STUDIES IN USING IMAGE SEGMENTATION TO IMPROVE OBJECT RECOGNITION

Caroline Rebecca Pantofaru

CMU-RI-TR-08-23

Submitted in partial fulfilment of
the requirements for the degree of
Doctor of Philosophy

Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213

May 2008

Thesis Committee:
Martial Hebert, Chair
Alexei A. Efros
Rahul Sukthankar
Cordelia Schmid, INRIA Rhône-Alpes

ABSTRACT

RECOGNIZING object classes is a central problem in computer vision, and recently there has been renewed interest in also precisely localizing objects with pixel-accurate masks. Since classes of deformable objects can take a very large number of shapes in any given image, a requirement for recognizing and generating masks for such objects is a method for reducing the number of pixel sets which need to be examined. One method for proposing accurate spatial support for objects and features is data-driven pixel grouping through unsupervised image segmentation. The goals of this thesis are to define and address the issues associated with incorporating image segmentation into an object recognition framework.

The first part of this thesis examines the nature of image segmentation and the implications for an object recognition system. We develop a scheme for comparing and evaluating image segmentation algorithms which includes the definition of criteria that an algorithm must satisfy to be a useful black box, experiments for evaluating these criteria, and a measure of automatic segmentation correctness versus human image labeling. This evaluation scheme is used to perform experiments with popular segmentation algorithms, the results of which motivate our work in the remainder of this thesis.

The second part of this thesis explores approaches to incorporating the regions generated by unsupervised image segmentation into an object recognition framework. Influenced by our experiments with segmentation, we propose principled methods for describing such regions. Given the instability inherent in image segmentation, we experiment with increasing robustness by integrating the information from multiple segmentations. Finally, we examine the possibility of learning explicit spatial relationships between regions. The efficacy of these techniques is demonstrated on a number of challenging data sets.

ACKNOWLEDGEMENTS

The PhD process is intellectually stimulating and emotionally draining, and as such requires the support of many people. First, I'd like to thank my committee. To my advisor Martial Hebert, thank you for your patience through the many ups and downs over the years. To Cordelia Schmid, thank you for your thoughtful input into my work, and for allowing me to spend time with your group at INRIA Rhône-Alpes. To Alyosha Efros, thank you for your insights into the broader picture. To Rahul Sukthankar, thank you for serving on my committee and for all of your support.

To all of my friends in Pittsburgh, thank you for making me laugh, for all of the research discussions, for listening to me, for letting me listen to you, for many coffee breaks, and for keeping me relatively sane.

To all of my friends outside of Pittsburgh, thank you for always being there when I came out of hiding.

To my family, thank you for being supportive and incredibly understanding despite my long absences. Sharing a meal with you, I know that I'm home.

To Nick, thank you most of all. The strength of our relationship amazes me every day. Thank you for being a true partner, whether near or far.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1. Motivation

The human eye perceives rays of light and the human brain perceives people, cars, buildings, and all the objects that make up the world. A camera perceives rays of light, and a computer perceives dots of color. This disappointing disconnect between the human and computer visual experiences has long frustrated the field of computer vision. How can colored dots be converted into a cup of coffee? How can we program computers to detect and recognize objects?

Reflecting on the problem, it quickly becomes clear that any single colored dot possesses very little information about the larger object to which it belongs. A black dot does not define a cheetah, a larger section of the visual field is needed to see the repeating patches of black on golden fur which are so distinctive. This begs the question: what part of an image should be examined to recognize an object? The obvious answer is: whichever pixels compose the object. Unfortunately, objects take on an enormous number of shapes, and even a single object like our cheetah can drastically change its shape. Searching through every possible combination of pixels is intractable, we need a way to limit the number of possibilities.

To overcome this issue, much of the work in object recognition has resigned itself to using subsets of pixels in a fixed shape. Image classification considers only one subset, the entire image, and tries to classify whether an object exists anywhere in a scene. Examples of image classification techniques are [38,39,65,66, 126] among others. There are also many patch-based approaches, which consider pixel information in a fixed-shape subset of the image defined a priori. Boxes or rectangles are a natural shape choice given that pixels are usually arranged in a grid pattern [2,31,121]. By placing these shapes in different positions in the image and with different variations of their height, width, orientation and other parameters, the number of pixel subsets to examine becomes tractable. The choice of shapes, however, is largely a matter of convenience. While a square may work well for defining a chessboard, it does a poor job of defining which pixels compose our cheetah.

Applications such as digital image editing and robotics demand more precision than a box around an object, they need to know exactly which pixels belong to the object. It would be unacceptable to have a robot reach for the handle of a mug but instead soak its hand in the coffee. Precise object masks are also beneficial for recognizing the identity of an object [72]. Consider the objects in Figure 1.1. A box placed around these objects contains more non-object pixels than object pixels. If we accumulate information over the entire box, the non-object information will certainly dominate, making object recognition difficult. If an image editing program changed the color of the cheetah by making the entire box blue, it would not sell very well. Both the process and applications benefit from accurate object location information. This motivates our goal of recognizing objects and also accurately denoting their pixel masks, termed *object recognition and object segmentation*.

2. Problem description

One possible approach to properly identifying the pixels which belong in an object mask is to learn the shape of the object. There exist a number of top-down methods which attempt to model the outline or silhouette of an object, such as [9, 73,81]. Additional approaches subdivide an object into a number of rigid parts and then model each part's shape and their relative configuration [12,41,67,82]. These methods show promise for rigid objects, or objects with a small number of rigid

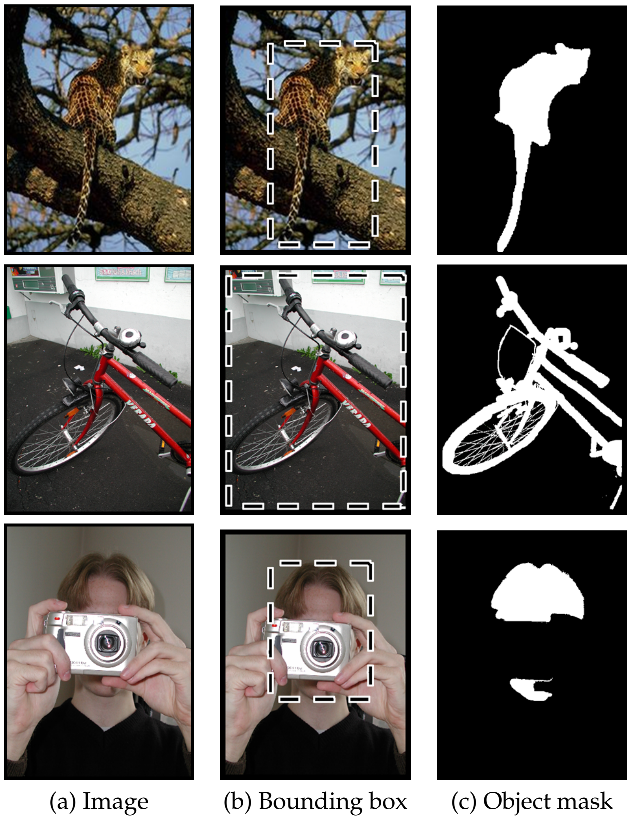
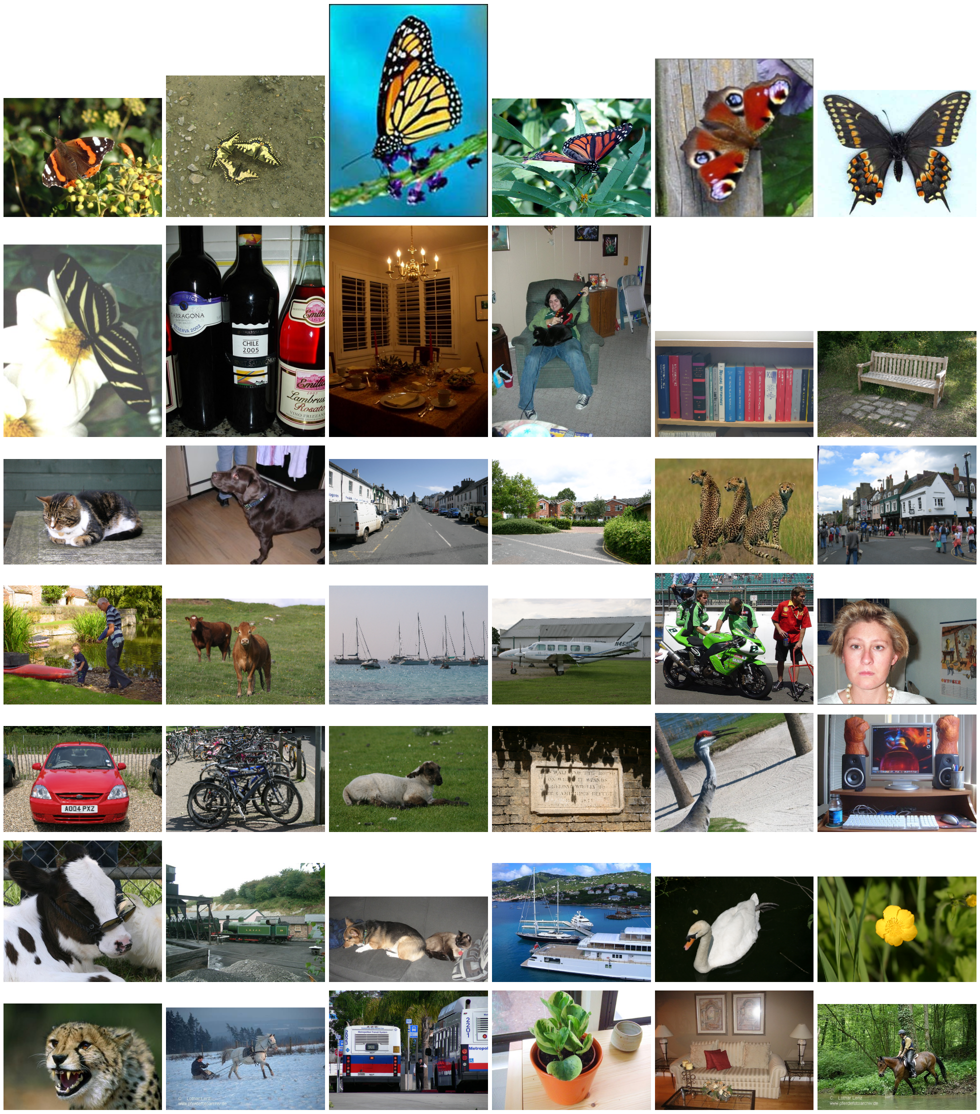


FIGURE 1.1. Examples of (a) images , (b) bounding boxes surrounding objects of interest, and (c) pixel-accuracy object masks.

parts, such as a fire hydrant, or even the side view of a particular breed of horse running.

Let us consider, however, all of the objects in Figure 1.2. These objects range from rigid objects such as a car, to extremely deformable objects such as the cheetah with its flexible body and tail in a variety of positions, all the way to objects whose shapes not only change but in fact are uninformative, such as water. These objects have a very large set of shapes they can take, so top-down object knowledge does not sufficiently limit the possible sets of pixels which might make up their object masks. We require a data-driven, or bottom-up approach, which can group together some of the pixels and so reduce the size of the configuration space.

For these reasons, there is a growing movement toward using unsupervised image segmentation to provide preliminary pixel grouping [16, 51–53, 68, 86, 88, 91, 93, 94, 111]. Unsupervised image segmentation is a broad term which refers to any



Object classes:
Bicycle, Bird, Boat, Body, Book, Bottle, Building, Bus, 7 Butterfly species, Car, Chair,
Cow, Dining table, Dog, Face, Flower, Grass, Horse, Motorcycle, Person, Plane, Potted plant,
Road, Sheep, Sign, Sky, Sofa, Spotted cat, Train, Tree, TV/Monitor, Water

FIGURE 1.2. Examples of the objects we will model throughout this thesis.

method of grouping together pixels which are similar in a feature space. Some of the most common feature spaces are image position, luminance value, color, or the texture in the pixel's vicinity. Throughout this document we will use some of the more popular segmentation methods, described in Chapter 2, but many other algorithms could be substituted with equal success.

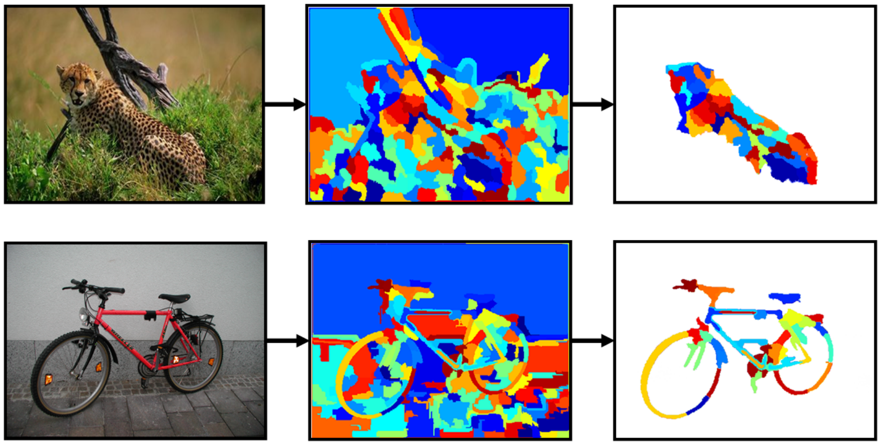


FIGURE 1.3. Illustration of selecting and combining segmentation-generated regions to form an object mask.

At the core of our approach is the belief that data-driven bottom-up pixel grouping can be used to define image regions which provide good spatial support for computing image features, and can be used to define the precise location of an object. A simple version of this concept is illustrated in Figure 1.3, where a union of segmentation regions produces an accurate object mask.

The goals of this thesis are to *define and explore the issues related to, and propose new methods for, combining bottom-up image segmentation and top-down object information to recognize object classes and produce pixel-accurate masks of their locations.*

3. Approach and document outline

To use image regions generated by unsupervised image segmentation for object recognition, we must first understand the relationship between regions and objects. The most straight-forward method for using image segmentation is as a 'black box', grouping the pixels in an image and assuming that each region corresponds to an object that needs to be recognized [94]. In Chapter 3, we propose a set of criteria, experiments, and a quantitative measure of segmentation 'correctness' (in joint work with R. Unnikrishnan) that can be used to determine if a segmentation algorithm would be an appropriate black box, as well as comparing the efficacy of multiple algorithms. This work was originally presented in [87,116,117].

As a result of experiments conducted using the segmentation algorithms in Chapter 2, we determine that in fact bottom-up image segmentation cannot be used in such a simplistic manner. Our experiments show that segmentation regions rarely correspond perfectly to objects. Instead, they may denote a portion of an object, or they may include both object and non-object pixels. These relationships between segmentation regions and objects, termed over- or under-segmentation, can vary with the algorithm used to create the segmentation, with the algorithm's parameters, with the image used, in fact they can even vary within an image. These discoveries motivate our approach to using image segmentation for object recognition.

Throughout the remainder of this document, we will present a number of object recognition and object segmentation experiments. For ease of reading, we pause in Chapter 4 to describe our experiment methodology and data sets.

With our evaluation methodology in place, we can proceed to discuss our approach to incorporating image segmentation into an object recognition system. At the highest level, our approach is to divide an image into regions using bottom-up unsupervised segmentation, describe each region, and then use top-down object knowledge to select and combine the regions to form an object mask. We begin in Chapter 5 by discussing the basis of any object recognition system: image description. Creating an image description involves converting image pixels into features that can be assigned an object label, and in our case the description will be based on the regions created by image segmentation. We present two approaches for representing the image structure within a region. The first is a more traditional representation of repetitive texture [71], which is excellent for discovering objects with distinctive patterns such as cheetah bodies. However, not all objects are composed of repetitive textures. Also, our experiments showed that segmentation-generated regions do not necessarily encompass entire objects, and their extend is subject to the whims of the particular segmentation algorithm and parameters used. Thus, our second representation is the novel Region-based Context Feature (RCF), which considers distinctive image structure in and *around* a region. By combining the structures around a region based on their scale, not on the region's area, the RCF provides a principled and more stable approach for choosing which structures are relevant to a region.

Our bottom-up image segmentation process has proposed groups of pixels which should possess the same object label, and has represented the information in each region in a more practical form. We now require an injection of top-down object information to label which regions and features belong to which object class. Much of the work in generating object masks has relied on fully supervised training data which contains human-drawn object masks. Such training data is extremely expensive to obtain, however, and so the process cannot be scaled to larger object and image sets. Completely unsupervised training data without any object labels can be obtained easily and used for object discovery, but without human input there is no guarantee that the discovered objects will be of interest. In Chapter 6, we learn to classify region features using a training data set of images which are weakly labeled with the objects they contain, but no information is given about the object locations or object masks. Detailed experiments are performed using our region representations and classifier on multiple data sets, testing the relative merits of each representation on different data sets and showing that our approach produces state-of-the-art results.

