results of our system output in Section 5.5, and conclude in Section 5.6. The results in this chapter were developed in collaboration with Antonio Torralba, Ce Liu, Rob Fergus, and William T. Freeman and appeared at the 2007 Conference on Neural Information Processing Systems [68].

5.2 Matching Scenes and Objects with the Gist Feature

We describe the gist feature [61], which is a low dimensional representation of an image region and has been shown to achieve good performance for the scene recognition task when applied to an entire image. To construct the gist feature, an image region is passed through a Gabor filter bank comprising 4 scales and 8 orientations. The image region is divided into a 4x4 non-overlapping grid and the output energy of each filter is averaged within each grid cell. The resulting representation is a $4 \times 8 \times 16 = 512$ dimensional vector. Note that the gist feature preserves spatial structure information and is similar to applying the SIFT descriptor [54] to the image region.

5.2.1 Evaluation of the Gist Feature

We motivate the choice of the gist feature by evaluating performance on the scene recognition task. We use a dataset [61, 29, 47] consisting of 15 scene categories used in previous scene recognition work. We compare the gist feature against three other features: (1) raw color pixels (using SSD to compare scenes), (2) bag of words [29] sampled densely over a regular grid on the image (note that spatial information is lost), (3) pyramid matching [47], which uses histograms of visual words that are computed over different regions of the image in order to keep information about the spatial organization of the image. For the classification task, we train a support vector machine (SVM) [96] using SVM-light [45] for each scene class over the different features. In all cases, we use a radial basis kernel.

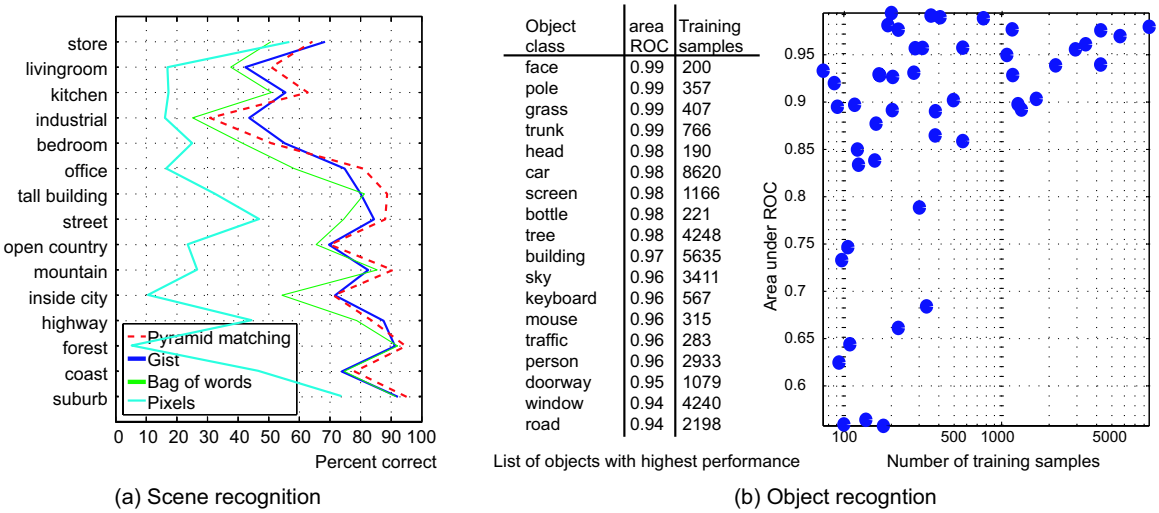


Figure 5-2: (a) Comparison of gist [61], raw color pixels, bag of words [29], and pyramid matching [47] performance on the scene recognition task. Notice that features incorporating spatial information perform best (gist, pyramid matching, which perform similarly across the 15 scene categories). The average performance for the different feature sets are gist: 71%, raw color pixels: 34.6%, bag of words: 64.1%, and pyramid matching: 74%. (b) Object recognition using gist features as a function of training set size. We enumerate the performance for a few object categories in the table and show a scatter plot for all of the object categories tested. Notice that the performance increases as more training data is available.

Figure 5-2(a) shows the performance of the different features on the scene recognition task[1]. We use 100 images from each scene class for training and the remaining images for testing (as in [47]). The raw intensities perform very poorly. The features that perform best are those that incorporate spatial structure into the representation (gist and pyramid matching), resulting in similar performance across the scene categories. The performance of the gist on this task provides motivation for its use throughout this work, although the system is flexible and can accomodate any other feature set.

Given the success of the gist feature applied to entire images for scene recognition, we may also consider applying gist to smaller image patches to recognize objects. To investigate recognition when the object location is known, we utilized the object labels from the LabelMe dataset to extract tight bounding box crops around the objects. More specifically, we evaluate performance on object categories with at least 100 labeled examples. We use 50 examples per object class for testing and the remainder for training. To avoid over-fitting, we ensure that the images in the test set were taken in locations different from the training set. The amount of training samples per object class varies greatly, with some classes having over 5000 examples. For classification, we train an SVM classifier for each class with a radial basis kernel.

Figure 5-2(b) shows recognition results for the gist feature. We show classification performance (area under ROC) as a function of the amount of training data. Notice that when the training set size is small, there is high variance in the performance across the different categories. Some objects need more training samples in order to generalize to new instances. Other classes can be learned with very few training samples (e.g. traffic signs) as they are very regular in visual appearance. When there are many training samples, recognition performance tends to be good.

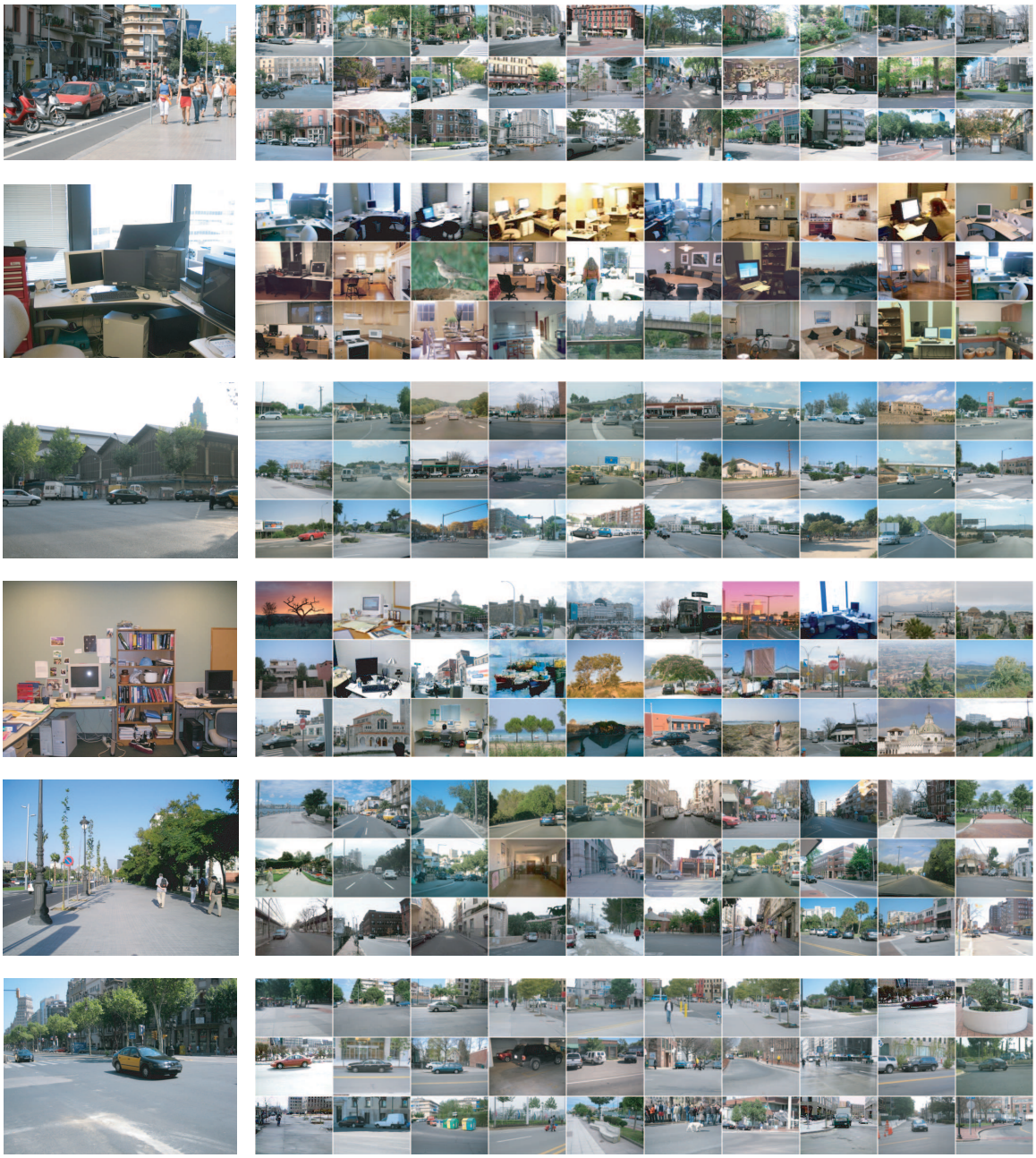


Figure 5-3: Retrieval set images. Each row depicts an input image (on the left) and 30 images from the LabelMe dataset [70] that best match the input image using the gist feature [61] and L1 distance (the images are sorted by their distances in raster order). Notice that the retrieved images generally belong to similar scene categories. Also the images contain mostly the same object categories, with the larger objects often matching in spatial location within the image. Many of the retrieved images share similar geometric perspective.

5.2.2 Finding Retrieval Sets

We consider the task of retrieving a set of images (which we refer to as the *retrieval set*) that closely matches the scene contents and geometrical layout of an input image. Figure 5-3 shows retrieval sets for typical input images using the gist feature. We show the top 30 closest matching images from the LabelMe database based on the L1-norm distance, which is robust to outliers. Notice that the gist feature retrieves images that match the scene type of the input image. Furthermore, many of the objects depicted in the input image appear in the retrieval set, with the larger objects residing in approximately the same spatial location relative to the image. Also, the retrieval set has many images that share a similar geometric perspective. Of course, not every retrieved image matches well and we account for outliers in Section 5.4.

We evaluate the ability of the retrieval set to predict the presence of objects in the input image. For this, we found a retrieval set of 200 images and formed a normalized histogram (the histogram entries sum to one) of the object categories that were labeled. We compute performance for object categories with at least 200 training examples and that appear in at least 15 test images. We compute the area under the ROC curve for each object category. As a comparison, we evaluate the performance of an SVM applied to gist features by using the maximal score over a set of bounding boxes extracted from the image. The area under ROC performance of the retrieval set versus the SVM is shown in Figure 5-4 as a scatter plot, with each point corresponding to a tested object category. As a guide, a diagonal line is displayed; those points that reside above the diagonal indicate better SVM performance (and vice versa). Notice that the retrieval set predicts well the objects present in the input image and outperforms the detectors based on local appearance information (the SVM) for most object classes.

[1]Note that the performance differs from that as reported in [47]. The difference is that we have cropped all of the images to be 256×256 pixels. The original dataset has images of different resolutions and aspect ratios that correlate with the scene categories and thereby provides non-visual discriminant cues.

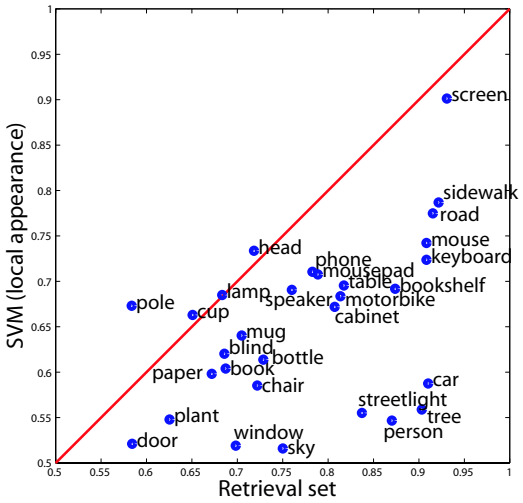


Figure 5-4: Evaluation of the goodness of the retrieval set by how well it predicts which objects are present in the input image. We build a simple classifier based on object counts in the retrieval set as provided by their associated LabelMe object labels. We compare this to detection based on local appearance alone using an SVM applied to bounding boxes in the input image (the maximal score is used). The area under the ROC curve is computed for many object categories for the two classifiers. Performance is shown as a scatter plot where each point represents an object category. Notice that the retrieval set predicts well object presence and in a majority cases outperforms the SVM output, which is based only on local appearance.

5.3 Utilizing Retrieval Set Images for Object Detection

In Section 5.2, we observed that the set of labels corresponding to images that best match an input image predict well the contents of the input image. In this section, we will describe a model that integrates local appearance with object presence and spatial likelihood information given by the object labels belonging to the retrieval set.

We wish to model the relationship between object categories o, their spatial location x within an image, and their appearance g. For a set of N images, each having M_i object proposals over L object categories, we assume a joint model that factorizes as follows:

$$p(o, x, g|\theta, \phi, \eta) = \prod_{i=1}^{N} \prod_{j=1}^{M_i} \sum_{h_{i,j}=0}^{1} p(o_{i,j}|h_{i,j}, \theta)\ p(x_{i,j}|o_{i,j}, h_{i,j}, \phi)\ p(g_{i,j}|o_{i,j}, h_{i,j}, \eta) \quad (5.1)$$

We assume that the joint model factorizes as a product of three terms: (i) $p(o_{i,j}|h_{i,j} = m, \theta_m)$, the likelihood of which object categories will appear in the image, (ii) $p(x_{i,j}|o_{i,j} = l, h_{i,j} = m, \phi_{m,l})$, the likely spatial locations of observing object category l in the image, and (iii) $p(g_{i,j}|o_{i,j} = l, h_{i,j} = m, \eta_{m,l})$, the appearance likelihood of object category l. We let $h_{i,j} = 1$ indicate whether object category $o_{i,j}$ is actually present in location $x_{i,j}$ ($h_{i,j} = 0$ indicates absence). Figure 5-5 depicts the above as a graphical model. We use plate notation, where the variable nodes inside a plate are duplicated based on the counts depicted in the top-left corner of the plate.

We instantiate the model as follows. The spatial location of objects are parameterized as bounding boxes $x_{i,j} = (c^x_{i,j}, c^y_{i,j}, c^w_{i,j}, c^h_{i,j})$ where $(c^x_{i,j}, c^y_{i,j})$ is the centroid and $(c^w_{i,j}, c^w_{i,j})$ is the width and height (bounding boxes are extracted from object labels by tightly cropping the polygonal annotation). Each component of $x_{i,j}$ is normalized with respect to the image to lie in $[0, 1]$. We assume θ_m are multinomial parameters and $\phi_{m,l} = (\mu_{m,l}, \Lambda_{m,l})$ are Gaussian means and covariances over the bounding box parameters. Finally, we assume $g_{i,j}$ is the output of a trained SVM applied to a gist feature $\tilde{g}_{i,j}$. We let $\eta_{m,l}$ parameterize the logistic function $(1 + \exp(-\eta_{m,l}\ [1\ g_{i,j}]^T))^{-1}$.