Our object recognition and segmentation framework thus far has implicitly relied on the regions generated by image segmentation to be 'useful', large enough to compute higher-order statistics of image structure, but small enough to fall within an object's boundaries. As we discover in our image segmentation experiments in Chapter 3, this is an unrealistic expectation for a single image segmentation. In Chapter 7, we join [6, 11, 52, 72, 86, 94, 111] in recommending the use of *multiple* bottom-up segmentations of each image. Unlike [94], however, we believe there is information in all of the segmentations generated by our multiple algorithms and parameters, and we utilize them all in concert to reinforce object recognition and object segmentation. Large regions can provide contextual indications of object presence, regions which lie within object boundaries give object-specific information, and small regions better capture unique local structures. In addition, the correct object boundaries are more likely to be a subset of the union of region

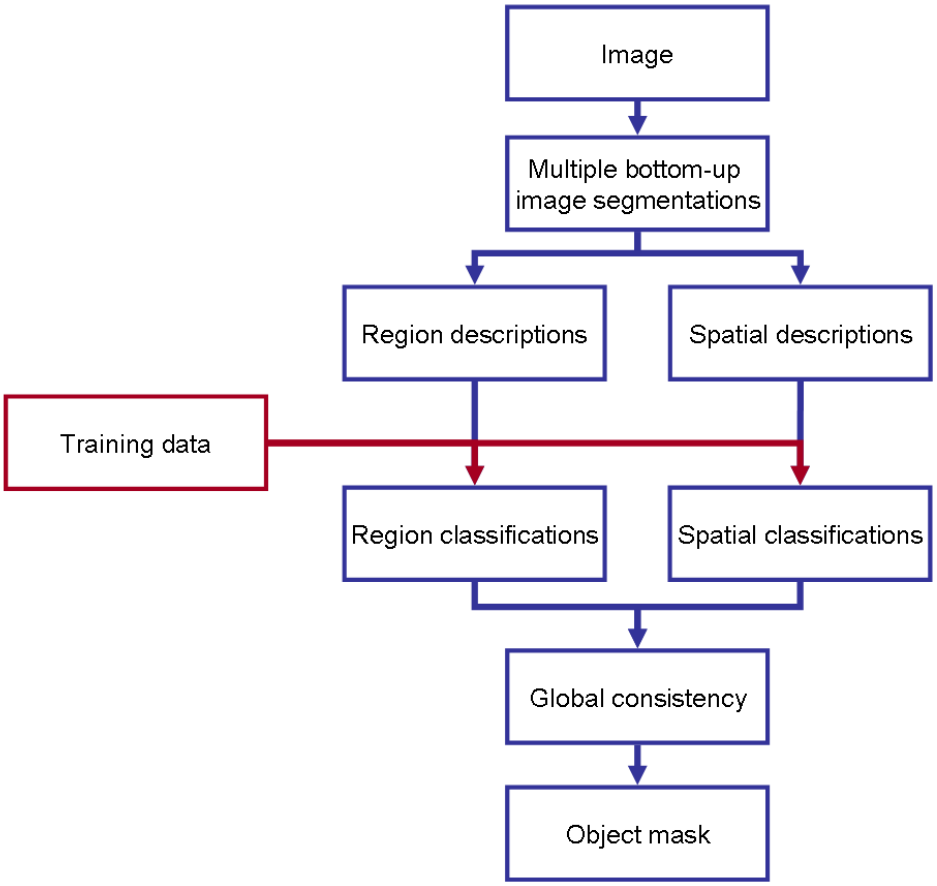


FIGURE 1.4. Overview of our algorithm.

boundaries in multiple segmentations rather than in one segmentation. We present experimental evidence of the benefit of using multiple segmentations within our object recognition framework.

Given the heterogeneity of the parts of certain objects, such as a person's shirt and pants, it is possible that even multiple image segmentations may not generate any region which crosses such part boundaries. The Region-based Context Features capture some information from outside of a region, however that information is still local. Chapter 8 considers the potential for improvement by capturing spatial information and enforcing spatial consistency through explicitly modeling the spatial arrangement of an object's constituent regions. As we discussed earlier, however, many of the objects we wish to model are in fact quite deformable, so a shape-based or part-based method will be too rigid. Instead, we choose to model the more flexible constraint of pairwise region adjacency to model spatial relationships, and utilize a random field formulation to enforce spatial consistency. Our publications on region description, classification methods and spatial consistency include [86] and [88].

In summary, this dissertation discovers and addresses a set of challenges related to incorporating image unsupervised segmentation into an object recognition and object segmentation framework. By performing a set of rigorous experiments regarding the relationship between segmentation regions and object masks, we motivate our approach. To address the weaknesses in image segmentation, we create features which better represent regions, utilize multiple segmentations per image, and consider enforcing spatial consistency between region labels. An algorithm overview is given in Figure 1.4. Through these methods we obtain state-of-the-art performance on a number of image data sets. In Chapter 9, we look at the bigger picture by considering some of the applications that could benefit from an object recognition and segmentation system such as ours.

4. Contributions

The key contributions of this dissertation include:

- A framework for quantitatively evaluating and comparing segmentation algorithms, including:
  - a quantitative measure of segmentation correctness (in joint work with R. Unnikrishnan),
  - the definition of criteria for a useful black-box segmentation algorithm and,
  - a set of experiments to measure those criteria.
- Extensive experiments using the above evaluation scheme which elucidate the relationship between segmentation-generated regions and object masks, and motivate our approach to using regions.

- A new region descriptor that adapts to the deformable, inconsistent shape of segmentation-generated regions to facilitate object recognition and segmentation.
- A framework and thorough experiments for recognizing objects and denoting their pixel masks, trained using only a weakly supervised image set.
- An approach to incorporating multiple segmentations into the recognition framework, thereby addressing the issue of image segmentation variability.
- A method for describing the spatial relationships between regions and enforcing spatial consistency in the final image labeling to increase robustness and combine heterogeneous object parts.

CHAPTER 2

IMAGE SEGMENTATION ALGORITHMS

THROUGHOUT this thesis, a number of unsupervised image segmentation algorithms will be employed. Although the recognition framework we will introduce is in fact agnostic to the segmentation algorithms used, we must instantiate our experiments with a set of algorithms. Thus, we begin by introducing the five popular image segmentation algorithms used in this work: mean shift-based segmentation algorithm [25], an efficient graph-based segmentation algorithm [36], a hybrid of the previous two, normalized cuts segmentation using boundaries [44,76,104], and expectation maximization [27]. Each of the algorithms has different strengths and weaknesses which we will briefly describe here and then expand upon in Chapter 3.

1. Mean Shift Segmentation

The mean shift based segmentation technique was introduced in [25] and is one of many techniques under the heading of "feature space analysis". The technique is comprised of two basic steps: a mean shift filtering of the original image data (in feature space), and a subsequent clustering of the filtered data points.

The filtering step of the mean shift segmentation algorithm consists of analyzing the probability density function underlying the image data in feature space. In the original implementation, the feature space consists of the (x, y) image location of each pixel and the (smoothed) pixel color in L*u*v* space (L^*, u^*, v^*). The modes of the pdf underlying the data in this feature space will correspond to the locations with highest data density, and data points close to these modes can be clustered together to form a segmentation. The mean shift filtering step consists of finding these modes through the iterative use of kernel density estimation of the gradient of the pdf, and associating with them any points in their basin of attraction. Details may be found in [25].

In the implementations of our object recognition algorithms, we extend the mean shift algorithm to also include texture as a feature. We compute texture using the algorithm from the Berkeley segmentation database website [71, 77] to generate texton histograms; the texton at each pixel is a vector of responses to 24 filters quantized into 30 textons, and the texton histogram centered at a pixel is an accumulation of the textons in a 19x19 pixel window. The low dimensionality of our texton histograms allows for generalization during segmentation, grouping together pixels surrounded by similar but not identical textures. For clarity, our discussion here will only include the spatial and color features.

A uniform kernel is used for gradient estimation. The kernel has radius vector $h = [h_s, h_s, h_r, h_r, h_r]$, with h_s the radius of the spatial dimensions, h_r the radius of the color dimensions. For every data point (pixel in the original image) the gradient estimate is computed and the center of the kernel, $\mathbf{x}$, is moved in that direction, iterating until the gradient is below a threshold. This change in position is the mean shift vector. The resulting points have gradient approximately equal to zero, and hence are the modes of the density estimate. Each datapoint is then replaced by its corresponding mode estimate.

Finding the mode associated with each data point helps to smooth the image while preserving discontinuities. Let $S_{\mathbf{x}_j,h_s,h_r}$ be the sphere in feature space, centered at point $\mathbf{x}$ and with spatial radius h_s and color radius h_r. The uniform kernel has non-zero values only on this sphere. Intuitively, if two points $\mathbf{x}_i$ and $\mathbf{x}_j$ are far from each other in feature space, then $\mathbf{x}_i \notin S_{\mathbf{x}_j,h_s,h_r}$ and hence $\mathbf{x}_j$ does not contribute to the mean shift vector and the trajectory of $\mathbf{x}_i$ will move it away from $\mathbf{x}_j$. Hence, pixels on either side of a strong discontinuity will not attract each other.

However, filtering alone does not provide a segmentation as the modes found are noisy. This "noise" stems from two sources. First, the mode estimation is an iterative process, hence it only converges to within the threshold provided (and with some numerical error). Second, consider an area in feature space larger than $S_{\mathbf{x},h_s,h_r}$ and where the color is uniform or has a gradient of one in each dimension. Since the pixel coordinates are uniform by design, the mean shift vector will be a 0-vector in this region, and the data points in this region will not move and hence not converge to a single mode. Intuitively, however, we would like all of these data points to belong to the same cluster in the final segmentation. For these reasons, mean shift filtering is only a preprocessing step, and a second step is required in the segmentation process: clustering of the filtered data points $\{\mathbf{x}'\}$.

After mean shift filtering, each data point in the feature space has been replaced by its corresponding mode. As described above, some points may have collapsed to the same mode, but many have not despite the fact that they may be less than one kernel radius apart. In the original mean shift segmentation paper [25], clustering is described as a simple post-processing step in which any modes that are less than one kernel radius apart are grouped together and their basins of attraction are merged. This suggests using single linkage clustering to convert the filtered points into a segmentation.

The only other paper using mean shift segmentation that describes the clustering stage is [22]. In this approach, a region adjacency graph (RAG) is created to hierarchically cluster the modes. Also, edge information from an edge detector is combined with the color information to better guide the clustering. This is the method used in the publicly available EDISON system, also described in [22]. The EDISON system, extended to handle texture, is the implementation we use here as our mean shift segmentation system.

As we can see in Figure 2.1 and will be further discussed in Chapter 3, the regions generated by mean shift segmentation follow image edges well. The number of regions is not specified, but is instead determined by the bandwidths used and the image data. This makes the number of regions highly variable. The region sizes may also vary widely as both large and small image features can be captured, although the range is dependant on the position bandwidth.

2. Efficient Graph-based Segmentation

Efficient graph-based image segmentation, introduced in [36] by Felzenszwalb and Huttenlocher, is another method of performing clustering in feature space. This method works directly on the data points in feature space, without first performing a filtering step, and uses a variation on single linkage clustering. The key to the success of this method is adaptive thresholding. To perform traditional single linkage clustering, a minimum spanning tree of the data points is first generated (using Kruskal's algorithm), from which any edges with length greater than a given hard threshold are removed. The connected components become the clusters in the segmentation. The method in [36] eliminates the need for a hard threshold, instead replacing it with a data-dependent term.

More specifically, let $G = (V, E)$ be a (fully connected) graph, with m edges $\{e_i\}$ and n vertices. Each vertex is a pixel, $\mathbf{x}$, represented in the feature space. Each edge connects two pixels. The final segmentation will be $S = (C_1, ..., C_r)$ where C_i is a cluster of data points. The algorithm is:

1. Sort $E = (e_1, ..., e_m)$ such that $|e_t| \leq |e_{t'}| \, \forall t < t'$
2. Let $S^0 = \big(\{\mathbf{x}_1\}, ..., \{\mathbf{x}_n\}\big)$, in other words each initial cluster contains exactly one vertex.
3. For $t = 1, ..., m$
   (a) Let $\mathbf{x}_i$ and $\mathbf{x}_j$ be the vertices connected by e_t.
   (b) Let $C^{t-1}_{\mathbf{x}_i}$ be the connected component containing point $\mathbf{x}_i$ on iteration $t-1$, and $l_i = \max_{\text{mst}} C^{t-1}_{\mathbf{x}_i}$ be the longest edge in the minimum spanning tree of $C^{t-1}_{\mathbf{x}_i}$. Likewise for l_j.
   (c) Merge $C^{t-1}_{\mathbf{x}_i}$ and $C^{t-1}_{\mathbf{x}_j}$ if
   $$|e_t| < \min\{l_i + \frac{k}{\left|C^{t-1}_{\mathbf{x}_i}\right|}, l_j + \frac{k}{\left|C^{t-1}_{\mathbf{x}_j}\right|}\}$$
   where k is a constant.
4. $S = S^m$

In contrast to single linkage clustering which uses a constant K to set the threshold on edge length for merging two components, efficient graph-based segmentation uses the variable threshold in (3c). This threshold effectively allows two components to be merged if the minimum edge connecting them does not have length greater than the maximum edge in either of the components' minimum spanning trees, plus a term $\tau = \frac{k}{\left|C_{\mathbf{x}_i}^{t-1}\right|}$. As defined here, τ is dependent on a constant k and the size of the component. On the first iteration, $l_i = 0$ and $l_j = 0$, and $\left|C_{\mathbf{x}_i}^0\right| = 1$ and $\left|C_{\mathbf{x}_j}^0\right| = 1$, so k represents the longest edge which will be added to any cluster at any time, $k = l_{max}$. As the number of points in a component increases, the tolerance on added edge length for new edges becomes tighter and fewer mergers are performed, thus indirectly controlling region size. However, it is possible to use any non-negative function for τ which reflects the goals of the segmentation system. Intuitively, in the function used here, k controls the final cluster sizes.

The merging criterion in (3c) allows efficient graph-based clustering to be sensitive to edges in areas of low variability, and less sensitive to them in areas of high variability. Examples of the efficient graph-based algorithm can be seen in Figure 2.1.

3. Hybrid Segmentation Algorithm

An obvious question emerges when describing the mean shift based segmentation method [25] and the efficient graph based clustering method [36]: can we combine the two methods to give better results than either method alone? More specifically, can we combine the two methods to create more stable segmentations that are less sensitive to parameter changes and for which the same parameters give reasonable segmentations across multiple images? In an attempt to answer these questions, the third algorithm we consider is a combination of the previous two algorithms: first we apply mean shift filtering, and then we use efficient graph-based clustering to give the final segmentation. As we will show, it is possible to achieve high-quality segmentations that are less sensitive to their parameters using this approach. Examples of the segmentations generated using this algorithm are available in Figure 2.1. Regions found by this algorithm are capable of capturing long, 'wiry' image structure, however it is also prone to hallucinating wiry structure where there is none.

4. Normalized cuts using boundary maps

Another frequently used segmentation algorithm is Normalized Cuts (NCuts) [104]. The normalized cuts algorithm views the image segmentation problem as a graph cut problem. Let the pixels in an image define a weighted graph $\mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathbf{W})$, where the weights W on the edges correspond to the similarity between the pixels joined by that edge (in some feature space). The cut between two sets of pixels A, B, is $cost(A, B) = \sum_{u \in A, v \in B} w(u, v)$, the total cost of all edges between pixels in A and pixels in B. The segmentation problem can be equated to cutting the graph G into n regions while minimizing the cost of the cuts between them. This, however, can lead to small degenerate regions which are cheap to cut, and other large regions with high variance. To avoid this situation, we can define the association between a region and the rest of the image to be $assoc(A, V) = \sum_{u \in A, t \in V} w(u, t)$, and find regions which minimize the cuts in the graph while also maximizing the associativity in the graph. In other words, normalized cuts seeks to minimize:

$$NCut = \frac{\mathrm{cut}(A, B)}{\mathrm{assoc}(A, V)} + \frac{\mathrm{cut}(A, B)}{\mathrm{assoc}(B, V)} \tag{2.1}$$

Approximations to the minimal normalized cut can be found by converting the cut problem into a Rayleigh quotient and solving for the second smallest eigenvalue, and clustering the corresponding eigenvectors. To create more than one cut, the process can be repeated iteratively on each region.

The graph weights we use in this thesis are computed by inverting the "probability of boundary", or P_b detector developed by Martin et al. [76]. To predict the likelihood of a boundary between two pixels, the "probability of boundary" P_b classifier considers the difference in brightness, color and texture on either side of the proposed boundary and compares these features to a distribution learned on a database of natural images [77].

To implement this procedure, we use the publicly available code by Fowlkes et al. [43,44,76].

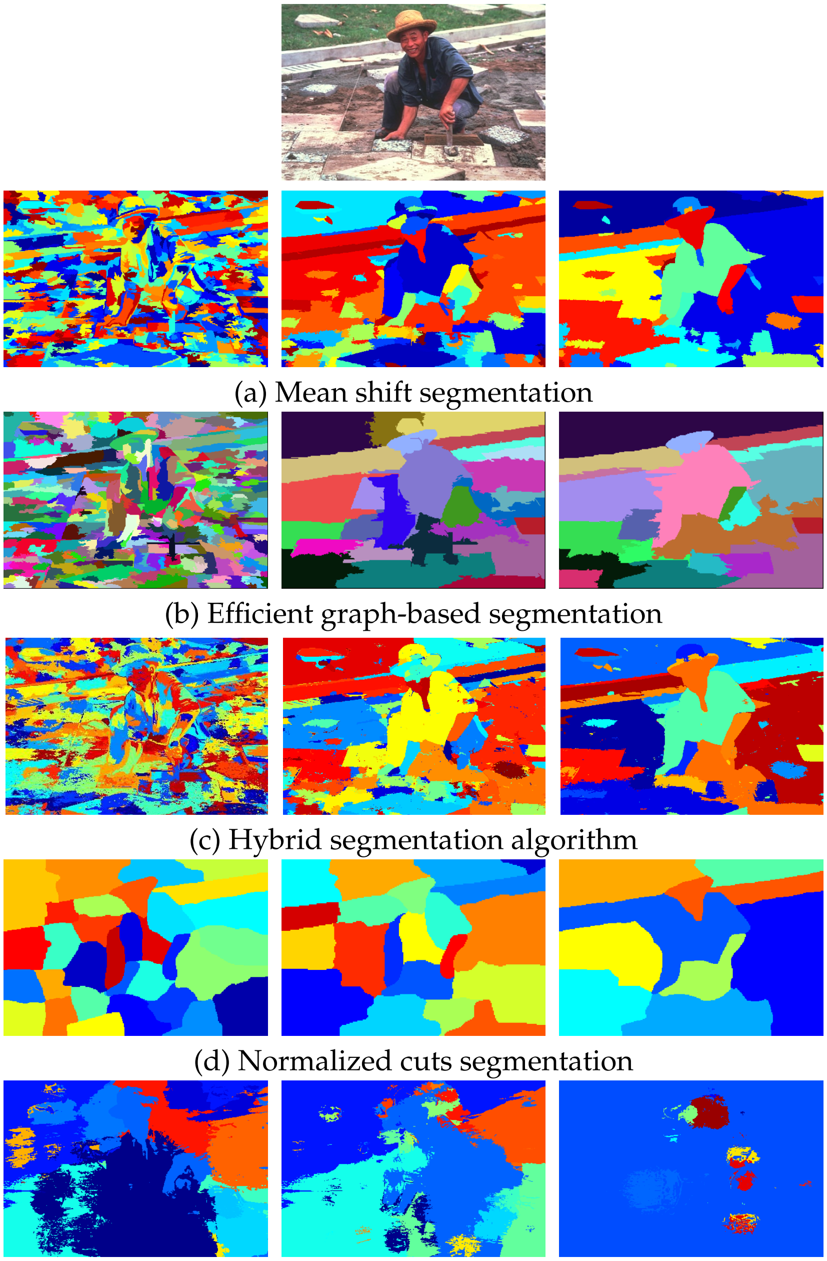
The nature of regions derived from normalized cuts differs from those created by the other algorithms mentioned here, as can be seen in Figure 2.1. Due to normalizing by the association function, regions are often very similar in size. This has the benefit of creating more regularly-shaped regions, but also the weakness that it is often cheaper to make short cuts in homogeneous regions rather than follow long but correct boundaries. Unlike the other segmentation algorithms discussed, Ncuts must be told exactly how many regions to create and cannot make soft decisions. Finally, while the eigenvalue solution generates two regions in a principled manner, extensions to more than two regions are approximate.

5. EM Segmentation Algorithm

As a baseline for the experiments in Chapter 3, we use the classic Expectation Maximization (EM) algorithm [27], with the Bayesian Information Criterion (BIC) to select the number of Gaussians in the model. By minimizing the BIC, we attempt to minimize model complexity while maintaining low error. The BIC is formulated as follows:

$$\mathrm{BIC} = n \ln\left(\frac{RSS}{n}\right) + g \ln(n)$$

where n is the sample size, g is the number of parameters, and RSS is the residual sum of squares. As above, the features used are image position (x, y) and pixel color in L*u*v* color space $(L*, u*, v*)$. We present results for the EM algorithm as a baseline for each relevant experiment, however we omit it in the detailed performance discussion. As can be seen in Figure 2.1, the segmentations generated by EM are of much lower quality.



(a) Mean shift segmentation

(b) Efficient graph-based segmentation

(c) Hybrid segmentation algorithm

(d) Normalized cuts segmentation

(e) Expectation maximization-based segmentation

FIGURE 2.1. Examples of unsupervised segmentations generated by various algorithms. Segmentations in row (a) were generated by the mean shift-based algorithm, row (b) by the efficient graph-based algorithm, row (c) by the hybrid algorithm, row (d) by normalized cuts, and row (e) by the expectation maximization segmentation algorithm. Each row shows the results of using three different parameters settings.

CHAPTER 3

CHARACTERISTICS OF IMAGE SEGMENTATION

IN the previous chapter, we described a number of segmentation algorithms which will be used to generate image regions, and showed examples of such regions. In order to incorporate these regions into an object recognition system we need to make a number of decisions regarding which segmentation algorithms to use and how to use them. The easiest approach would certainly be to use segmentation as a 'black box' which could outline objects for us. Then the rest of our object recognition system would simply need to recognize the regions generated by our segmentation. The segmentations described in the previous chapter lead us to believe that regions are not guaranteed to properly denote objects, but is this in fact always the case? Were we just unlucky with our parameter choice? If we had found the right parameters for one of the algorithms, would it have been possible to denote the objects in all of the images? If not, how much difference is there between a segmentation region and an object? In this chapter, we propose a quantitative study of segmentation performance. We define a set of characteristics which would allow a segmentation algorithm to be a good 'black box', a set of experiments which measure these characteristics and a measure of segmentation accuracy which allows us to perform these experiments.

We present an implementation of our evaluation approach by comparing the segmentation algorithms described in the previous chapter. While we find that we can in fact make recommendations about the relative quality of these algorithms as a black box solution, none of the algorithms gives completely satisfactory results. So instead, we use the information gathered from our experiments to motivate our methods for using segmentation regions in the remainder of this thesis.

1. Segmentation Evaluation Framework

For a segmentation algorithm to be a useful 'black box' in a larger system, we propose that it should have three crucial characteristics:

1. *Correctness*: the ability to produce segmentations which agree with ground truth. That is, segmentations which correctly identify structures in the image at neither too fine nor too coarse a level of detail.
2. *Stability with respect to parameter choice*: the ability to produce segmentations of consistent correctness for a range of parameter choices.
3. *Stability with respect to image choice*: the ability to produce segmentations of consistent correctness using the same parameter choice on different images.

If a segmentation scheme satisfies these three requirements, then it will give useful and predictable results which can be reliably incorporated into a larger system without excessive parameter tuning. It has been argued that the correctness of a segmentation algorithm is only relevant when measured in the context of the larger system into which it will be incorporated. However, most such systems assume that a segmentation algorithm satisfies a subset of the criteria above. In addition, there is value in weeding out algorithms which give nonsensical results and limiting the list of possible algorithms to those that are well-behaved even if the components of the rest of the system are unknown.

It is important to note that a segmentation algorithm can provide useful segmentations despite not meeting the criteria listed here, however the segmentation algorithm could not act as a *black box*. If a segmentation algorithm is weak in any of the above criteria, the larger system would have to make allowances. For example, if an algorithm is not stable with respect to image choice, one could segment each image with multiple parameters, raising the probability that object boundaries are captured in at least one segmentation. This is in fact our approach, which will be discussed later in this chapter and thesis. Thus, in addition to discussing which segmentation algorithm provides the best black box, we will use our experiments to motivate our approach to incorporating segmentation into our object recognition framework.

To evaluate a segmentation algorithm for the above characteristics, we perform a set of experiments on a database of images which have ground truth segmentations, namely the Berkeley segmentation database [77]. For each image in this database, there are roughly 5-7 ground truth human segmentations to which to compare machine-generated segmentations, with examples given in Figure 3.1. To test an algorithm for each characteristic listed above, we will perform the following experiments, with numbers corresponding to each characteristic:

1. To measure the correctness of an algorithm, we will generate multiple segmentations of each image in the database with multiple parameter settings. For each image, the segmentation that best corresponds to the ground truth is an approximation of the best performance possible by the algorithm.
2. To measure the stability of an algorithm with respect to parameter choice, we must once again segment each image with multiple parameters. If the segmentation quality differs substantially with different parameters, then the algorithm is unstable.
3. To measure the stability of an algorithm with respect to image choice, we segment all of the images in the database with the same parameters. If the segmentation quality differs substantially between images, then the algorithm is unstable.

Many past evaluations of segmentation performance have been merely qualitative. In contrast, we wish to perform a quantitative evaluation, so we require a measure of the accuracy of an image segmentation compared to its ground truth. In order to effectively carry out the experiments suggested, the measure must be able to compare a segmentation against *multiple* ground truth human segmentations. In

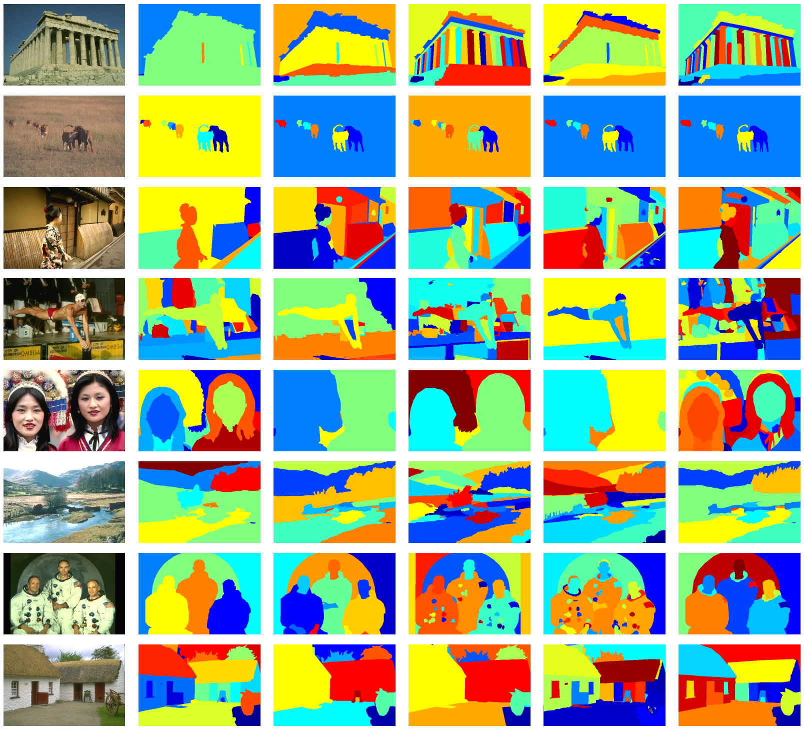


FIGURE 3.1. Examples of images from the Berkeley image segmentation database [77] with five of their human segmentations. Note the variation in region refinement between the human segmenters.

addition, it must not have any degenerate cases in which a particular segmentation, (such as every pixel belonging to a separate region), is given an artificially high score. Next, since we wish to compare multiple segmentations of the same image, as well as segmentations of different images, the measure cannot make any assumptions about how the regions are generated, nor the number or size of the regions. Since we also wish to compare to multiple ground truth segmentations which may agree or disagree in different image areas, it would be beneficial if the measure could adaptively accommodate to the level of agreement over the image. Note that in the human segmentations in Figure 3.1, there are some image areas where all of the ground truth segmentations are in agreement, and other areas where there are varying levels of disagreement. Finally, it is important that the score generated by the measure be easily interpretable and comparable across images. In the next section, we introduce the Normalized Probabilistic Rand index, a measure of segmentation accuracy which does indeed meet all of these criteria.

2. NPR measure

In [115], Unnikrishnan and Hebert introduce a measure for evaluating the similarity of a novel segmentation to a set of ground truth image segmentations, the Probabilistic Rand (PR) Index. The original Rand Index [92], a popular non-parametric measure, measures the agreement between two segmentations as a function of the number of pairs of points whose label (or region) relationships agree in both segmentations. In other words, if two points belong to the same region in one segmentation, they should also belong to the same region in the other segmentation. The same holds true if the two points belong to different regions. Formally, let $X = \{x_i\}$, $i = 1..N$ be a set of points, and S and S' two segmentations of those points. Let l_i be the label, or region, of point x_i in segmentation S, and l'_i the label in segmentation S'. The Rand Index is the fraction of pairs of points whose relationships agree in both segmentations, computed as:

$$R(S,S') = \frac{1}{\binom{N}{2}} \sum_{\substack{i,j\\ i\neq j}} \left[\ \mathbb{I}\big(l_i = l_j \wedge l'_i = l'_j\big) + \mathbb{I}\big(l_i \neq l_j \wedge l'_i \neq l'_j\big)\ \right] \tag{3.1}$$

where $\mathbb{I}$is an indicator function. Notice the lack of reference to *which* region a pair of points belongs. Also, S and S' may have different numbers of regions. If $S' = S_{test}$ is a novel segmentation and S is a human segmentation, the Rand Index gives the correctness of segmentation S'.

The Probabilistic Rand (PR) index extends the Rand Index to handle multiple ground truth segmentations. Let $\{S_1, ..S_K\}$ be a set of ground truth segmentations for image $X = \{x_i\}$, and $l_i^{S_k}$ the label of pixel x_i in segmentation S_k. Define p_{ij} to be the probability that x_i and x_j are given the same label over all consistent human segmentations of the image. The PR index is defined as:

(3.2)

$$\begin{aligned}\mathrm{PR}(S_{\text{test}}, \{S_k\}) &= \frac{1}{\binom{N}{2}} \sum_{\substack{i,j\\ i<j}} \left[\mathbb{I}\left(l_i^{S_{\text{test}}} = l_j^{S_{\text{test}}}\right) p_{ij} + \left(1 - \mathbb{I}\left(l_i^{S_{\text{test}}} = l_j^{S_{\text{test}}}\right)\right)(1-p_{ij})\right] \\ &= \frac{1}{\binom{N}{2}} \sum_{\substack{i,j\\ i<j}} \left[c_{ij}p_{ij} + (1-c_{ij})(1-p_{ij})\right]\end{aligned}$$

where $c_{ij} = \mathbb{I}\Big(l_i^{S_{\text{test}}} = l_j^{S_{\text{test}}}\Big)$. The PR index has a range of $[0, 1]$, with 1 being most desirable. If we consider that the set of available human segmentations is representative of all human segmentations, then a useful simplification is to assign:

$$p_{ij} = \frac{1}{K} \sum_{k=1..K} \mathbb{I}\Big(l_i^{S_{\text{k}}} = l_j^{S_{\text{k}}}\Big) \tag{3.3}$$

In which case the PR index can be computed efficiently, as shown in [117].

The PR index possesses three of the characteristics we require of a measure well-suited to a comparison of segmentation algorithms. We discuss other measures of segmentation accuracy in Chapter 2.1, however none of these measures possess all three characteristics.

The first characteristic is a lack of degenerate cases where the measure is artificially high for unrealistic scenarios. In order to have a high PR index, a novel segmentation must be well-represented in the ground truth segmentation set.

The second characteristic is a lack of assumptions about the segmentation generation method. The number of labels and region sizes in each segmentation can vary without constraint.

The third characteristic that the PR index possesses is the ability to accommodate to region refinement adaptively given the ground truth segmentations. In other words, a novel segmentation is not penalized for subdividing a larger region if there is support among the ground truth segmentations for the subdivision. However, if all of the ground truth segmentations agree on a region, then the novel segmentation is penalized for subdividing it. This differs from other measures which either do not allow refinement at all, or allow all refinement regardless of the ground truth segmentations.

The results of the PR index, however, are difficult to interpret and compare across images. What is a ‘good’ PR index? Does a value of, say, 0.5 have the same meaning for two different images? How difficult is a given image to segment? Unfortunately, the meaning of the PR Index is linked to a given image, and there is no baseline of ‘good enough’ performance for all images. This is a problematic weakness with respect to our segmentation evaluation experiments.

In joint work with Unnikrishnan and Hebert [116, 117], we seek to solve these issues regarding the PR index by introducing the Normalized Probabilistic Rand (NPR) index. Our goal is to assign a baseline of expected performance for the PR index on a given image, and then normalize the PR index with respect to that baseline:

$$\text{Normalized index} = \frac{\text{Index} - \text{Expected index}}{\text{Maximum index} - \text{Expected index}} \tag{3.4}$$

In this formulation any segmentation with a PR index above the expected index on a given image, or an NPR index above 0, is better than random. In addition, since the NPR index properly accounts for the variance across the set of images, its value on different images is comparable.

The maximum index in the normalization equation can be set to an image-specific data-dependent value or 1. The statistics literature is split on the correct choice [125]. We choose to set Maximum index $= 1$ due to the computational complexity of evaluating the maximal-scoring *consistent* segmentation for each image.