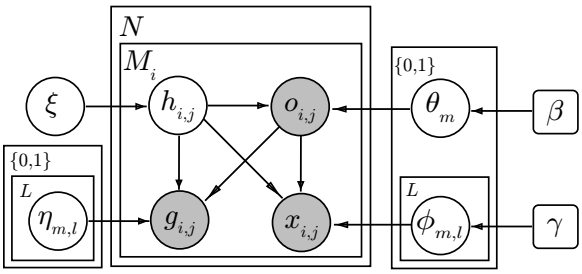


Figure 5-5: Graphical model that integrates information about which objects are likely to be present in the image o, their appearance g, and their likely spatial location x. The parameters for object appearance η are learned offline using positive and negative examples for each object class. The parameters for object presence likelihood θ and spatial location ϕ are learned online from the retrieval set. For all possible bounding boxes in the input image, we wish to infer h, which indicates whether an object is present or absent.

The parameters $\eta_{m,l}$ are learned offline by first training SVMs for each object class on the set of all labeled examples of object class l and a set of distractors. We then fit logistic functions to the positive and negative examples of each class. We learn the parameters θ_m and $\phi_{m,l}$ online using the object labels corresponding to the retrieval set. These are learned by simply counting the object class occurrences and fitting Gaussians to the bounding boxes corresponding to the object labels.

For the input image, we wish to infer the latent variables $h_{i,j}$ corresponding to a dense sampling of all possible bounding box locations $x_{i,j}$ and object classes $o_{i,j}$ using the learned parameters θ_m, $\phi_{m,l}$, and $\eta_{m,l}$. For this, we compute the postierior distribution $p(h_{i,j} = m|o_{i,j} = l, x_{i,j}, g_{i,j}, \theta_m, \phi_{m,l}, \eta_{m,l})$, which is proportional to the product of the three learned distributions, for $m = \{0, 1\}$.

The procedure outlined here allows for significant computational savings over naive application of an object detector. Without finding similar images that match the input scene configuration, we would need to apply an object detector densely across the entire image for all object categories. In contrast, our model can constrain which object categories to look for and where. More precisely, we only need to consider object categories with relatively high probability in the scene model and bounding boxes within the range of the likely search locations. These can be decided based on thresholds. Also note that the conditional independences implied by the graphical

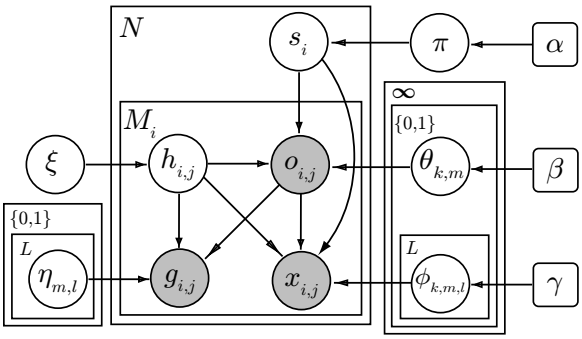


Figure 5-6: (a) Graphical model for clustering retrieval set images using their object labels. We extend the model of Figure 5-5 to allow each image to be assigned to a latent cluster s_i, which is drawn from mixing weights π. We use a Dirichlet process prior to automatically infer the number of clusters. We illustrate the clustering process for the retrieval set corresponding to the input image in (b). (c) Histogram of the number of images assigned to the five clusters with highest likelihood. (d) Montages of retrieval set images assigned to each cluster, along with their object labels (colors show spatial extent), shown in (e). (f) The likelihood of an object category being present in a given cluster (the top nine most likely objects are listed). (g) Spatial likelihoods for the objects listed in (f). Note that the montage cells are sorted in raster order.

model allows us to fit the parameters from the retrieval set and train the object detectors separately.

Note that for tractability, we assume Dirichlet and Normal-Inverse-Wishart conjugate prior distributions over θ_m and $\phi_{m,l}$ with hyperparemters β and $\gamma = (\kappa, \vartheta, \nu, \Delta)$ (expected mean ϑ, κ pseudocounts on the scale of the spatial observations, ν degrees of freedom, and sample covariance Δ). Furthermore, we assume a Bernoulli prior distribution over $h_{i,j}$ parameterized by $\xi = 0.5$. We hand-tuned the remaining parameters in the model. For $h_{i,j} = 0$, we assume the noninformative distributions $o_{i,j} \sim Uniform(1/L)$ and each component of $x_{i,j} \sim Uniform(1)$.

5.4 Clustering Retrieval Set Images for Robustness to Mismatches

While many images in the retrieval set match the input image scene configuration and contents, there are also outliers. Typically, most of the labeled objects in the outlier

images are not present in the input image or in the set of correctly matched retrieval images. In this section, we describe a process to organize the retrieval set images into consistent clusters based on the co-occurrence of the object labels within the images. The clusters will typically correspond to different scene types and/or viewpoints. The task is to then automatically choose the cluster of retrieval set images that will best assist us in detecting objects in the input image.

We augment the model of Section 5.3 by assigning each image to a latent cluster s_i. The cluster assignments are distributed according to the mixing weights π. We depict the model in Figure 5-6. Intuitively, the model finds clusters using the object labels $o_{i,j}$ and their spatial location $x_{i,j}$ within the retrieved set of images. To automatically infer the number of clusters, we use a Dirichlet Process prior on the mixing weights $\pi \sim Stick(\alpha)$, where $Stick(\alpha)$ is the stick-breaking process of Griffiths, Engen, and McCloskey [44, 60, 85] with concentration parameter α. In the Chinese restaurant analogy, the different clusters correspond to tables and the parameters for object presence θ_k and spatial location ϕ_k are the dishes served at a given table. An image (along with its object labels) corresponds to a single customer that is seated at a table.

Figures 5-7 and 5-8 illustrate the scene clustering model applied to the object labels of the retrieval sets belonging to several input images. Given an input image, shown in the left-most column, the five clusters with highest likelihood are visualized in the adjacent columns. The set of image montages in the top row show example retrieval images with highest likelihood that were assigned to each cluster by the Gibbs sampler. The total number of retrieval images that were assigned to each cluster are shown as a histogram beneath the input image in the first column. The number of images assigned to each cluster is proportional to the cluster mixing weights, π. The second row depicts the object labels that were provided for the images in the top row, with the colors showing the spatial extent of the object labels. Notice that the images and labels belonging to each cluster share approximately the same object categories and geometrical configuration. Also, the cluster that best matches the input image tends to have the highest number of retrieval images assigned to it