To compute the normalization in Equation 3.4, we need to compute the expected value of the PR index on a given image:

(3.5)
$$\begin{aligned} \mathbb{E}\Big[\text{PR}(S_{\text{test}}, \{S_k\})\Big] &= \frac{1}{\binom{N}{2}} \sum_{\substack{i,j \\ i<j}} \Big\{ \mathbb{E}\Big[\mathbb{I}\Big(l_i^{S_{\text{test}}} = l_j^{S_{\text{test}}}\Big)\Big] p_{ij} + \mathbb{E}\Big[\mathbb{I}\Big(l_i^{S_{\text{test}}} \neq l_j^{S_{\text{test}}}\Big)\Big](1 - p_{ij}) \Big\} \\ &= \frac{1}{\binom{N}{2}} \sum_{\substack{i,j \\ i<j}} \Big[p'_{ij} p_{ij} + (1 - p'_{ij})(1 - p_{ij}) \Big] \end{aligned}$$

What is a meaningful way to compute $p'_{i,j} = \mathbb{E}\big[\mathbb{I}\big(l_i^{S_{\text{test}}} = l_j^{S_{\text{test}}}\big)\big]$? We propose that the baseline for normalization must be representative of perceptually consistent groupings of random but *realistic* images. This translates to estimating p'_{ij} from segmentations of *all* images. Let Φ be the images in a dataset, and K_ϕ the number of ground truth segmentations of image ϕ. Then p'_{ij} can be expressed as:

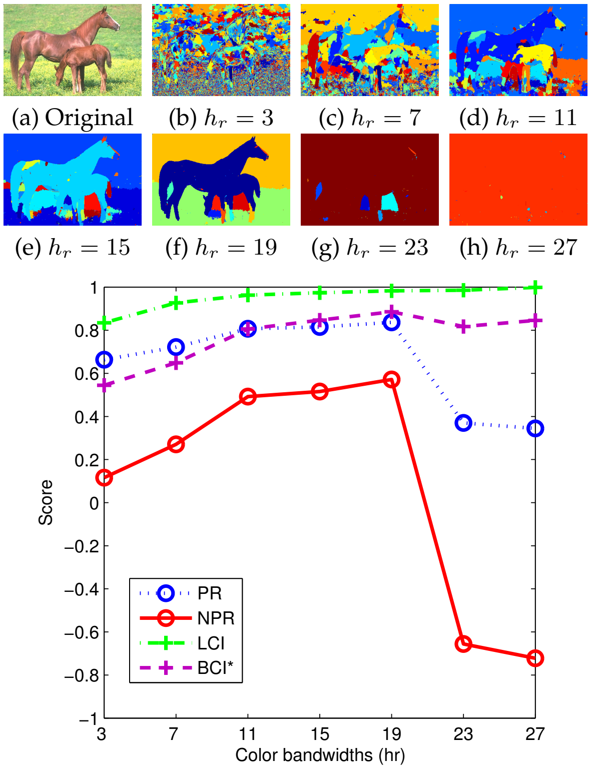
$$p'_{ij} = \frac{1}{\Phi} \sum_{\phi} \frac{1}{K_\phi} \sum_{k=1}^{K_\phi} \mathbb{I}\Big(l_i^{S_k^\phi} = l_j^{S_k^\phi}\Big) \tag{3.6}$$

Using this formulation for p'_{ij} implies that $\mathbb{E}[\mathrm{PR}(S_{\text{test}}, \{S_k\})]$ is just a (weighted) sum of $\mathrm{PR}(S_k^{\phi}, \{S_k\})$. Although $\mathrm{PR}(S_k^{\phi}, \{S_k\})$ can be computed efficiently [117], performing this computation for every segmentation S_k^{ϕ} is expensive, so in practice we uniformly sample 5×10^6 pixel pairs for an image size of 321×481 ($N = 1.5 \times 10^5$) (the size of images in the Berkeley segmentation database) instead of computing it exhaustively over all pixel pairs. Experiments performed using a subset of the images indicated that the loss in precision in comparison with exhaustive evaluation was not significant for this number of samples.

As intuition for what the expected index might look like as an actual segmentation, Figure 3.3 shows two segmentations with NPR indices close to zero.

Our philosophy that the baseline should depend on the empirical evidence from all of the images in a ground truth training set is fundamentally different from previous normalization schemes. Both [56] and [45] introduce normalization schemes for the Rand Index which assume that segmentations are generated by a hypergeometric distribution. This has the unfortunate assumptions that segmentations are independent, and that the probabilities of each label are constant. In addition, the expected value in these schemes is computed over all theoretically possible segmentations with constant cluster proportions, regardless of their plausibility. In comparison, the empirical baseline in the Normalized Probabilistic Rand index (NPR) has two important benefits:

First, since p'_{ij} and p_{ij} are modeled from the ground truth data, the number and size of the clusters in the images do not need to be held constant. Thus, the error produced by two segmentations with differing cluster sizes can be compared. In a segmentation algorithm evaluation, this property allows the comparison of the algorithm's performance with different parameters. In Figure 3.2, the top two rows show an image from the segmentation database [77] and segmentations of different granularity created by different parameters to the segmentation algorithm. The PR index does reflect the correct relationship among the segmentations, however its range is small and the expected value is unknown, hence it is difficult to decide whether any of the segmentations are "good". The NPR index fixes these problems.


(a) Original (b) $h_r = 3$ (c) $h_r = 7$ (d) $h_r = 11$

(e) $h_r = 15$ (f) $h_r = 19$ (g) $h_r = 23$ (h) $h_r = 27$

FIGURE 3.2. Example of changing scores for different segmentation granularities: (a) Original image, (b)-(h) mean shift segmentations using scale bandwidth (h_s) 7 and color bandwidths (h_r) $3, 7, 11, 15, 19, 23$ and 27 respectively. The plot shows the LCI, BCI*, PR and the NPR similarity scores for each segmentation. Only the NPR index reflects the intuitive accuracy of each segmentation of the image. The NPR index correctly shows that segmentation (f) is the best one, segmentations (d), (e), and (f) are reasonable, and segmentations (g) and (h) are worse than a random segmentation.

It reflects the desired relationships among the segmentations with no degenerate cases, and any segmentation with a score significantly above 0 is known to be useful. This comparison would have been impossible with the other normalization schemes.

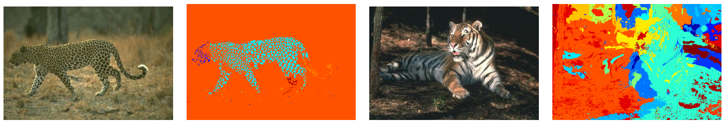


FIGURE 3.3. Examples of segmentations with NPR indices near 0. Segmentations with NPR indices higher than 0 are better than a random realistic and consistent segmentation, while those with NPR indices below 0 are worse than expected.

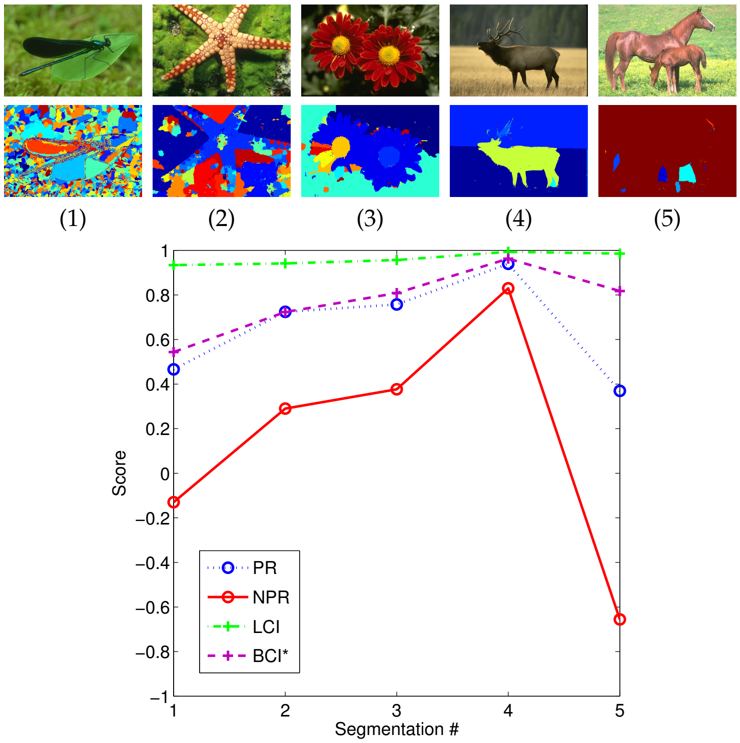


FIGURE 3.4. Example comparison of segmentations of different images: (1)-(5) Top row: Original images, Second row: corresponding segmentations. The plot shows the LCI, BCI*, PR and the NPR similarity scores for each segmentation as numbered. Only the NPR index reflects the intuitive accuracy of each segmentation across images.

Second, since p'_{ij} is modeled using *all* of the ground truth data, the NPR indices of segmentations of *different* images are comparable. This facilitates the comparison of an algorithm's performance on different images. Figure 3.4 shows the scores of segmentations of different images. The first row contains the original images and the second row contains the segmentations. Once again, the NPR is the only index which both shows the desired relationship among the segmentations and whose output is easily interpreted.

The images in Figure 3.5 and Figure 3.6 show additional examples of comparison across images, and demonstrate the consistency of the NPR index. In Figure 3.5(b), both segmentations are perceptually equally "good" (given the ground truth segmentations), and correspondingly their NPR indices are high and similar. The segmentations in Figure 3.6(b) are both perceptually "bad" (over-segmented), and correspondingly both of their NPR indices are very low.

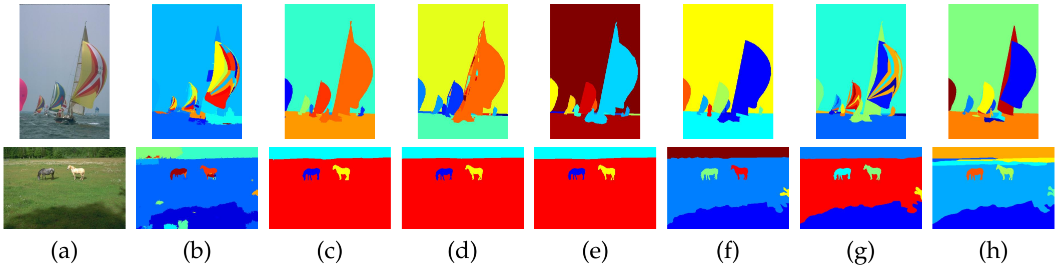


FIGURE 3.5. Examples of "good" segmentations: (a) Images from the Berkeley segmentation database, (b) mean shift segmentations (using $h_s = 15$, $h_r = 10$), and (c-h) their ground truth hand segmentations. Top image: NPR = 0.8938, Bottom image: NPR = 0.8495

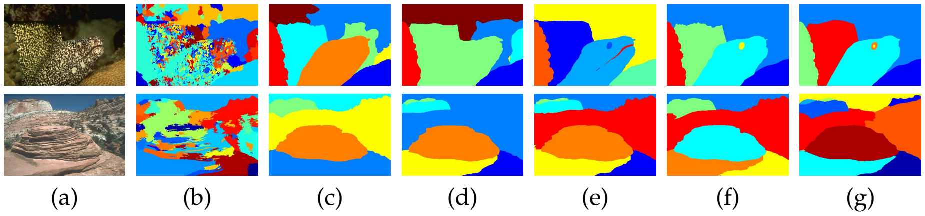


FIGURE 3.6. Examples of "bad" segmentations: (a) Images from the Berkeley segmentation database, (b) mean shift segmentations (using $h_s = 15$, $h_r = 10$), and (c-g) their ground truth hand segmentations. Top image: NPR = -0.7333, Bottom image: NPR = -0.6207

2.1. Other measures of segmentation quality

There are a variety of other measures of segmentation quality in the literature. Our NPR index is inspired by a group of non-parametric measures including Cohen's Kappa [23], Jaccard's index, Fowlkes and Mallow's index [45]and the (Adjusted) Rand index [56, 92]. None of these measures, however, can compare more than two segmentations and so cannot cope with multiple ground truth segmentations.

Comparison methods which look at the agreement between boundaries, including [30, 46, 55, 75], are fragile. They require a method for matching boundary pixels which may not align perfectly. In addition, they do not allow any region refinement. Finally, boundary matching methods ignore the quality of *unmatched* boundary pixels, which means that the unmatched boundaries can be anywhere in the image and yet give the same score.

The measures introduced in [33, 75] phrase the segmentation problem as a binary classifier and examine such statistics as false positive and negative rates, and the associated precision and recall. As a consequence of this representation, they must assume only one ground truth. In addition, the amalgamated statistics computed discard all spatial information.

In [44, 75], the mutual information score between the pixel pairs in a test segmentation and ground truth segmentations is used as a measure of segmentation quality. In its implementation, however, this score only counts pixel pairs which have the same region relationship in all of the ground truth segmentations. Pixel pairs which are ambiguous are ignored, regardless of the degree of ambiguity.

The Variation of Information measure [78, 79] is a promising approach which considers the amount of information in each segmentation as well as the mutual information between segmentations. If extended to handle multiple ground truth images, this approach would have interesting properties [79].

The measures in [21, 72] consider the overlap between each cluster in a test segmentation and its best approximation in a ground truth segmentation. As a consequence, they must perform expensive region matching, and they are completely intolerant of refinement. A common formulation of this paradigm is the region overlap score:

$$\text{Overlap}(R_1, R_2) = \frac{|R_1 \cap R_2|}{|R_1 \cup R_2|} \tag{3.7}$$

Perhaps the most studied and applied measures of segmentation quality are those by Martin et al. [75,77], which include multiple measures for comparing segmentations of different granularity. We describe two of the measures here to which we compare the NPR index in Figures 3.2, 3.4, 3.7, and 3.8.

Following the notation for images and segmentations above, let $C(S, x_i)$ be the segment (class) that contains pixel x_i in segmentation S. For a pixel x_i, define the local refinement error (LRE) as:

$$\text{LRE}(S, S', x_i) = \frac{|C(S, x_i) \backslash C(S', x_i)|}{|C(S, x_i)|}$$

where \ denotes the directed set differencing operator.

A simple approach to combining the LRE at each pixel is the Global Consistency Error (GCE), which either allows the test segmentation to be a refinement of the ground truth segmentation, or vice versa, but does not allow different refinement directions in different parts of the image. The GCE is defined as:

$$\text{GCE}(S, S') = \frac{1}{N} \min\Big\{ \sum_i \text{LRE}(S, S', x_i), \sum_i \text{LRE}(S', S, x_i) \Big\}$$

We compare instead to the Local Consistency Error (LCE) since it, like the NPR index, allows different refinement relationships in different parts of the image. The LCE is defined as:

$$\text{LCE}(S, S') = \frac{1}{N} \sum_i \min\big\{ \text{LRE}(S, S', x_i), \text{LRE}(S', S, x_i) \big\}$$

Due to the variation in refinement error in the LCE, it is always the case that LCE $\leq$ GCE. Since the NPR index is a measure of similarity, for comparison purposes we convert the Local Consistency *Error* to the Local Consistency *Index* (LCI) by LCI $= 1 -$ LCE. The LCI has a value of 0 when the two segmentations are most dissimilar, and a score of 1 when they are most similar.

The LCI measure has two weaknesses. First, it can only compare two segmentations, not a set of ground truth segmentations. In order to allow for comparison with other measures, the LCI reported in Figures 3.2 and 3.4 is the average of the LCI scores of the test segmentation with each ground truth segmentation.

Second, as discussed by Martin [75], the LCI has degenerate cases which give unjustifiably high scores. Any segmentation in which each pixel is a separate segment will receive a score of 1, as will any segmentation with only one region. In general, subdividing ground truth regions, or grouping them together, does not lower the LCI. In Figure 3.7 we can see an image that has been over-segmented compared to the ground truth segmentations, and in Figure 3.8 an image that has been under-segmented. The LCI gives both segmentations artificially high scores. In Figure 3.2 we can see how this lack of sensitivity leads to flat scores across varying segmentation granularities. In Figure 3.4, we can see the same situation across different images. In this Figure, the NPR index is the only index which both penalizes the segmentations, and has easily interpretable negative scores which indicate poor performance.

In order to solve some of the problems of the LCI, Martin introduces the Bidirectional Consistency Error* (BCE*) [75], defined as:

$$\text{BCE*}(S_{\text{test}}, \{S_k\}) = \frac{1}{N}\sum_{i=1}^{N} \min_k \Big\{ \max\big\{ \text{LRE}(S_{\text{test}}, S_k, x_i), \text{LRE}(S_k, S_{\text{test}}, x_i) \big\} \Big\}$$

This formulation compares a pixel's segment to its best-approximating segment in any segmentation in a ground truth set. By taking the maximum over the two directions of the LRE, the BCE* avoids degenerate cases. However, by taking a hard minimum over the maximal overlapping region scores, the BCE* ignores the relative frequency information available in the ground truth data. To facilitate comparison, we define the Bidirectional Consistency *Index** (BCI*) as BCI* $= 1 -$ BCE*, which also takes values in $[0, 1]$. In Figures 3.8, 3.7, 3.2, and 3.4, we can see that the BCI* does indeed penalize over- and under- segmentations more than the LCI. However it is still difficult to interpret the score, and its high value is misleading.

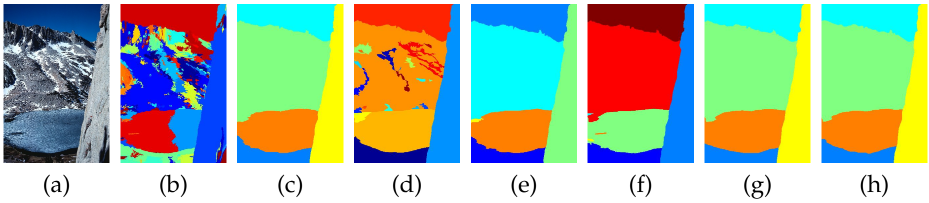


FIGURE 3.7. An example of over-segmentation: (a) Image from the Berkeley segmentation database,(b) its mean shift segmentation (using $h_s = 15$ (spatial bandwidth), $h_r = 10$ (color bandwidth)), and (c-h) its ground truth hand segmentations. Average LCI = 0.9370, BCI* = 0.7461, PR = 0.3731, NPR = -0.7349

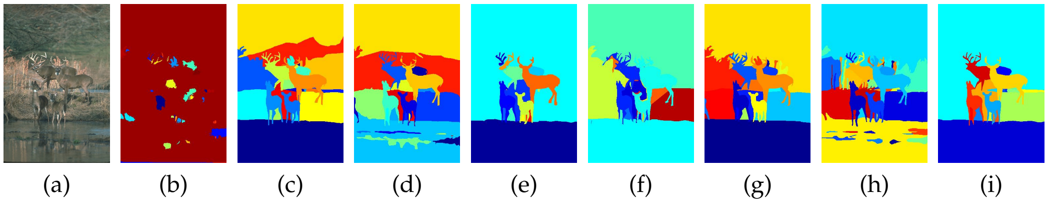


FIGURE 3.8. An example of under-segmentation: (a) Image from the Berkeley segmentation database,(b) its mean shift segmentation (using $h_s = 15$, $h_r = 10$), and (c-i) its ground truth hand segmentations. Average LCI = 0.9497, BCI* = 0.7233, PR = 0.4420, NPR = -0.5932

3. Experiments

As an example of our evaluation framework, and motivation for the remainder of this thesis, we present an evaluation of four of the segmentation techniques described in the previous chapter: mean shift segmentation [25], denoted 'MS' or 'EDISON' for the publicly available EDISON implementation, the efficient graph-based segmentation algorithm introduced by Felzenszwalb and Huttenlocher [36], denoted 'FH', the hybrid variant that combines these algorithms [87,117], denoted 'MS+FH', and expectation maximization [27], denoted 'EM', as a baseline. All of the experiments were performed on the publicly available Berkeley image segmentation database [77].

In all of the following experiments we have fixed the spatial bandwidth $h_s = 7$, since it seems to be the least sensitive parameter and removing it makes the comparison more approachable. Also, although the FH algorithm as defined previously only had one parameter, k, we need to add two more. In order to properly compute distance in our feature space $\{x, y, L^*, u^*, v^*\}$, we rescale the data by dividing each dimension by the corresponding $\{h_s, h_r\}$. The same procedure is applied to the EM algorithm. So each algorithm was run with a parameter combination from the sets: $h_s = 7$, $h_r = \{3, 7, 11, 15, 19, 23\}$, and $k = \{5, 25, 50, 75, 100, 125\}$. We mildly abuse notation by using h_r and h_s to denote parameters for all of the algorithms to avoid introducing extra terms. The axes for each plot type are kept constant for ease of comparison.

3.1. Maximum performance

The first set of experiments examines the correctness of the segmentations produced by the three algorithms with a reasonable set of parameters. The left plot in Figure 3.9 shows the maximum NPR index on each image for each algorithm. The indices are plotted in increasing order for each algorithm, hence image 190 refers to the images with the 190th lowest index for each algorithm, and may not represent the same image across algorithms. The right plot in Figure 3.9 is a histogram of the same information, showing the number of images per maximum NPR index bin.

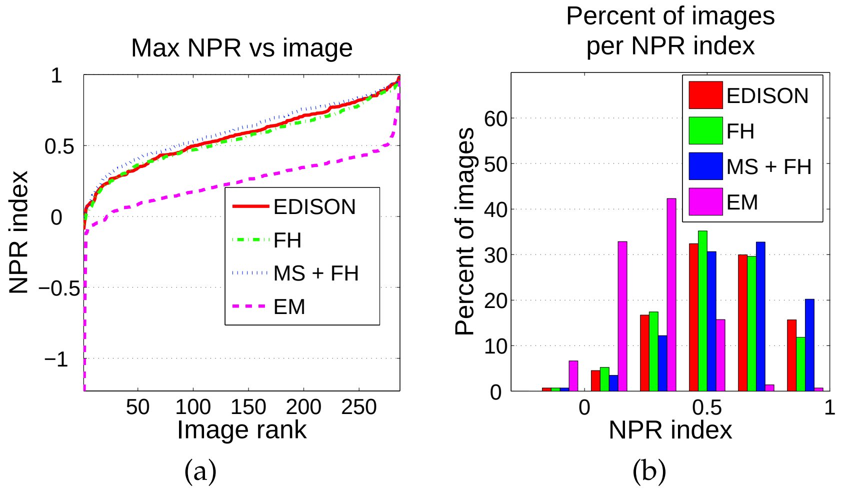


FIGURE 3.9. Maximum NPR indices achieved on individual images with the set of parameters used for each algorithm. Plot (a) shows the indices achieved on each image individually, ordered by increasing index. Plot (b) shows the same information in the form of a histogram. Recall that the NPR index has an expected value of 0 and a maximum of 1.

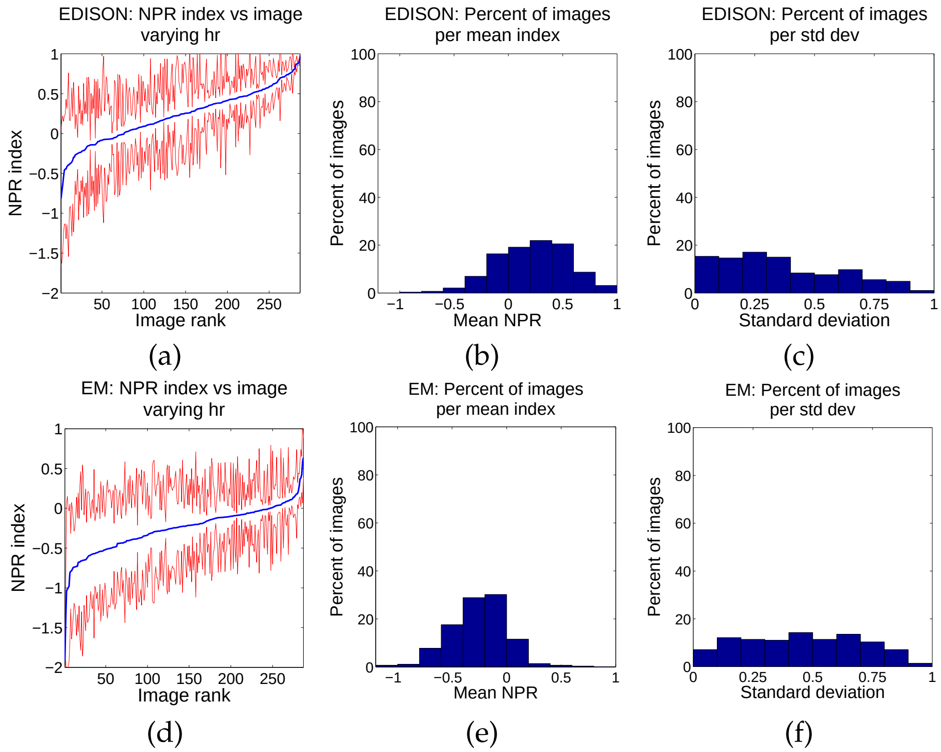
All of the algorithms, except EM, produce similar maximum NPR indices, demonstrating that they have roughly equal ability to produce correct segmentations with the parameter set chosen. There are very few images which have below-zero maximum NPR index, hence all of the algorithms almost always have the potential to produce useful results. These graphs also demonstrate that our parameter choices for each algorithm are reasonable.

3.2. Average performance per image

Given that each algorithm (except EM) has the potential to produce equally correct segmentations, the next issue we explore is whether they also produce correct segmentations *on average*. The next set of plots in Figures 3.10-3.15 examine correctness through the mean index achieved on each image. The first plot in each row shows the mean NPR index on each image achieved over the set of parameters used (in increasing order of the mean), along with one standard deviation. The second plot in each row is a histogram of the mean information, showing the number of images per mean NPR index bin. An algorithm which creates good segmentations will have a histogram skewed to the right. The third plot in each row is a histogram of the standard deviations.

These plots also partially address the issue of stability with respect to parameters. A standard deviation histogram that is skewed to the left indicates that the algorithm in question is less sensitive to changes in its parameters.

Using the means as a measure certainly makes us more dependent on our choice of parameters for each algorithm. Although we cannot guarantee that we have found the best or worst parameters for any individual algorithm, we can compare the performance of the algorithms with identical parameters.



(a) (b) (c)

(d) (e) (f)

FIGURE 3.10. Mean NPR indices achieved on individual images over the parameter set $h_r = \{3, 7, 11, 15, 19, 23\}$. Results for the mean shift-based system (EDISON) are given in plots (a), (b) and (c), and results for EM are given in (d), (e) and (f). Plots (a) and (d) show the mean indices achieved on each image individually, ordered by increasing index, along with one standard deviation. Plots (b) and (e) show histograms of the means. Plots (c) and (f) show histograms of the standard deviations.

3.2.1. Average performance over different values of the color bandwidth h_r. We compare the NPR indices averaged over values of h_r, with k held constant. The plots showing this data for the EDISON method are in Figure 3.10. Figure 3.11 gives the plots for the efficient graph-based segmentation system (FH) and the hybrid algorithm (MS+FH) for $k = \{5, 25, 125\}$. We only show three out of the six values of k in order to keep the amount of data presented reasonable. The most interesting comparison here is between the EDISON system and the hybrid system, which reflects the impact the addition of the efficient graph-based clustering has had on the segmentations produced.

The experiments show that for $k = 5$, the performance of the hybrid (MS+FH) system is slightly better and certainly more stable than that of the mean shift-based (EDISON) system. For $k = 25$, the performance is more comparable, but the standard deviation is still somewhat lower. Finally, for $k = 125$, the hybrid system performs comparably to the mean-shift based system. Thus the change to using the efficient graph-based clustering after the mean shift filtering has maintained the correctness of the mean shift-based system while improving its stability.

Looking at the graphs for the efficient graph-based segmentation system alone in Figure 3.11, we can see that although for $k = 5$ the mean performance and standard deviation are promising, they quickly degrade for larger values of k. This decline is much more gradual in the hybrid algorithm.

The results show that the mean indices of the hybrid system are both higher and more stable (with respect to changing values of k) than those of the efficient graph-based segmentation system. Hence, adding a mean shift filtering preprocessing step to the efficient graph-based segmentation system is an improvement.

3.2.2. Average performance over different values of k. The mean NPR indices as k is varied through $k = \{5, 25, 50, 75, 100, 125\}$ and h_r is held constant are displayed in figure Figure 3.15. Once again we only look at a representative three out of the six possible h_r values, $h_r = \{3, 7, 23\}$. Since the mean shift-based system does not use k, this comparison is between the efficient graph-based segmentation system and the hybrid system.

Examples of these results can be seen in Figures 3.2, 3.13 and 3.14. In Figure 3.2, we can see that mean shift segmentation using the EDISON system produces segmentations that correspond well to human perception. However, this algorithm is quite sensitive to its parameters. Variations in the color bandwidth h_r can cause large changes in the granularity of the segmentation. By adjusting the color bandwidth we can produce over-segmentations as in Figure 3.2b, to reasonably intuitive segmentations as in Figure 3.2f, to under-segmentations as in Figure 3.2g.

The merging criterion in Equation 3c allows efficient graph-based clustering to be sensitive to edges in areas of low variability, and less sensitive to them in areas of high variability. However, Figure 3.13 shows that the resulting segmentations do not have the same degree of correctness with respect to the ground truth as mean shift-based segmentation. This algorithm also suffers somewhat from sensitivity to its parameter, k.

The result of applying the hybrid algorithm with different parameters can be seen in Figure 3.14. For $h_r = 15$ the quality of the segmentation is high. Also, the rate of granularity change is slower than either of the previous two algorithms, even though the parameters cover a wide range.

3.2.3. Conclusions. From our experiments regarding average performance per image, we conclude that the hybrid algorithm is the preferable black box to use in a larger system. It provides equal performance to the mean shift-based algorithm and better performance than the efficient graph-based algorithm, as well as less variance with changing parameters.

3.3. Average performance per parameter choice

In the above experiments, we determined that for each image the segmentation quality differs as the segmentation parameters are changed. We now explore whether one set of parameters gives consistent results over the entire set of images, which would allow us to simply use the 'best' parameter setting for all images. In each experiment results are shown with respect to a particular parameter, with averages and standard deviations taken over segmentations of all the images in the database.

3.3.1. Average performance over all images for different values of h_r. The first three sets of graphs show the results of keeping k constant and choosing from the set $h_r = \{3, 7, 11, 15, 19, 23\}$. Figure 3.12 shows the results of running the EDISON system with these parameters, averaged over the image set and with one standard deviation. Figure 3.16 shows the same information for the efficient graph-based segmentation (FH) and the hybrid (MS+FH) system on a representative three of the six possible values of k. For completeness, the graphs for the remaining values of k can be found in [87].

As before, we can see that the hybrid algorithm gives slight improvements in stability over the mean shift-based system, but only for smaller values of k. We can also see that, except for $k = 5$, both the mean shift-based system and the hybrid system are more stable across images than the efficient graph-based segmentation system.

3.3.2. Average performance over all images for different values of k. The last two sets of graphs, in Figure 3.17, examine the stability of k over a set of images. Each graph shows the average algorithm performance taken over the set of images with a particular h_r and each point shows a particular k. The graphs show a representative subset of the choices for h_r, and the remaining graphs can be found in [87]. Once again we see that combining the two algorithms has improved performance and stability. The hybrid algorithm has higher means and lower standard deviations than the efficient graph-based segmentation over the image set for each k, and especially for lower values of h_r.

3.3.3. Conclusions. Across images, the hybrid algorithm gives comparable performance to the mean shift algorithm for higher values of k, so with respect to this criteria either algorithm would be an appropriate black box.

4. Segmentation Evaluation Conclusions

In this section we have proposed a framework for comparing image segmentation algorithms using the NPR index, and performed one such comparison. Our framework consists of comparing the performance of segmentation algorithms based on three important characteristics: correctness, stability with respect to parameter choice, and stability with respect to image choice. We chose to compare four segmentation algorithms: mean shift-based segmentation [22, 25], a graph-based segmentation scheme [36], a proposed hybrid algorithm, and expectation maximization [27] as a baseline.

The first three algorithms had the potential to perform equally well on the dataset given the correct parameter choice. However, examining the results from the experiments which averaged results over parameter sets, the hybrid algorithm performed slightly better than the mean shift algorithm, and both performed significantly better than the graph-based segmentation. We can conclude that the mean shift filtering step is indeed useful. As expected, EM performed significantly worse than any of the other algorithms both in terms of maximum and average performance.

In terms of stability with respect to parameters, the hybrid algorithm showed less variability when its parameters were changed than the mean shift segmentation algorithm, although the amount of improvement did decline with increasing values of k. Although the graph-based segmentation did show very low variability with $k = 5$, changing the value of k decreased its stability drastically.

Finally, in terms of stability of a particular parameter choice over the set of images, we see that the graph-based algorithm has low variability when $k = 5$, however its performance and stability decrease rapidly with changing values of k. The difference between the mean shift segmentation and the hybrid method is negligible.

We conclude that both the mean shift segmentation and hybrid segmentation algorithms can create realistic segmentations with a wide variety of parameters, however the hybrid algorithm has slightly improved stability. We thus recommend the hybrid algorithm as the best black box segmentation algorithm we studied. None of the algorithms, however, created segmentations that were stable or corresponded to human intuition.

Our segmentation evaluation compared algorithms which have equivalent parameters. Due to the experimental scheme, we could not compare algorithms such as normalized cuts because they require as input the number of regions to generate. Although altering the bandwidths of the above algorithms does change the number of regions, it is unclear which parameters should be changed to generate a given number of regions. For example, if both changing h_r or k can create segmentations with, say, 20 regions, which parameter should be used? In addition, in the above experiments, using small bandwidths often generated segmentations with very large numbers of regions, however generating normalized cuts segmentations with a large number of regions is computationally very expensive. For these reasons, we look to other comparisons discussed below for additional information regarding the normalized cuts algorithm.

5. Other evaluations of segmentation algorithms

Upon introducing a new segmentation algorithm, most publications supply only a subjective visual assessments of segmentation quality on a small image set. These assessments give a notion as to possible segmentation issues, but cannot quantify their results. Visual examinations can also be misleading, as found by Everingham et al [33].

Shaffrey et al. [101] propose to have human subjects pick the best segmentation of a group for each image. This is an interesting notion, however completely impractical, as the same human subjects would have to be re-employed each time a new algorithm was added to the evaluation.

Martin [75] uses the LCE measure to study whether, allowing refinement, humans agree on the segmentation of an image. From the ground truth segmentations in the Berkeley database, he concludes that they are in fact consistent up to refinement. However, since the LCE measure does not penalize refinement at all, he cannot draw any conclusions about whether different people interpret the notion of objects differently. In fact, we can see from our examples of Berkeley database images and segmentations that in fact the human segmentations can differ quite substantially.

Martin [75] and Estrada and Jepson [30] perform experiments examining the accuracy of region boundaries, experiments which are extended on the Berkeley database webpage [77]. It is interesting to note which algorithms produce edge maps which are supersets of the human object boundary maps, however such a comparison is substantially different from our own. Given that the comparison is between edge maps only, it does not require that the machine-generated maps provide closed contours, and hence it is not straight-forward to convert these maps into segmentations. In addition, as was discussed above, measures used to assess the accuracy of boundary maps ignore the quality of edge fragments which do not have a good map in the ground truth, with random permutations of these edge fragments giving the same score.

A comparison of segmentation algorithms on the Sowerby data set is performed in [33]. Their ideas of considering additional algorithm characteristics, such as execution time, are interesting. However, they admit to not fully developing a list of or measures for these characteristics. In addition, it seems that the desired linear combination of multiple characteristics could be extremely difficult to establish.

Ge et al. [48] consider segmentation as a figure-ground denotation problem. They construct a 1023 image benchmark data set in which every image has an obvious, centered, dominant object, and generate unambiguous figure-ground labels. They also perform the questionable act of removing all color from the data set, reducing images to grayscale. Given their simplistic data set, they use the overlap score as their measure of segmentation accuracy. Although they compare the same segmentation algorithms we do, plus a few more, their evaluation scheme does not allow them to come to any concrete conclusions.

In [72], Malisiewicz and Efros examine whether segmentations can indeed correctly denote objects given multiple parameter choices for each algorithm. They also explore the question of whether taking the union of up to three regions can better denote objects than one region alone. Their results are discussed further in the next section.

6. Motivation for Our Object Recognition Framework

Our experimental evaluation of the maximum performance of the mean shift algorithm, the efficient graph-based algorithm and the hybrid algorithm, shown in Figure 3.9, has unfortunately shown that none of the algorithms consistently gives human-level segmentations. This is not entirely surprising given that to segment an image like a human, it would be necessary to have the notion of 'objects' that humans do. In [72], Malisiewicz and Efros come to the same conclusion regarding the first two algorithms, as well as normalized cuts segmentation. Through experiments on the MSRC 21-class data set (see Chapter 4 for a description of the data set), Malisiewicz and Efros show that none of the three algorithms they compared could consistently provide one region which properly corresponded to a human-denoted object. This problem is further aggravated by the fact that not even humans can agree on a 'correct' segmentation for an image, as evidenced by the examples from the Berkeley segmentation database given in Figure 3.1. Instead of being at the mercy of poor segmentation results, we do not use image segmentation as a black box pre-processing step, and instead ask how we can utilize its strengths and compensate for its weaknesses.