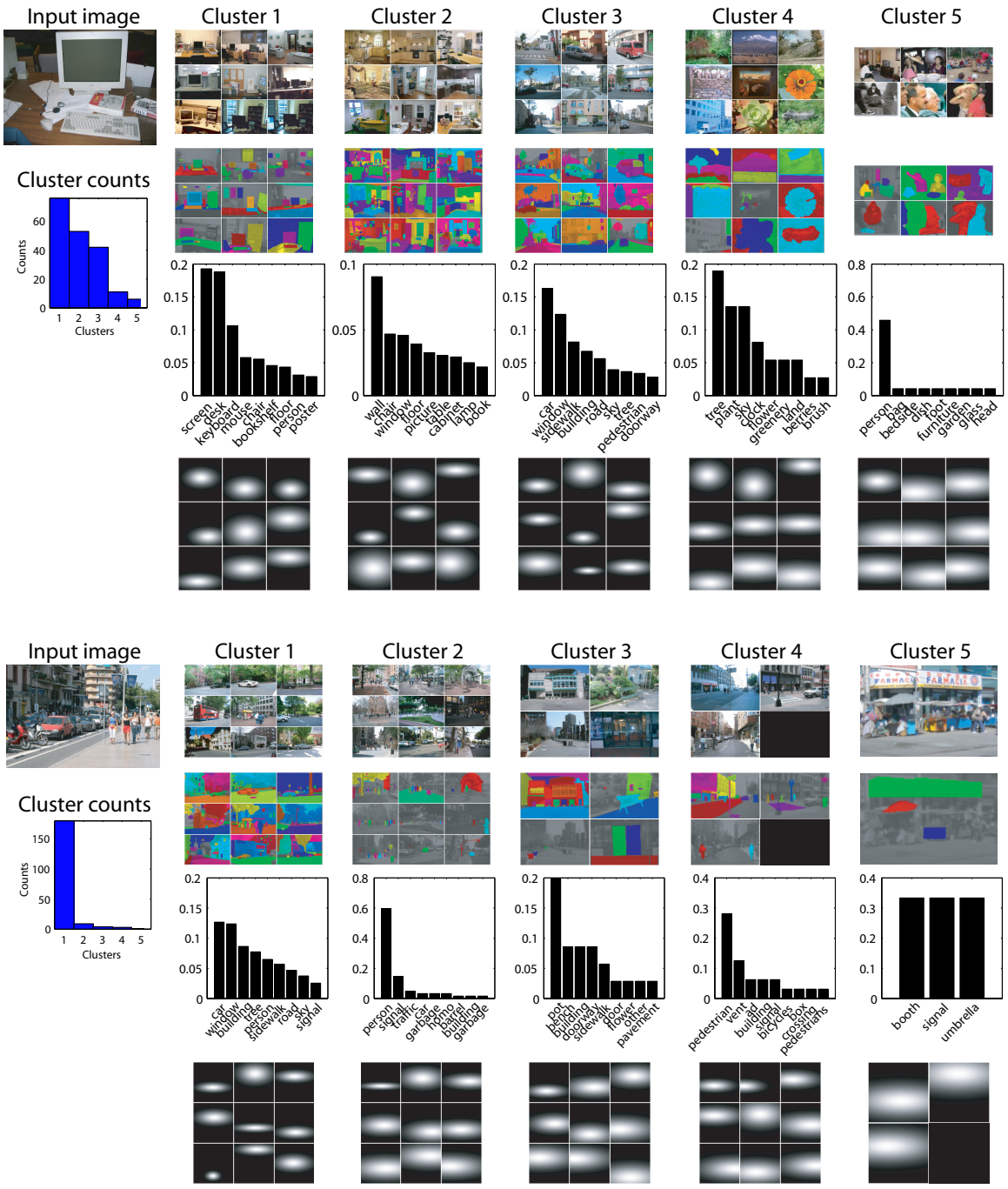


Figure 5-7: Illustration of the scene clustering model (using the retrieval set object labels) for two example input images. *Top-left*: input image. *Adjacent five columns*: visualization of five clusters given by the model. *First row*: montages of example retrieval set images assigned to each cluster. The total number of retrieval set images assigned to each cluster is displayed as a histogram below the input image. *Second row*: object labels (colors show spatial extent) corresponding to the images in the first row. *Third row*: the likelihood of an object category being present in a given cluster (the top nine most likely objects are listed) *Fourth row*: spatial likelihoods for the objects listed in the third row (the montage cells are sorted in raster order). See text for more details.

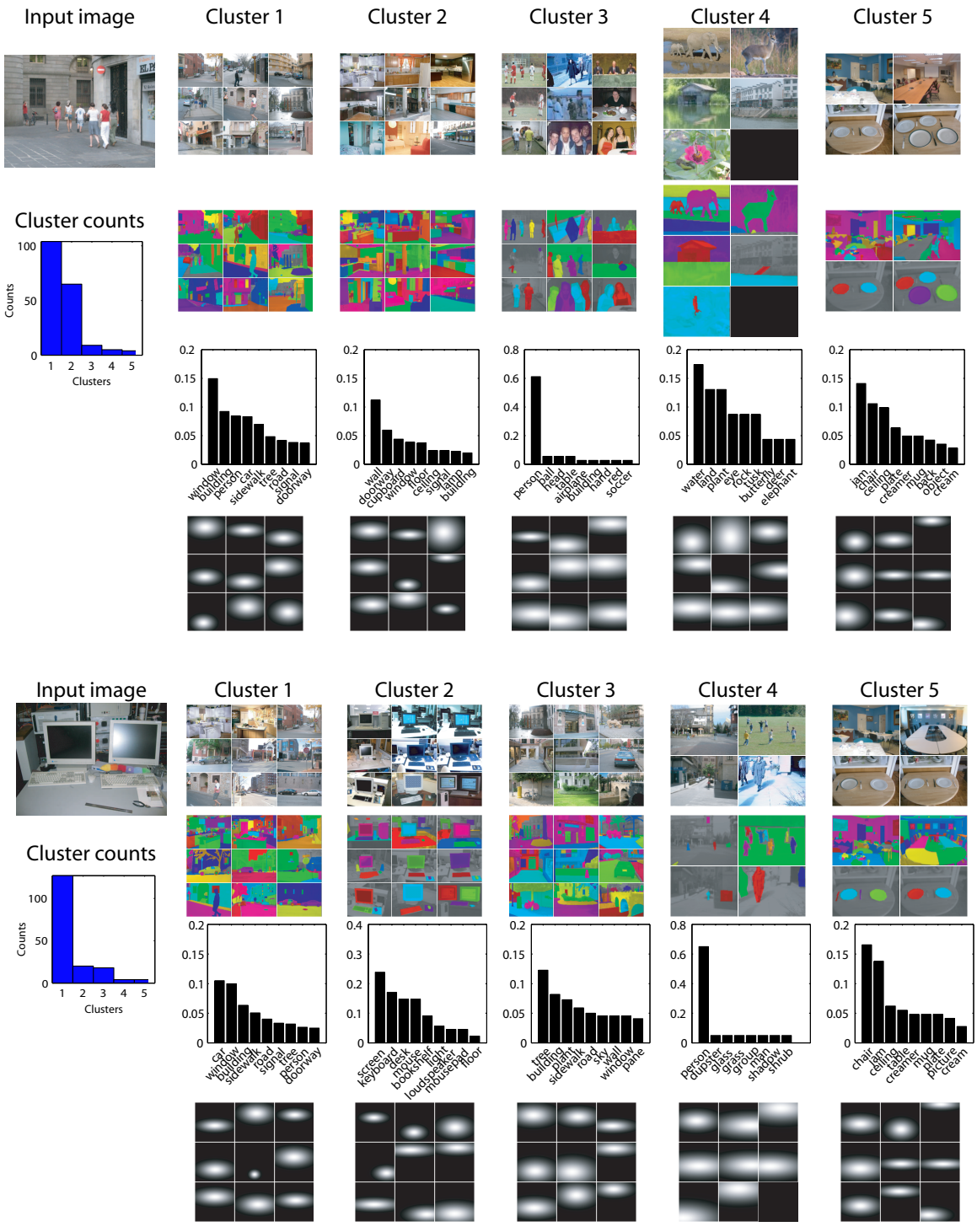


Figure 5-8: Additional scene cluster examples. See Figure 5-7 and text for more details.

(highest likelihood). The third row shows the likelihood of objects that appear in the cluster (the nine objects with highest likelihood are shown). This corresponds to θ in the model. The bottom row depicts the spatial distribution of the object centroid within the cluster. The montage of nine cells correspond to the nine objects listed in the third row, sorted in raster order. The spatial distributions illustrate ϕ. Notice that typically at least one cluster predicts well the objects contained in the input image, in addition to their location, via the object likelihoods and spatial distributions.

To learn θ_k and ϕ_k, we use a Rao-Blackwellized Gibbs sampler to draw samples from the posterior distribution over s_i given the object labels belonging to the set of retrieved images. We ran the Gibbs sampler for 100 iterations. Empirically, we observed relatively fast convergence to a stable solution. Note that improved performance may be achieved with variational inference for Dirichlet Processes [57, 86]. We manually tuned all hyperparameters using a validation set of images, with concentration parameter $\alpha = 100$ and spatial location parameters $\kappa = 0.1$, $\vartheta = 0.5$, $\nu = 3$, and $\Delta = 0.01$ across all bounding box parameters (with the exception of $\Delta = 0.1$ for the horizontal centroid location, which reflects less certainty a priori about the horizontal location of objects). We used a symmetric Dirichlet hyperparameter with $\beta_l = 0.1$ across all object categories l.