Our experiments have shown that the performance of a segmentation algorithm on a given image varies with its parameters, and no one set of parameters performs equally well on all images. As can be seen in our example segmentations, this variation is largely due to the granularity of the regions produced; producing too many small regions or too few large regions will both incur a penalty. There are advantages to this variation, however. Small regions allow us to capture small image structures which may be very discriminative, without overwhelming them by other image information. Regions which capture entire object parts generate a more holistic representation of the object, while regions which encompass both the object and background provide context which may be discriminative for certain objects. In order to capture this multi-scale information, and insure that object edges are indeed included among the set of segmentation edges, we choose to create multiple segmentations for each image and retain regions from all of the segmentations. In Chapter 6, we study the performance of our region descriptors and classification method using three segmentations generated by different parameters and the mean shift algorithm. Both our experiments and those of Malisiewicz and Efros have shown that mean shift segmentation gives promising results while retaining the variation necessary to capture multi-scale information. Malisiewicz and Efros have also shown that the combination of segmentations from multiple algorithms can in fact be even more effective than using one segmentation algorithm alone. This is an intuitive result given that different segmentation algorithms have different weaknesses. So, in Chapter 7 we expand our multiple segmentation framework to use the mean shift, efficient graph-based and normalized cuts algorithms, with an even larger set of parameters.

Our experiments have also shown that images are often over-segmented, due to complex objects with multiple different parts, or due to choosing suboptimal parameters for the segmentation algorithm. Fortunately, experiments in [72] show that the majority of the time the full object mask can be generated by the union of multiple regions. Allowing up to three regions from a ‘soup’ of regions generated by multiple segmentations to be merged together into an object mask achieved a high degree of similarity with the actual object location. This implies that we require methods of capturing information from both within a specific region, and from the regions around it. In Chapter 5, we introduce a region descriptor, the Region-based Context Feature (RCF), which incorporates image structure from in and *around* a region into a single region descriptor. In this manner, the RCF implicitly includes information from other object regions, as well as providing image context. In addition, in Chapter 8 we explicitly combine the information from neighboring regions through a random field formulation.

7. Contributions

Our contributions in this chapter include:

- A framework for evaluating and comparing multiple segmentation algorithms [87,117], including a statement of the required properties for a segmentation algorithm to be used as a black box, and a description of the experiments required to evaluate these properties.
- In joint work with Ranjith Unnikrishnan [116, 117], a new measure, the Normalized Probabilistic Rand Index, for evaluating the performance of a segmentation algorithm.
- An evaluation of four segmentation algorithms using our proposed framework [87,117].

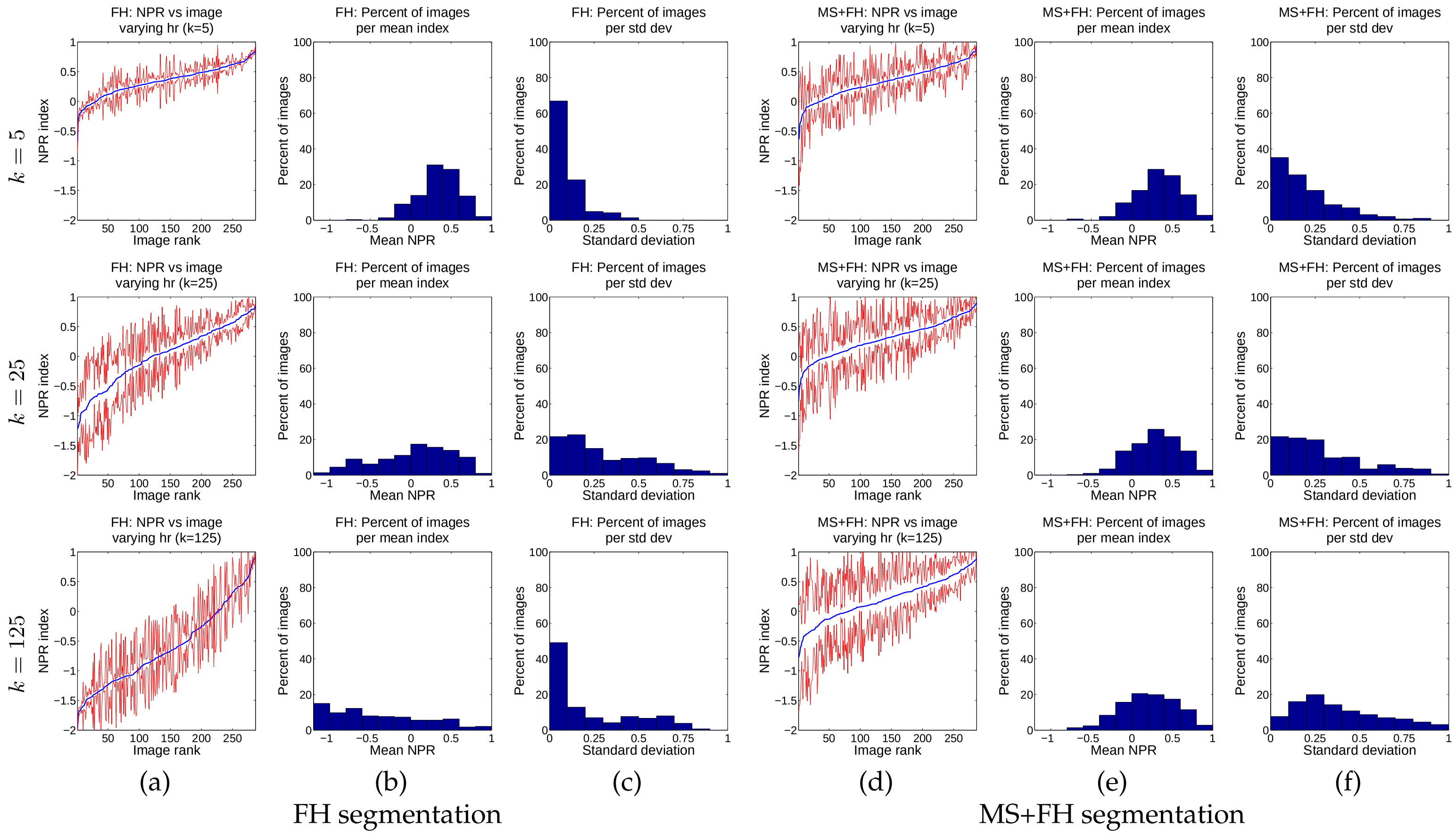


FIGURE 3.11. Mean NPR indices achieved on individual images over the parameter set $h_r = \{3, 7, 11, 15, 19, 23\}$ with a constant k. Results for the efficient graph-based segmentation system (FH) are shown in columns (a), (b) and (c), and results for the hybrid segmentation system (MS+FH) are shown in columns (d), (e) and (f). Columns (a) and (d) show the mean indices achieved on each image individually, ordered by increasing index, along with one standard deviation. Columns (b) and (e) show histograms of the means. Columns (c) and (f) show histograms of the standard deviations.

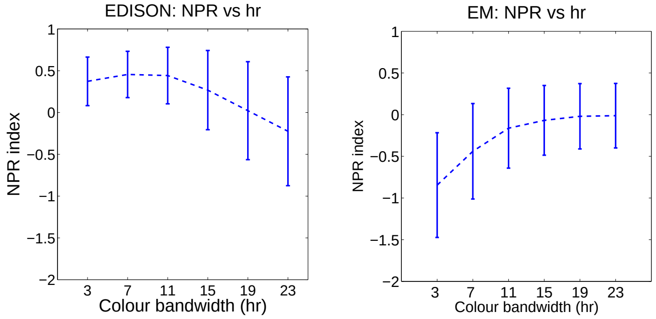
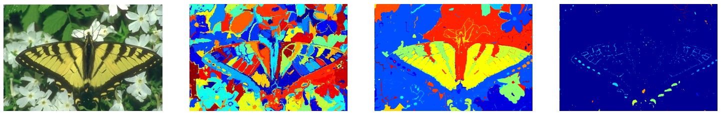
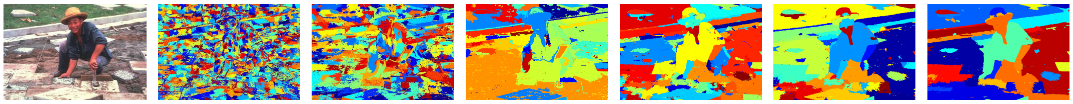


FIGURE 3.12. Mean NPR indices achieved on each color bandwidth (h_r) over the set of images, with one standard deviation. The left plot shows results for the EDISON segmentation system, and the right plot shows results for EM.



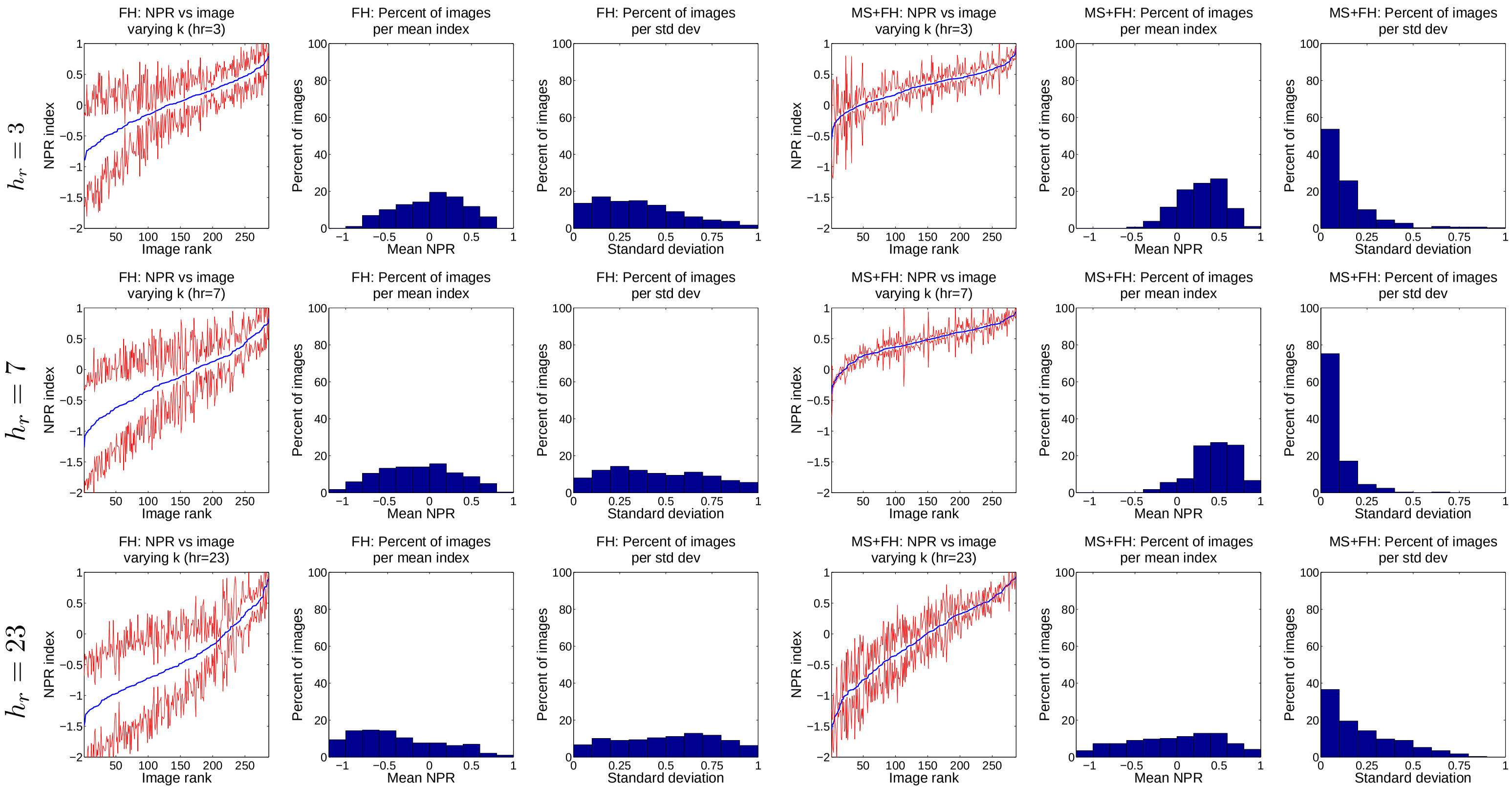
(a) (b) (c) (d)

FIGURE 3.13. Example of segmentation quality for different parameters using efficient graph-based segmentation: (a) Original image, (b)-(d) efficient graph-based segmentations using spatial normalizing factor $h_s = 7$, color normalizing factor $h_r = 7$ and k values $5, 25, 125$ respectively.



(a) (b) (c) (d) (e) (f) (g)

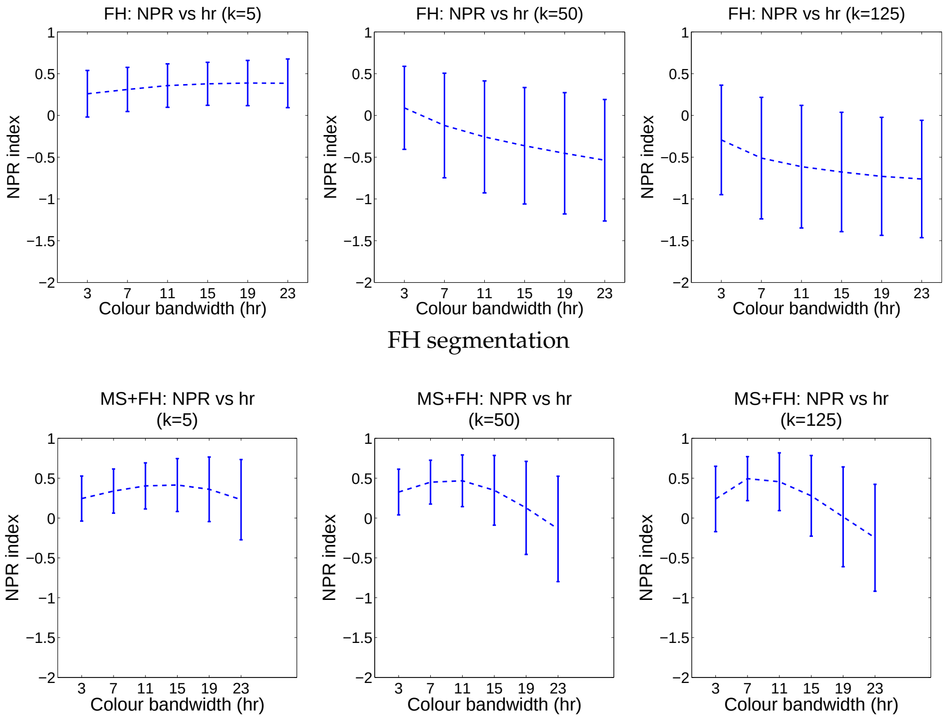
FIGURE 3.14. Example of segmentation quality for different parameters using a hybrid segmentation algorithm which first performs mean shift filtering and then efficient graph-based segmentation: (a) Original image, (b)-(g) segmentations using spatial bandwidth $h_s = 7$, and color bandwidth (h_r) and k value combinations $(3, 5)$, $(3, 25)$, $(3, 125)$, $(15, 5)$, $(15, 25)$, $(15, 125)$ respectively.



(a) (b) (c) (d) (e) (f)

FH segmentation MS+FH segmentation

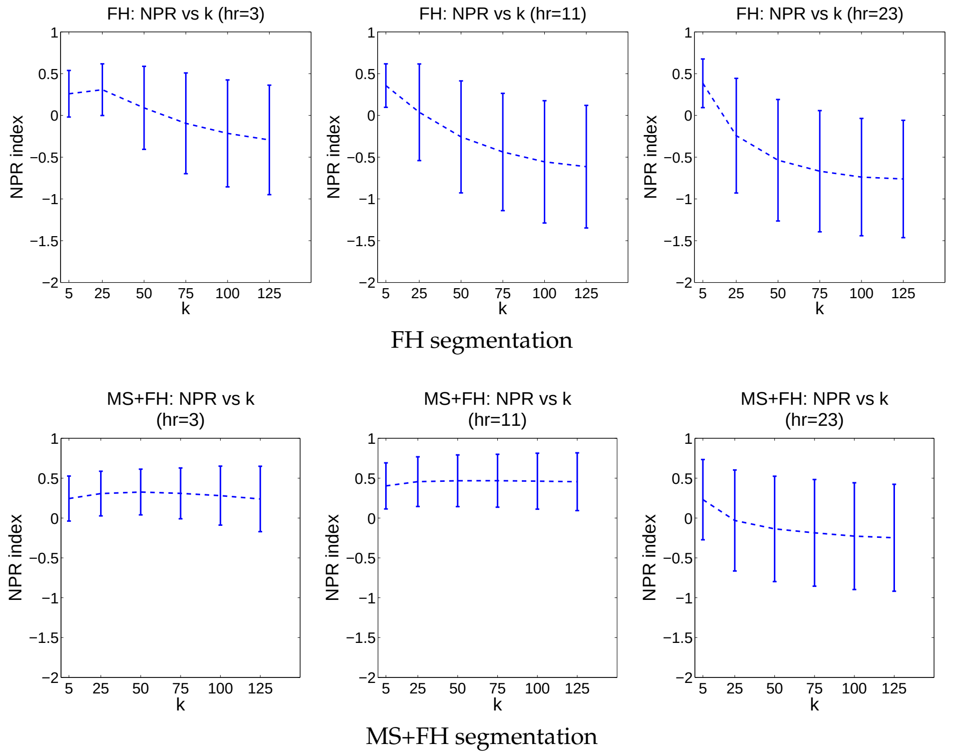
FIGURE 3.15. Mean NPR indices achieved on individual images over the parameter set $k = \{5, 25, 50, 75, 100, 125\}$ with a constant h_r. Results for the efficient graph-based segmentation system (FH) are shown in columns (a), (b) and (c), and results for the hybrid segmentation system (MS+FH) are shown in columns (d), (e) and (f). Columns (a) and (d) show the mean indices achieved on each image individually, ordered by increasing index, along with one standard deviation. Columns (b) and (e) show histograms of the means. Columns (c) and (f) show histograms of the standard deviations.