For final object detection, we use the learned parameters π, θ, and ϕ to infer $h_{i,j}$. Since s_i and $h_{i,j}$ are latent random variables for the input image, we perform hard EM by marginalizing over $h_{i,j}$ to infer the best cluster s_i^*. We then in turn fix s_i^* and infer $h_{i,j}$, as outlined in Section 5.3.

5.5 Experimental Results

In this section we show qualitative and quantitative results for our model. We use a subset of the LabelMe dataset for our experiments, discarding spurrious and non-labeled images. The dataset is split into training and test sets. The training set has 15691 images and 105034 annotations. The test set has 560 images and 3571

annotations. The test set comprises images of street scenes and indoor office scenes. To avoid overfitting, we used street scene images that were photographed in a different city from the images in the training set. To overcome the diverse object labels provided by users of LabelMe, we used WordNet [30] to resolve synonyms. For object detection, we extracted 3809 bounding boxes per image. For the final detection results, we used non-maximal suppression.

Example object detections from our system are shown in Figure 5-9(b),(d),(e). Notice that our system can find many different objects embedded in different scene type configurations. When mistakes are made, the proposed object location typically makes sense within the scene. In Figure 5-9(c), we compare against a baseline object detector using only appearance information and trained with a linear kernel SVM [96] using SVM-light [45]. Thresholds for both detectors were set to yield a 0.5 false positive rate per image for each object category ($\sim$1.3e-4 false positives per window). Notice that our system produces more detections and rejects objects that do not belong to the scene. In Figure 5-9(e), we show typical failures of the system, which usually occurs when the retrieval set is not correct or an input image is outside of the training set.

In Figure 5-10, we show quantitative results for object detection for a number of object categories. We show ROC curves (plotted on log-log axes) for the local appearance detector, the detector from Section 5.3 (without clustering), and the full system with clustering. We scored detections using the PASCAL VOC 2006 criteria [25], where the outputs are sorted from most confident to least and the ratio of intersection area to union area is computed between an output bounding box and ground-truth bounding box. If the ratio exceeds 0.5, then the output is deemed correct and the ground-truth label is removed. While this scoring criteria is good for some objects, other objects are not well represented by bounding boxes (e.g. buildings and sky).

Notice that the detectors that take into account context typically outperforms the detector using local appearance only. Also, clustering does as well and in some cases outperforms no clustering. Finally, the overall system sometimes performs worse for

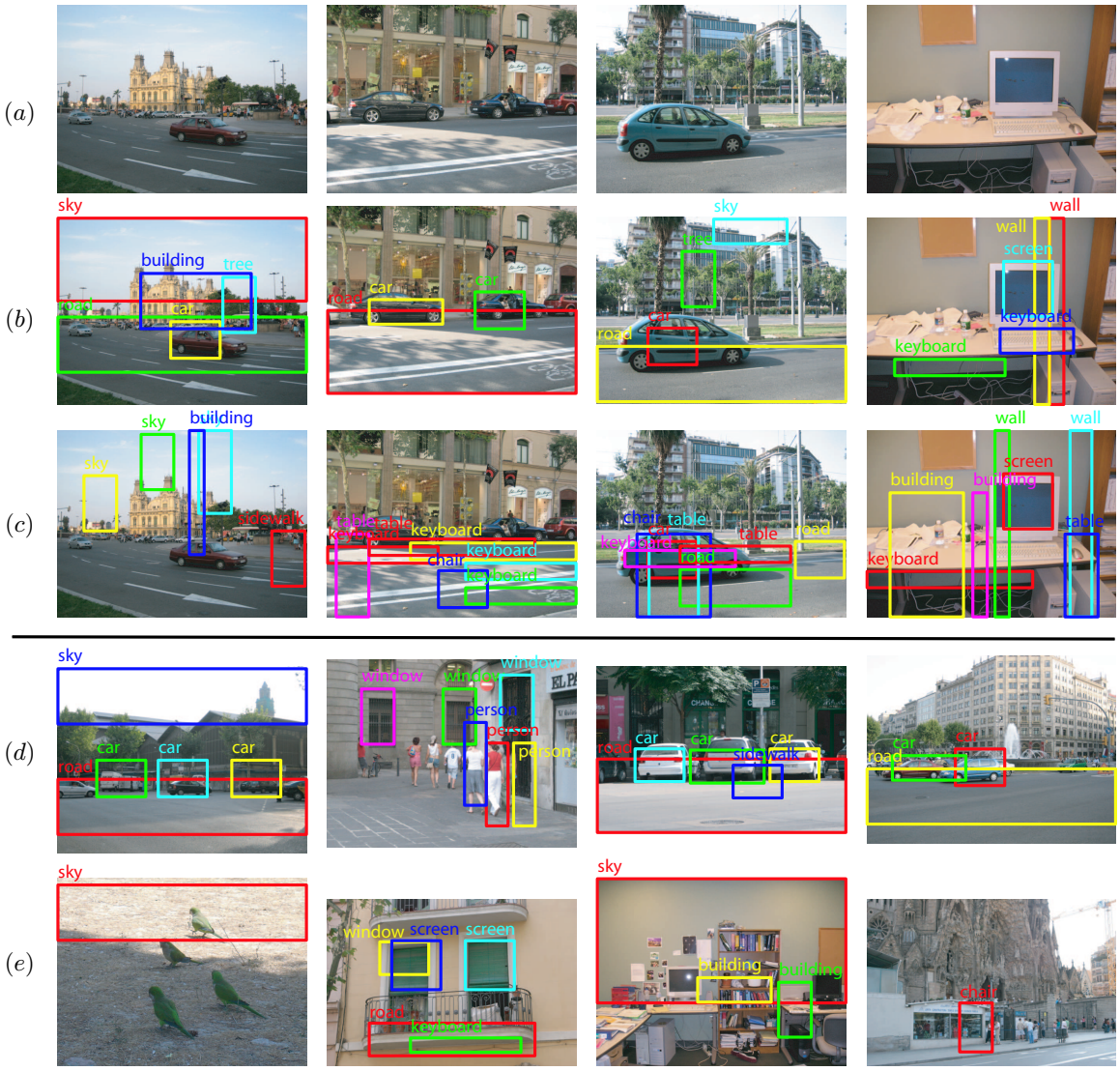


Figure 5-9: (a) Input images. (b) Object detections from our system combining scene alignment with local detection. (c) Object detections using appearance information only with an SVM. Notice that our system detects more objects and rejects out-of-context objects. (d) More outputs from our system. Notice that many different object categories are detected across different scenes. (e) Failure cases for our system. These often occur when the retrieval set is incorrect.