FH segmentation

MS+FH segmentation

FIGURE 3.16. Mean NPR indices on each color bandwidth $h_r = \{3, 7, 11, 15, 19, 23\}$ over the set of images. One plot is shown for each value of k. Experiments were run with $k = \{5, 25, 50, 75, 100, 125\}$, and we show a representative subsample of $k = \{5, 50, 125\}$. The plots in the top row show results achieved using the efficient graph-based segmentation (FH) system and the plots in the bottom row show results achieved using the hybrid segmentation (MS+FH) system.



FH segmentation

MS+FH segmentation

FIGURE 3.17. Mean NPR indices with $k = \{5, 25, 50, 75, 100, 125\}$ over the set of images. One plot is shown for each value of h_r. Experiments were run with $h_r = \{3, 7, 11, 15, 19, 23\}$, and we show a representative subsample of $h_r = \{3, 11, 23\}$. The plots in the top row show results achieved using the efficient graph-based segmentation (FH) system and the plots in the bottom row show results achieved using the hybrid segmentation (MS+FH) system.

CHAPTER 4

METHODOLOGY FOR OBJECT RECOGNITION AND OBJECT SEGMENTATION EVALUATION

THE previous chapters have elucidated the strengths and weaknesses of image segmentation. In the remainder of this thesis, we will incorporate image segmentation into a system for object recognition. For each system component, multiple results will be presented on a number of data sets. So for ease of reading, we begin here by introducing each data set and describing our methodology for evaluating object recognition and object segmentation.

1. Methodology

Our overall goal is to produce correct object recognition and object segmentation masks at the pixel level. In order to differentiate this from image classification or bounding box detection, we employ the strict measure of evaluating pixel classification accuracy. Specifically, for each image we will produce a probability or score map denoting the likelihood of each pixel belonging to each object class. Throughout this thesis, our score maps will be shown in the 'jet' color map, with dark blue representing the lowest score, through to dark red representing the highest score. An example of such a score map and the jet color map is given in Figure 4.1. From the maps over an entire testing data set, we compute the cumulative precision and recall for one object class at each threshold. Specifically:

$$\text{Recall}(\text{Object c}, \text{Threshold t}) = \frac{\text{Number of true positive pixels}}{\text{Number of object pixels}} \tag{4.1}$$

$$= \frac{\sum_M \sum_{p \in M} I(g(p) = c \text{ AND } s_c(p) \geq t)}{\sum_M \sum_{p \in M} I(g(p) = c)} \tag{4.2}$$

$$\text{Precision}(\text{Object c}, \text{Threshold t}) = \frac{\text{Number of true positive pixels}}{\text{Number of positively labeled pixels}} \tag{4.3}$$

$$= \frac{\sum_M \sum_{p \in M} I(g(p) = c \text{ AND } s_c(p) \geq t)}{\sum_M \sum_{p \in M} I(s_c(p) \geq t)} \tag{4.4}$$

where $I = 1$ if its argument is true and $I = 0$ otherwise, M is an image in the data set, p is a pixel, c is an object class, t is a threshold, $g(p)$ is the actual (ground truth) class of p, and $s_c(p)$ is the score for label c at pixel p. Notice that this is a sum over all of the pixels in the data set.

The recall and precision for all thresholds can be plotted to reveal the overall performance of the algorithm. Other users of this measure have included [113]. This measure requires ground truth labeling of the testing set with pixel accuracy. Note that some of the data sets have labeling errors due both to human inaccuracy and actual ambiguity.

Another frequently used measure of the amount of overlap between two regions is the ‘intersection over union’ measure:

$$\text{Overlap}(\text{Object c}, \text{Threshold t}) = \sum_M \frac{\sum_{p \in M} I(g(p) = c \text{ AND } s_c(p) \geq t)}{\sum_{p \in M} I(g(p) = c \text{ OR } s_c(p) \geq t)} \tag{4.5}$$

This measure is valid provided there is only one real object in each image. If there is more than one object in an image, some kind of assignment between blobs and objects must be performed, often in post-processing. Also, if there is no real object in an image then the score for that image is 0, regardless of how many pixels are labeled as belonging to the object class by the algorithm. Finally, note that this measure is an average over images, while the precision-recall measure is an average over all of the pixels in the data set.

The results we produce are pixel-level, in other words they compute the recall and precision of classifying each individual pixel in each image as object or background. This is in contrast to many of the existing evaluations for recognition and segmentation (which do not rely on human segmentations) that are presented as either image classification or object localization as bounding boxes or object centers, with a few exceptions such as [51, 94, 106, 113].

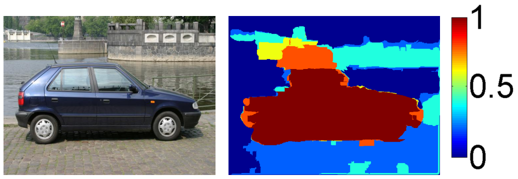


FIGURE 4.1. Example of a probability map produced by our algorithm. The left image is the original, and the middle image shows the probability of each pixel having the label 'car'. Probability and score maps are presented in the 'jet' color map, as shown in the rightmost image.

2. Data Sets

The following is a list of the data sets used throughout this thesis.

The Butterflies Data Set

- First used or created by: Lazebnik et al. in 2004 [65].
- Ground truth labels by: Pantofaru et al. in 2006 [86].
- Image source: Internet
- Unlabeled or 'void' pixels in the ground truth: No, all pixels are labeled.
- Image sizes: Variable, 150x172 to 900x647.
- Number of classes: 7 plus background
- Number of images: Total = 619, Training = 182, Testing = 437
- Training vs. test division as in the original publication: yes.
- Class names, sizes, and training set vs. testing set division:

	Total	Training	Testing
Class	images	images	images
Admiral butterflies	111	1-26	27-111
Machaon butterflies	83	1-26	27-83
Monarch butterflies - closed	74	1-26	27-74
Monarch butterflies - open	84	1-26	27-84
Peacock butterflies	134	1-26	27-134
Black swallowtail butterflies	42	1-26	27-42
Zebra butterflies	91	1-26	27-91

Examples of the Butterflies data set are given in Figure 4.2. The task is to differentiate the butterflies from each other, as well as from the background. The butterfly species are similar in many ways, increasing discrimination difficulty. It is especially difficult to differentiate between the two monarch classes which only differ in their pose. Also note that the images are different sizes, and the butterflies themselves vary widely in number and size.

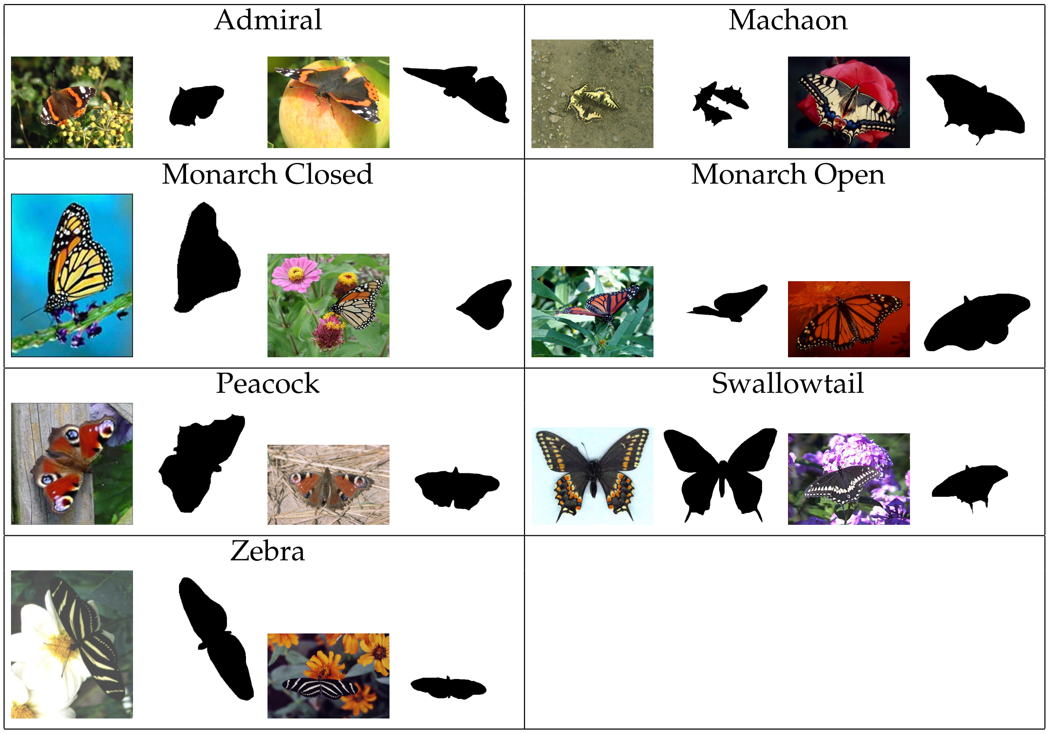


FIGURE 4.2. Examples of images and ground truth object segmentation masks from the butterflies data set. The data set contains 7 butterfly species: admiral, machaon, monarch closed, monarch open, peacock, swallowtail and zebra.

The Graz02 Bicycles Data Set

- First used or created by: Opelt et al. in 2005 [85].
- Ground truth labels by: Opelt et al. in 2005 [85].
- Image source: Opelt et al.
- Unlabeled or 'void' pixels in the ground truth: No, all pixels are labeled.
- Image sizes: 640x480.
- Number of classes: 2
- Number of images: Total = 600, Training = 300, Testing = 300
- Training vs. test division as in the original publication: yes.
- Class names, sizes, and training set vs. testing set division:

Class	Total images	Training images	Testing images
Bikes	300	Odd numbers	Even numbers
Background	300	Odd numbers	Even numbers

Examples of the Graz02 Bicycles data set are given in Figure 4.3. The task for this dataset is to differentiate between the bicycles and the background. Bicycles are a particularly difficult object for our methods because they are 'wiry' and better defined by shape than texture. Most of the discriminative bicycle features lie on the object boundary, and hence are unstable with changing backgrounds.

Notice that the size of the objects in this data set varies drastically, as do the pose, level of occlusion, background and lighting.

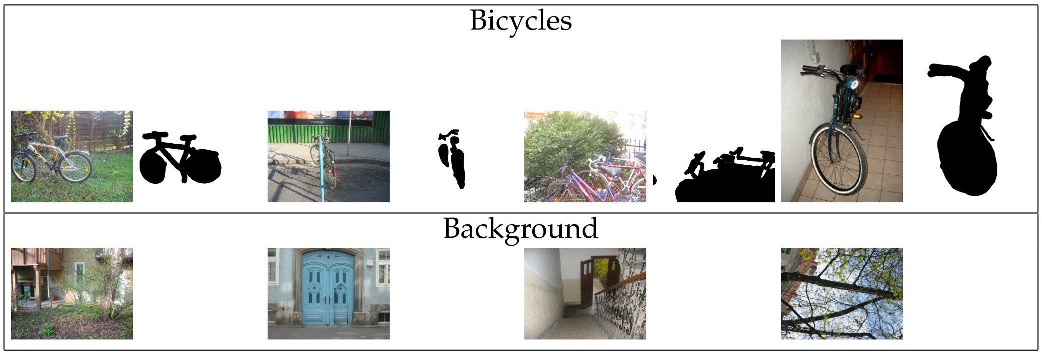


FIGURE 4.3. Examples of images and ground truth object segmentation masks from the Graz02 Bicycles data set. The data set contains two classes, bicycles and background.

The Corel Image Database Leopards, Jaguars and Cheetahs Data Set

- First used or created by: Corel Image Database.
- Ground truth labels by: Pantofaru et al. in 2006 [86].
- Image source: Corel Image Database.
- Unlabeled or 'void' pixels in the ground truth: No, all pixels are labeled.
- Image sizes: 640x480 in portrait or landscape orientation.
- Number of classes: 2
- Number of images: Total = 200, Training = 101, Testing = 99
- Training vs. test division as in the original publication: not applicable.
- Class names, sizes, and training set vs. testing set division:

Class	Total images	# Training images	# Testing images
Cheetahs, Leopards and Jaguars folder	100	51	49
Backyard Wildlife	100	50	50

In this document, we will call this data set the 'Spotted Cats' data set. Examples of the Spotted Cats data set are given in Figure 4.4. The task for this dataset is to differentiate between the cats and the background. The difficulty in correctly identifying and segmenting cats lies in their extreme deformability. Note also the variation in size, occlusion, number of cats, background and lighting within the images.

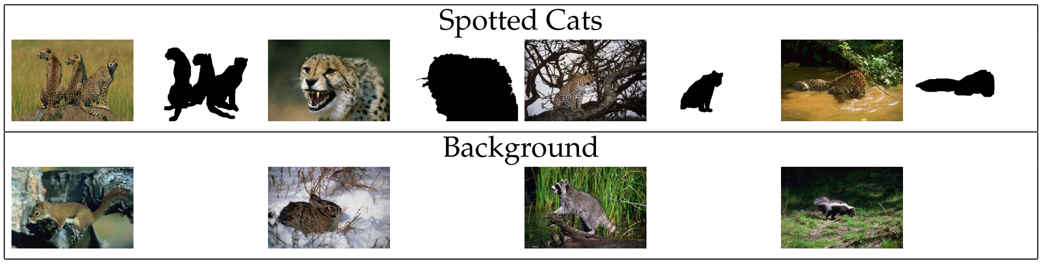


FIGURE 4.4. Examples of images and ground truth object segmentation masks from the Spotted Cats data set. The data set contains two classes, spotted wildcats and background images.

PASCAL Visual Object Challenge (VOC) 2006 Cars Data Set

- First used or created by: Everingham et al. in 2006 [31].
- Ground truth labels by: Pantofaru et al. in 2007 [88].
- Image source: Microsoft Research Cambridge (MSRC) employees and flickr [42].
- Unlabeled or 'void' pixels in the ground truth: No, all pixels are labeled.
- Image sizes: Varying from 130x120 to 640x480.
- Number of classes: 2
- Number of images: Total = 1606, Training = 1062, Testing = 544
- Training vs. test division as in the original publication: The positive images are as in the original publication, the negative images are a subset of the original negative images.
- Class names, sizes, and training set vs. testing set division:

Class	Total images	Training images	Testing images
Cars	1097	553	544
Background	509	509	0

  - The Cars images are all 553 images from the PASCAL training plus validation sets for training, all 544 images from the testing set for testing.
  - The Background set was created by combining the images from the Bus and Bicycle training and validation sets which do not contain cars.

Examples of the PASCAL VOC2006 Cars data set are given in Figure 4.5. The task for this dataset is to differentiate between the cars and the background. The PASCAL VOC2006 dataset is extremely challenging, with a large variety of object sizes, poses, lighting conditions, and occlusion. A large number of the images contain multiple cars. Cars themselves are additionally challenging for our methods as they contain large regions with little or no texture. Since their color can vary widely, correctly identifying these textureless regions depends on the use of features in neighboring regions.

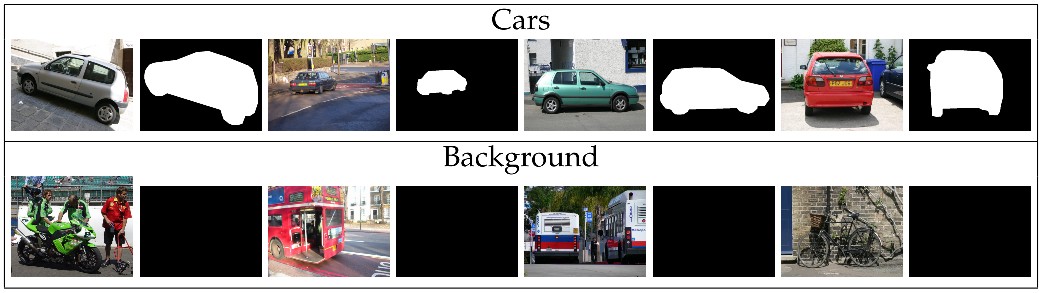


FIGURE 4.5. Examples of images and ground truth object segmentation masks from the PASCAL VOC2006 Cars data set. The data set contains two classes, cars and background images.