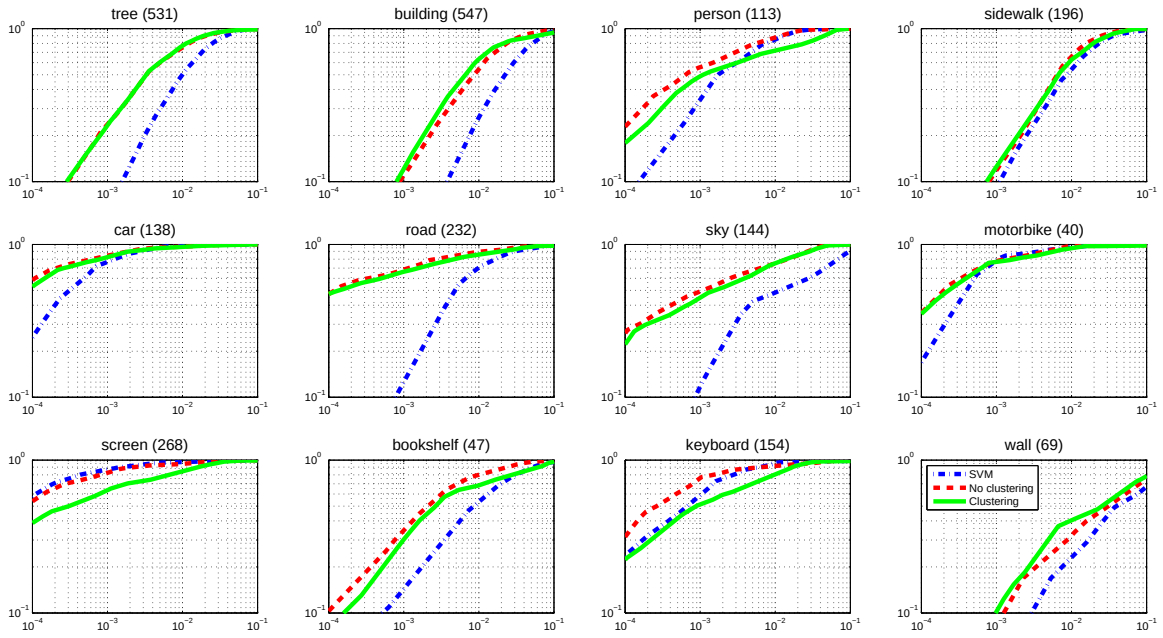


Figure 5-10: Comparison of full system against local appearance only detector (SVM). Detection rate for a number of object categories tested at a fixed false positive per window rate of 2e-04 (0.8 false positives per image per object class). The number of test examples appear in parenthesis next to the category name. We plot performance for a number of classes for the baseline SVM object detector (blue), the detector of Section 5.3 using no clustering (red), and the full system (green). Notice that detectors taking into account context performs better in most cases than using local appearance alone. Also, clustering does as well, and sometimes exceeds no clustering. Notable exceptions are for some indoor object categories. This is due to poor retrieval set matching, which causes a poor context model to be learned.

indoor scenes. This is due to poor retrieval set matching, which causes a poor context model to be learned.

In addition to transferring knowledge about likely objects in the image, we can also transfer other ground-truth information that is associated with the labeled objects. Here, we investigate depth ordering information. Currently, the LabelMe dataset does not contain explicit depth ordering labels. However, the relative depth ordering of the labels can be inferred reliably [70]. For the clustered labels, we count the occlusion relationship between each of pair candidate objects. We assign two candidate objects to different layers if the occlusion rate exceeds 60%. In Figure 5-1, we depict the depth layer relationships between the detected objects.

5.6 Conclusion

We presented a framework for object detection in scenes based on transferring knowledge about objects from a large labeled image database. We have shown that a relatively simple parametric model, trained on images loosely matching the spatial configuration of the input image, is capable of accurately inferring which objects are depicted in the input image along with their location. We showed that we can successfully detect a wide range of objects depicted in a variety of scene types.

Chapter 6

Conclusion

The goal of this work was to pursue directions that may eventually lead us to recognize the many objects that comprise our visual world. The challenging image in Figure 1-1 illustrates the difficult nature of this problem. We observed that many confounding factors may trick a computer vision system. These include, but are not limited to, intra-class object variation, clutter, pose, lighting, dealing with never-before seen objects, scale, and lack of visual experience.

This thesis investigated three issues in object recognition: (i) the role and importance of labeled image databases for recognition, (ii) what can be learned from simply looking at images, and (iii) ways of exploiting large labeled image databases to detect objects embedded in scenes. The results from this investigation suggest the following contributions.

6.1 Contributions

Visual experience is important if we want computer vision systems to be able to recognize objects. For this, we need to have seen examples and know the identity of many objects classes and how they are embedded in the world. In Chapter 2, we built a large database of images and developed an online annotation tool to allow many people to label the location and identity of objects in images. Through this effort, which we called *LabelMe*, we have collected to date almost 200K annotations

from users across the globe. The dataset contains a rich set of objects in many different scene classes. Furthermore, the quality of the annotations are quite good. We showed that this database is one of the largest of its kind by comparing it with existing standard datasets used for object recognition.

As is seen in other applications, large amounts of labeled data dramatically increases the performance of classification. We also saw this in the classification task illustrated in Figure 5-2. A labeled database of images is important for training and validating recognition systems. As a result, the LabelMe database drove the contents and analysis of the other chapters of this thesis.

With such a large database of images and objects, we explored a few simple extensions that provide additional useful information about the labeled objects. First, we employed the use of WordNet [30] to give semantic meaning to the object labels provided by the users of the annotation tool. This is important because users freely type in the object's identity into the tool, thereby resulting in a diverse range of descriptions for objects belonging to the same class. Mapping labels to semantic meanings allows us to overcome the problem of synonyms. Furthermore, the WordNet output provides a hierarchy of relationships between the classes, which allows exploration of the database using superordinate object categories.

Second, we showed how to exploit the co-occurrence of object labels in images to discover object-part relationships. This is important as part-based recognition systems have been shown to be a useful way to recognize objects. Discovering an object's parts also provides an additional hierarchy to the object labels.

Third, we showed a simple heuristic over the label control points that allows the recovery of a depth ordering relationship between the labeled objects in a scene. Objects often occlude each other in scenes, and reasoning about occlusion is a difficult task. The automatic recovery of the occlusion relationships between objects in a scene is an important step in this regard. As a result, this extension augments the object labels and provides an accurate ground truth for depth ordering.

Lastly, we developed a way to simplify the labeling process by bootstrapping from the existing object labels in the database. We trained an object detector and used it

to classify candidate regions as given by an automatic image segmenter. The most confident regions are selected and presented to the labeler. Instead of having to click control points along the boundary of an object, the user now only has to decide whether the automatic object boundary is good enough. This can produce rapid labeling of images for some categories.

While many labeled images are necessary for recognition, it is also important to consider the ability to reason about unfamiliar and unlabeled objects. In Chapter 3, we studied the extreme situation when *no labels* are provided to a recognition system. We drew inspiration from the text understanding literature where semantic models were developed to discover latent topics in text. We employed the visual words representation for images [78], which allowed us to use these semantic models. We primarily investigated the Latent Dirichlet Allocation (LDA) model [15].

As a result, we were able to recover latent topics that corresponded to object categories. We validated our method on a simple dataset where a single object occupied most of the image. We looked at three outputs of the LDA model on the simple dataset: (i) the image features that best describe the discovered categories, (ii) the images that correspond to each topic, and (iii) localization of the objects within the image given the learned model. We saw that the image features best describing the discovered categories corresponded to semantically meaningful object parts. Furthermore, the images depicting a single object category were accurately grouped together given their likely assignments to topics. Finally, objects were reasonably segmented when mixed with some background clutter.

While the object discovery results were encouraging, a problem was encountered when attempting to apply the technique to entire scenes composed of many objects. This was a result of two issues of the image representation: (i) the visual words representation histograms quantized descriptions of local regions in the image, thereby losing the spatial relationship between features extracted from different parts of the image and (ii) the frequent occurrence of polysemy among the visual words, where two different visually semantic parts are described by the same feature. These issues demanded that we consider a way to introduce spatial constraints when discovering

objects.

We considered an augmentation to the visual words representation, called doublets, where we grouped two nearby features together. We observed slightly improved performance over the visual words representation. However, the doublet representation lacked enough spatial support to see large gains in performance.

In Chapter 4, we utilized image segmentation to parcel images into visually coherent pieces. As we saw, image segmentation is not perfect: it often splits objects into separate image fragments or includes multiple objects in the same fragment. However, over a large dataset of images, image segmentation works a lot of the time for many object categories. We reasoned that if this holds true, then we have reduced the problem to the earlier discovery problem, which worked well for images depicting a single object.

We showed improved object discovery for a more complicated set of images extracted from LabelMe. To visualize the results, we used a simple metric to measure the goodness of a particular image fragment of belonging to a discovered topic and displayed the best scoring fragments for each topic. We noticed that the best scoring fragments for a given topic matched well in visual appearance. An important consequence of this visualization is the ability to automatically parse and display the contents of a large image set. We showed this for the satellite image example in Figure 4-7.

After considering the case of learning about objects without supervision, we returned to the situation where we have supervision. In Chapter 5, we developed a system for detecting objects embedded in scenes. While this problem has been addressed before, we proposed a system based on aligning the components of a scene depicted in an input image to a large database of labeled images. The scenes that align best in the image database were used to assist in detecting objects in the input image. This step offers the advantage that we do not have to model various complex interactions that may exist between objects embedded in a scene. We showed results on a held-out subset of LabelMe over a range of objects and scenes. We also compared our output to object detection when we do not consider information about the scene.

The results described in this thesis are encouraging. However, we have only managed to show convincing results for at most hundreds of object categories. Also, we have only analyzed a subset of the visual world: frozen instances in time captured by static images. This compels us to consider the following open issues in recognition: (i) How do we effectively scale recognition to cope with the thousands of object categories that exist and are recognizable by humans? (ii) How can we generalize our results to motion and actions in video? We hope that the techniques and data presented in this thesis will offer a good starting point in addressing these issues.

Bibliography

[1] http://www.maps.google.com.

[2] http://www.robots.ox.ac.uk/ vgg/research/affine/.

[3] Y. Abramson and Y. Freund. Semi-automatic visual learning (seville): a tutorial on active learning for visual object recognition. In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR05), San Diego*, 2005.

[4] Shivani Agarwal, Aatif Awan, and Dan Roth. Learning to detect objects in images via a sparse, part-based representation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(11):1475–1490, 2004.

[5] N. Ahuja and S. Todorovic. Learning the taxonomy and models of categories present in arbitrary images. In *IEEE Intl. Conf. on Computer Vision*, 2007.

[6] K. Barnard and D. A. Forsyth. Learning the semantics of words and pictures. In *IEEE Intl. Conf. on Computer Vision*, volume 2, pages 408–415, 2001.

[7] Kobus Barnard, Pinar Duygulu, Nando de Freitas, David Forsyth, David Blei, and Michael I. Jordan. Matching words and pictures. *J. of Machine Learning Research*, 3:1107–1135, 2003.

[8] A. R. Barron and T. M. Cover. Minimum complexity density estimation. *IEEE Trans. on Information Therory*, 4:1034–1054, 1991.

[9] A. Berg, T. Berg, and J. Malik. Shape matching and object recognition using low distortion correspondence. In *CVPR*, volume 1, pages 26–33, June 2005.

[10] T. Berg, A. Berg, J. Edwards, R. White, Y. W. Teh, E. Learned-Miller, and D. Forsyth. Names and faces in the news. In *CVPR*, pages 848–854, 2004.

[11] T. L. Berg and D. A. Forsyth. Animals on the web. In *CVPR*, volume 2, pages 1463–1470, 2006.

[12] I. Biederman. Recognition by components: a theory of human image interpretation. *Pyschological review*, 94:115–147, 1987.

[13] S. Bileschi. CBCL streetscenes. Technical report, MIT CBCL, 2006. The CBCL-Streetscenes dataset can be downloaded at `http://cbcl.mit.edu/software-datasets`.

[14] D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[15] D. M. Blei, T. Griffiths, M. I. Jordan, and J. Tenenbaum. Hierarchical topic models and the nested chinese restaurant process. In *Advances in Neural Info. Proc. Systems*, 2003.

[16] E. Borenstein, E. Sharon, and S. Ullman. Combining top-down and bottom-up segmentation. In *CVPR*, 2004.

[17] J. Burianek, A. Ahmadyfard, and J. Kittler. Soil-47, the Surrey object image library. http://www.ee.surrey.ac.uk/Research/VSSP/demos/colour/soil47/, 2000.

[18] Owen Carmichael and Martial Hebert. Word: Wiry object recognition database. www.cs.cmu.edu/~owenc/word.htm, January 2004. Carnegie Mellon University.

[19] D. Comaniciu and P. Meer. Robust analysis of feature spaces: Color image segmentation. In *CVPR*, 1997.

[20] G. Csurka, C. Dance, C. Bray, L. Fan, and J. Willamowski. Visual categorization with bags of keypoints. In *ECCV workshop on statistical learning in computer vision*, 2004.

[21] B. de Finetti. *Theory of Probability, Vol 1-2*. John Wiley and Sons Ltd., 1990.

[22] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407, 1990.

[23] P. Duygulu, K. Barnard, N. de Freitas, D. Forsyth, and M. Jordan. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *ECCV*, 2002.

[24] M. Everingham, A. Zisserman, C. Williams, L. Van Gool, M. Allan, C. Bishop, O. Chapelle, N. Dalal, T. Deselaers, G. Dorko, S. Duffner, J. Eichhorn, J. Farquhar, M. Fritz, C. Garcia, T. Griffiths, F. Jurie, D. Keysers, M. Koskela, J. Laaksonen, D. Larlus, B. Leibe, H. Meng, H. Ney, B. Schiele, C. Schmid, E. Seemann, J. Shawe-Taylor, A. Storkey, S. Szedmak, B. Triggs, I. Ulusoy, V. Viitaniemi, and J. Zhang. The 2005 pascal visual object classes challenge. In *First PASCAL Challenges Workshop*. Springer-Verlag, 2005.

[25] M. Everingham, A. Zisserman, C.K.I. Williams, and L. Van Gool. The pascal visual object classes challenge 2006 (voc 2006) results. Technical report, September 2006. The PASCAL2006 dataset can be downloaded at `http://www.pascal-network.org/challenges/VOC/voc2006/`.

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

[27] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories. In *IEEE. CVPR 2004, Workshop on Generative-Model Based Vision*, 2004.

[28] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE Trans. Pattern Recognition and Machine Intelligence*, In press. The Caltech 101 dataset can be downloaded at `http://www.vision.caltech.edu/Image_Datasets/Caltech101/Caltech101.html`.

[29] L. Fei-Fei and P. Perona. A Bayesian hierarchical model for learning natural scene categories. In *CVPR*, 2005.

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

[31] P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. *Intl. J. Computer Vision*, 61(1), 2005.

[32] R. Fergus, L. Fei-Fei, P. Perona, and A. Zisserman. Learning object categories from google's image search. In *IEEE Intl. Conf. on Computer Vision*, October 2005.

[33] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, 2003.

[34] A. W. Fitzgibbon and A. Zisserman. On affine invariant clustering and automatic cast listing in movies. volume 3, pages 304–320, 2002.

[35] G. Griffin, A.D. Holub, and P. Perona. The Caltech-256. Technical report, California Institute of Technology, 2006.

[36] T. Griffiths and M. Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101:5228–5235, 2004.

[37] James Hays and Alexei Efros. Scene completion using millions of photographs. In *"SIGGRAPH"*, 2007.

[38] B. Heisele, T. Serre, S. Mukherjee, and T. Poggio. Feature reduction and hierarchy of classifiers for fast object detection in video images. In *CVPR*, 2001.

[39] T. Hofmann. Probabilistic latent semantic indexing. In *SIGIR*, 1999.

[40] T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 43:177–196, 2001.

[41] D. Hoiem, A. Efros, and M. Hebert. Putting objects in perspective. In *CVPR*, 2006.

[42] D. Hoiem, A.A. Efros, and M. Hebert. Geometric context from a single image. In *IEEE Intl. Conf. on Computer Vision*, 2005.

[43] N. Ide and J. Vronis. Introduction to the special issue on word sense disambiguation: the state of the art. *Computational Linguistics*, 24(1):1–40, 1998.

[44] H. Ishwaran and M. Zarepour. Exact and approximate sum-representations for the dirichlet process. *Canadian Journal of Statistics*, 30:269–283, 2002.

[45] T. Joachims. *Making large-Scale SVM Learning Practical. Advances in Kernel Methods - Support Vector Learning*. MIT-Press, 1999.

[46] M. I. Jordan. Graphical models. *Statistical Science (Special Issue on Bayesian Statistics)*, 19:140–155, 2004.

[47] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *cvpr*, pages 2169–2178, 2006.

[48] Yann LeCun, Fu-Jie Huang, and Leon Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Proceedings of CVPR'04*. IEEE Press, 2004.

[49] B. Leibe. *Interleaved object categorization and segmentation*. PhD thesis, 2005.

[50] B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with an implicit shape model. In *ECCV*, 2004.

[51] B. Leibe and B. Schiele. Analyzing appearance and contour based methods for object categorization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'03)*, Madison, WI, June 2003.

[52] Y. Li and L. G. Shapiro. Consistent line clusters for building recognition in cbir. In *Proceedings of the International Conference on Pattern Recognition*, 2002.

[53] David G. Lowe. Object recognition from local scale-invariant features. In *Proc. of the International Conference on Computer Vision ICCV, Corfu*, pages 1150–1157, 1999.

[54] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Intl. J. Computer Vision*, 60(2):91–110, 2004.

[55] D. Martin, C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using brightness and text ure. In *Advances in Neural Info. Proc. Systems*, 2002.

[56] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regi ons. In *British Machine Vision Conference*, pages 384–393, 2002.

[57] J. McAuliffe, D. Blei, and M. Jordan. Nonparametric empirical bayes for the Dirichlet process mixture model. *Statistics and Computing*, 16:5–14, 2006.

[58] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. Springer-Verlag, 2002.

[59] T. Minka and J. Lafferty. Expectation propagation for the generative aspect model. In *Proc. of the Conf. on Uncertainty in AI*, 2002.

[60] R. M. Neal. Density modeling and clustering using Dirichlet diffusion trees. *In Bayesian Statistics*, 7:619–629, 2003.

[61] A. Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *Intl. J. Computer Vision*, 42(3):145–175, 2001.

[62] A. Opelt, M. Fussenegger, A. Pinz, and P. Auer. Weak hypotheses and boosting for generic object detection and recognition. In *ECCV*, 2004.

[63] A. Opelt, A. Pinz, M.Fussenegger, and P.Auer. Generic object recognition with boosting. *IEEE Transactions on Pattern Recognition and Machine Intelligence (PAMI)*, 28(3), 2006.

[64] A. Opelt, A. Pinz, and A. Zisserman. A boundary-fragment-model for object detection. In *ECCV*, 2006.

[65] P. Quelhas, F. Monay, J. M. Odobez, D. Gatica-Perez, T. Tuytelaars, and L. Van Gool. Modeling scenes with local descriptors and latent aspects. In *IEEE Intl. Conf. on Computer Vision*, 2005.

[66] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, and S. Belongie. Objects in context. In *IEEE Intl. Conf. on Computer Vision*, 2007.

[67] B. C. Russell, A. A. Efros, J. Sivic, W. T. Freeman, and A. Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *CVPR*, 2006.

[68] B. C. Russell, A. Torralba, C. Liu, R. Fergus, and W. T. Freeman. Object recognition by scene alignment. In *To appear in Advances in Neural Information Processing Systems*, 2007.

[69] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. LabelMe: a database and web-based tool for image annotation. *To appear in a special issue of the International Journal of Computer Vision.*

[70] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. Labelme: a database and web-based tool for image annotation. Technical Report AIM-2005-025, MIT AI Lab Memo, September, 2005.

[71] F. Schaffalitzky and A. Zisserman. Multi-view matching for unordered image sets, or "How do I organize my holiday snaps?". volume 1, pages 414–431. Springer-Verlag, 2002.

[72] Henry Schneiderman and Takeo Kanade. A statistical model for 3D object detection applied to faces and cars. In *CVPR*, 2000.

[73] Flickr Photo Sharing Service. http://www.flickr.com.

[74] E. Sharon, A. Brandt, and R. Basri. Segmentation and boundary detection using multiscale intensity measurements. In *CVPR*, pages I:469–476, 2001.

[75] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2000.

[76] J. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman. Discovering object categories in image collections. Technical Report AIM-2005-05, MIT AI Lab, 2005.

[77] J. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman. Discovering objects and their location in images. In *IEEE Intl. Conf. on Computer Vision*, 2005.

[78] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *IEEE Intl. Conf. on Computer Vision*, 2003.

[79] J. Sivic and A. Zisserman. Video data mining using configurations of viewpoint invariant regions. June 2004.

[80] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3d. *ACM Transactions on Graphics*, 25(3):137–154, 2006.

[81] D. G. Stork. The open mind initiative. *IEEE Intelligent Systems and Their Applications*, 14(3):19–20, 1999.

[82] E. Sudderth, A. Torralba, W. T. Freeman, and W. Willsky. Describing visual scenes using transformed dirichlet processes. In *Advances in Neural Info. Proc. Systems*, 2005.

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

[84] M. J. Swain and D. H. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1), 1991.

[85] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 2006.

[86] Y. W. Teh, D. Newman, and Welling M. A collapsed variational bayesian inference algorithm for latent dirichlet allocation. In *Advances in Neural Info. Proc. Systems*, 2006.

[87] S. Todorovic and N. Ahuja. Extracting subimages of an unknown category from a set of images. In *CVPR*, 2006.

[88] L. Tolstoy. *Anna Karenina*. 1877.

[89] A. Torralba. Contextual priming for object detection. *Intl. J. Computer Vision*, 53(2):153–167, 2003.

[90] A. Torralba, R. Fergus, and W.T. Freeman. Tiny images. Technical Report AIM-2005-025, MIT AI Lab Memo, September, 2005.

[91] A. Torralba, K. Murphy, and W. Freeman. Contextual models for object detection using boosted random fields. In *Advances in Neural Info. Proc. Systems*, 2004.

[92] A. Torralba, K. Murphy, and W. Freeman. Sharing features: efficient boosting procedures for multiclass object detection. In *CVPR*, 2004.

[93] A. Torralba, K. Murphy, W. Freeman, and M. Rubin. Context-based vision system for place and object recognition. In *Intl. Conf. Computer Vision*, 2003.

[94] Z.W. Tu and S.C. Zhu. Image Segmentation by Data-Driven Markov Chain Monte Carlo. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(5):657–673, 2002.

[95] M. Turk and A. Pentland. Eigenfaces for recognition. *J. of Cognitive Neuroscience*, 3(1):71–86, 1991.

[96] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, 1999.

[97] T. Vetter, M. Jones, and T. Poggio. A bootstrapping algorithm for learning linear models of object classes. In *CVPR*, 1997.

[98] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple classifiers. In *CVPR*, 2001.

[99] L. von Ahn and L. Dabbish. Labeling images with a computer game. In *Proc. SIGCHI conference on Human factors in computing systems*, 2004.

[100] L. von Ahn, R. Liu, and M. Blum. Peekaboom: A game for locating objects in images. In *In ACM CHI*, 2006.

[101] M. Weber, M. Welling, and P. Perona. Towards automatic discovery of object categories. In *CVPR*, pages 101–109, 2000.

[102] J. Winn, A. Criminisi, and T. Minka. Object categorization by learned universal visual dictionary. In *IEEE Intl. Conf. on Computer Vision*, 2005. The MSRC dataset can be downloaded at `http://research.microsoft.com/vision/cambridge/recognition/default.htm`.

[103] J. Winn and N. Jojic. Locus: Learning object classes with unsupervised segmentation. In *IEEE Intl. Conf. on Computer Vision*, 2005.