The Microsoft Research Cambridge (MSRC) 21-Class Data Set

- First used or created by: Shotton et al. in 2006 [105,106].
- Ground truth labels by: Shotton et al. in 2006 [105,106].
- Image source: Unknown. The images were likely taken by Shotten et al.
- Unlabeled or 'void' pixels in the ground truth: Yes, many of the pixels on object boundaries and in non-object regions. Shown in black in the images.
- Image sizes: Varying from 213x320 to 240x320 in either orientation.
- Number of classes: 21
- Number of images: Total = 591, Training = 335, Testing = 256
- Training vs. test division as in the original publication: Yes.
- Class names, sizes, and training set vs. testing set division:

Class	Total	Train	Test	Class	Total	Train	Test
Building	153	84	69	Flower	35	20	15
Grass	213	122	91	Sign	31	18	13
Tree	134	81	53	Bird	38	21	17
Cow	45	26	19	Book	35	20	15
Sheep	35	19	16	Chair	30	17	13
Sky	158	90	68	Road	151	87	64
Aeroplane	30	17	13	Cat	24	14	10
Water	75	42	33	Dog	30	17	13
Face	60	33	27	Body	68	38	30
Car	44	25	19	Boat	33	18	15
Bike	32	18	14				

Examples of the MSRC 21-class data set are given in Figure 4.6. The task for this dataset is to differentiate between the classes. Background pixels are labeled 'void' and are not considered in the evaluation.

A central difficulty in classifying the objects in this data set is that they vary in their defining characteristics. For example, grass or water are best characterized

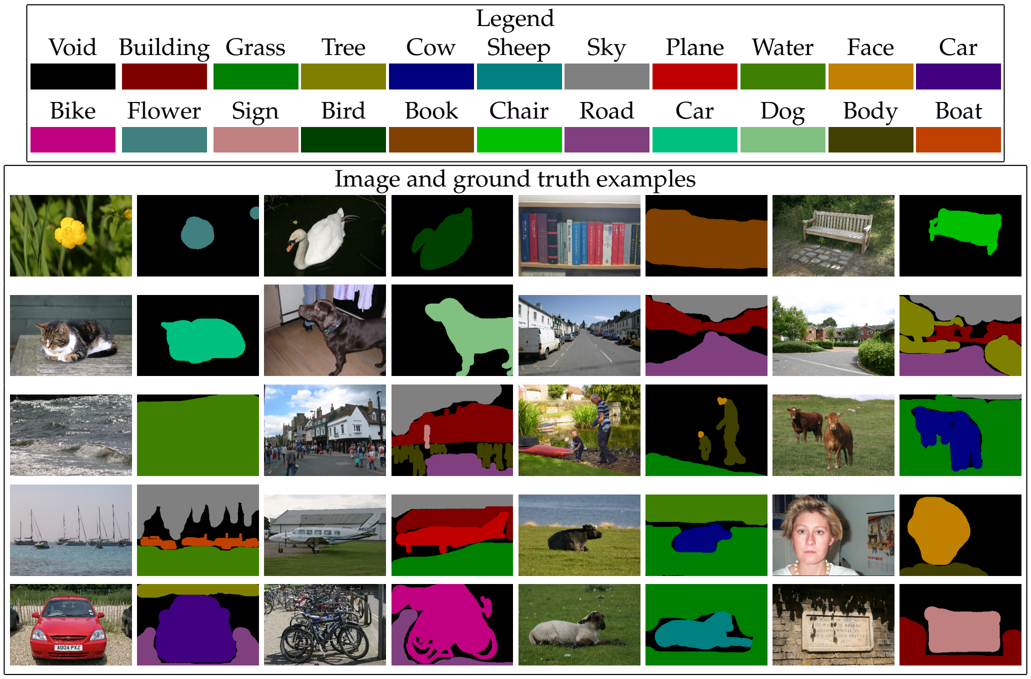


FIGURE 4.6. Examples of images and ground truth object segmentation masks from the Microsoft Research Cambridge (MSRC) 21-class (v2) data set. The data set contains 21 classes as listed in the color legend at top. Black pixels are unlabeled, or 'void'.

by their color or texture while cars are characterized by their parts. This makes the use of multiple features crucial for this dataset.

Training is also difficult on this dataset. Many of the pixels on the object boundaries have void labels, making supervised training of shape features difficult. Weakly supervised training is also difficult due to the frequent co-occurrence of objects, such as faces and bodies. Finally, there are few training images for classes such as 'cat'.

PASCAL Visual Object Challenge (VOC) 2007 Segmentation Challenge Data Set

- First used or created by: Everingham et al. in 2007 [32].
- Ground truth labels by: Everingham et al. in 2007 [32].
- Image source: MSRC employees and flickr [42].
- Unlabeled or 'void' pixels in the ground truth: Yes, many of the pixels on object boundaries and some pixels in non-object regions. Shown in beige in the images.
- Image sizes: Variable. Heights range from 176 to 500, widths range from 174 to 500. All image dimensions were reduced to 80% of their original values.
- Number of classes: 20 plus background.

- Number of images: Total = 632, Training = 422, Testing = 210
- Training vs. test split as in the original publication: Yes.
- Classes names, sizes, and training set vs. testing set division:

Class	Total	Train	Test	Class	Total	Train	Test
Aeroplane	40	25	15	Dining table	45	31	14
Bicycle	32	21	11	Dog	44	31	13
Bird	38	26	12	Horse	43	32	11
Boat	33	20	13	Motorbike	39	26	13
Bottle	43	30	13	Person	263	171	92
Bus	37	25	12	Potted plant	45	34	11
Car	55	31	24	Sheep	31	21	10
Cat	44	30	14	Sofa	45	30	15
Chair	67	46	21	Train	39	23	16
Cow	31	21	10	Tv/monitor	50	33	17

Examples of the PASCAL VOC2007 Segmentation Challenge data set are given in Figure 4.7. The task for this dataset is to differentiate between the different classes as well as the background.

The variability in object size, pose, color, number, occlusion, etc. makes this a challenging data set. In addition, the images are often of questionable quality and vary in white balance and brightness.

The background class in this data set is large as it includes all of the texture-based object classes such as water, grass, etc. In addition to the variability in the other classes, the variability in the background increases discrimination complexity. The small size of each object's training set further increases difficulty.

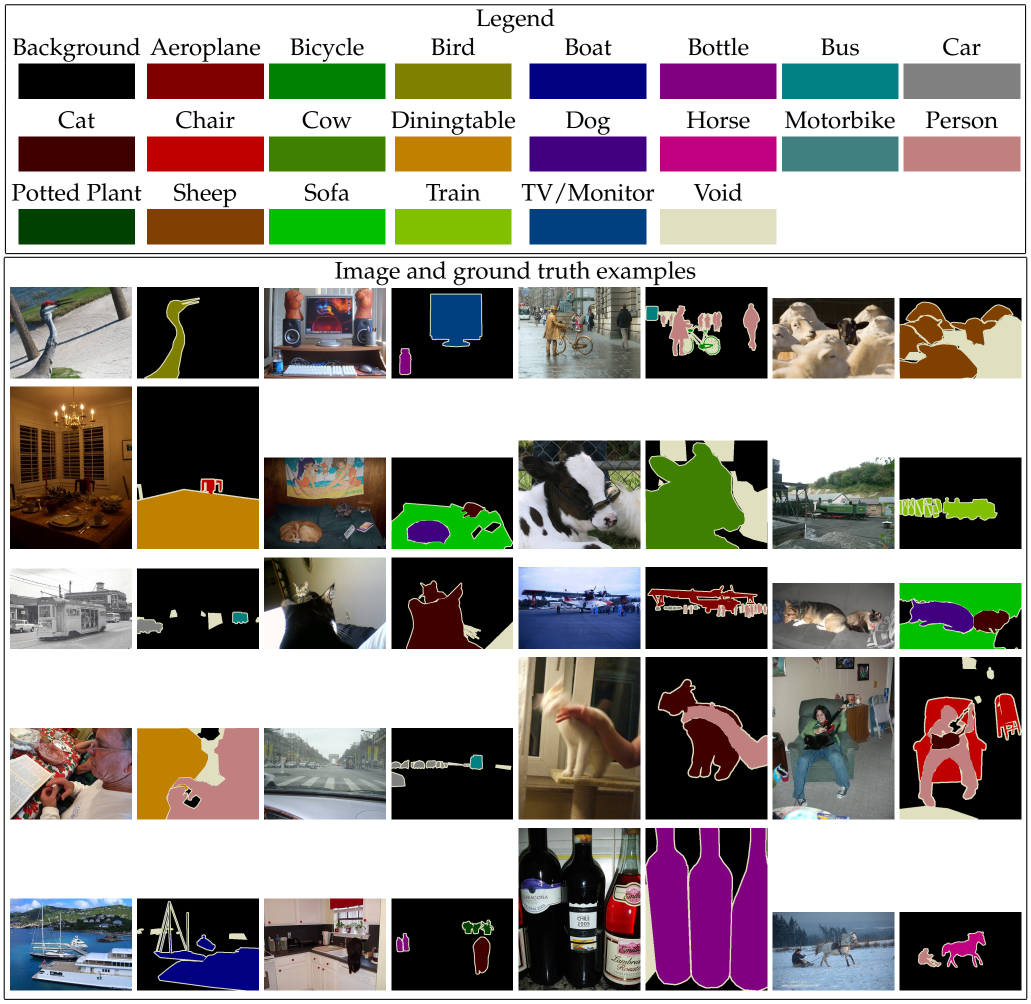


FIGURE 4.7. Examples of images and ground truth object segmentation masks from the from the PASCAL VOC2007 segmentation challenge data set. The data set contains 20 classes plus a background class as listed in the color legend at top. Beige pixels are unlabeled, or 'void'.

CHAPTER 5

DESCRIBING REGIONS

THE first requirement of any object recognition system is an image description which converts the image into features appropriate for a classifier. There has been a vast amount of work on representing images, with methods categorized by the portion of the image they describe. The most local descriptions interpret pixel information such as position, intensity or color in one of many color spaces. Patch-based methods consider image regions of a parameterized shape centered at a pixel. These patches may be a square, rectangle, circle, ellipse, or even a specific object shape, and their size can vary from one pixel to the size of the image, however their parametric family is known a priori. These larger patches allow for higher order statistics of their member pixels' characteristics to be computed, such as the mean color or a histogram of edge orientations. The most global descriptions consider information from the entire image, accumulating pixel or patch features such as in the spatial pyramid approach in [66], or the 'gist' descriptor [84]. However, there has been little effort put into creating a proper description of irregularly-shaped regions arising from unsupervised segmentation, with most approaches being ad hoc adaptations of patch-based descriptors.

In contrast to image patches, segmentation regions have unpredictable shapes and sizes, each one different, as shown in Figure 5.2. As we discussed in Chapter 3, the tendency of a region's shape and size to be regular or irregular, small or large, is influenced both by the segmentation algorithm and the underlying image data.

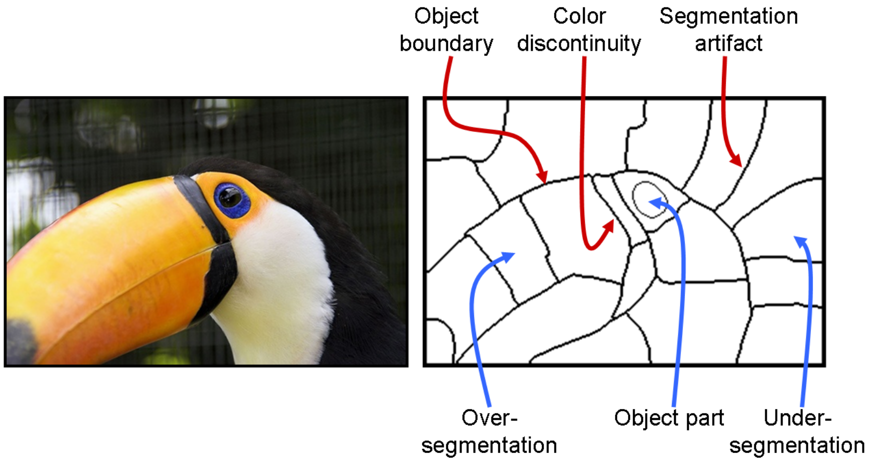


FIGURE 5.1. Illustration of the different types of segmentation-generated regions (in blue) and region boundaries (in red). Ideally, region boundaries coincide with object boundaries. However, some region boundaries coincide with image structures such as color discontinuities, or are simply artifacts of the segmentation algorithm and do not correspond to any image structure. The regions themselves will ideally contain a full object part, such as the eye. However, they may over-segment an object, such as breaking the beak into pieces. Regions may also under-segment an image by including pixels from multiple objects, such as the parrot's black feathers and the background.

This results in three kinds of regions and three kinds of region boundaries, as illustrated in Figure 5.1. Regions will ideally correspond to a full object part, such as the parrot's eye. However, they may over-segment an object, such as breaking the beak into pieces. Regions may also under-segment an image by including pixels from multiple objects, such as the parrot's black feathers and the background. The region boundaries will ideally agree with object boundaries, but they may also reflect discontinuities in the image's feature space but not actual object boundaries, or they may simply be artifacts of the segmentation algorithm. Unfortunately, it is rare to find regions that completely denote an object. To complicate matters, the object classes we wish to recognize are often highly deformable, and so even regions which agree with human-defined parts may not be the same shape from one object instance to the next. Consider the leg of the cheetah in Figure 5.2. When the cheetah is sitting, its leg is bent and against the body, however when it is standing its leg may be straight and not overlap the body at all. Such situations encourage us

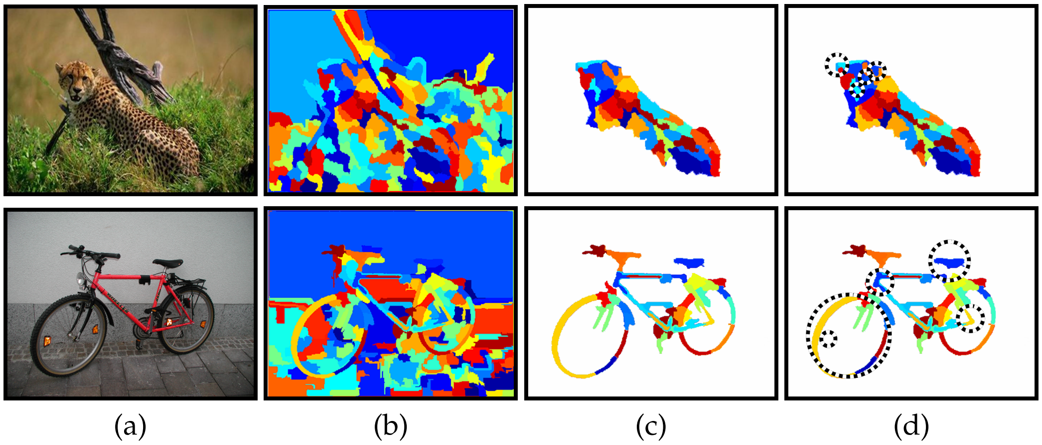


FIGURE 5.2. Examples of unsupervised image segmentations, interest points and object masks. The original images (a), an unsupervised segmentation for each image (b), those regions which lie on the object (c), and some interest point patches (circles) (d). By combining the regions on the object, we can form an object mask. The interest points, however, do not facilitate object mask creation due to their fixed shape.

to model the features and spatial relationships in and around regions in a flexible manner.

We also require our region representations to be amenable to weakly supervised training. Most representations of irregular object shape require that the training data contain accurate object masks. Since generating ground-truth object masks on large training sets is prohibitively expensive, we wish to avoid this requirement.

In this chapter, we propose multiple solutions to the region representation problem. The first group of representations is texture-based and represents information only within each region. These are inspired by traditional bag-of-words patch representations. As our experiments in Chapter 6 will show, these representations are excellent for certain object parts such as cheetah bodies, whose texture is extremely discriminative.

On the other hand, there are object parts such as cheetah heads which do not contain repetitive texture. These are best represented by more discriminative descriptors. In addition, it may be useful to represent image structure on the region boundaries, such as the tips of the ears, or in neighboring regions as was discussed our segmentation evaluation in Chapter 3. To tackle these issues, we propose a new region descriptor, the Region-based Context Feature (RCF), which aggregates the image structure in and around a region in a principled manner.

1. Related work

The following describes related work in segmentation region or object mask representations. Although there are multiple representations for image patches as well, we do not discuss those here.

One family of region descriptors attempts to encapsulate the shape of an object silhouette. Shape context [8] and shapemes [81] model object silhouettes by placing a set of markers around the object boundary and accumulating their relative locations in a histogram. However, they assume that the silhouette is known and defines the entire object at once, neither of which is the case in our problem. They also require a fixed shape for matching, which does not apply to the deformable objects we wish to model. Kumar et al. [60] also use shape context, along with histograms of quantized intensity patches, to represent possible parts in a Layered Pictorial Structure model. Given that their objects of interest are side views of standing horses and cows, they do not need to cope with large deformations.

In [108], Todorovic and Ahuja model regions using the mean and standard deviation of the grayscale values within the region, along with the entropy of a histogram which models the amount of the region's mass in each of k equal 'pie slices' centered at the region's center of mass. They also model some spatial structure by combining region information with relative weights defined by the distance between region centers of mass. Matching both the internal and external features is reliant upon regions having repeatable shape between segmentations of different object class exemplars. Their region representations in [5] have similar issues. The danger of relying on a region's center of mass can be seen in Figure 5.3. Compact regions produce reasonable centroids, however the quality varies with region shape, with the worst centroids being outside the regions altogether.

Borenstein and Malik [11] and Levin and Weiss [68], use binary shape templates as a top-down guide for bottom-up superpixel clustering. Their approach

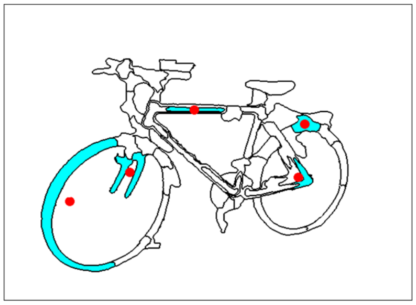


FIGURE 5.3. Examples of the varying quality of region centroids. For a compact region such as the one on the rear wheel, the centroid is within the region and close to the region's pixels. As the shape elongates, such as that of the region on the bike frame, the centroid may become quite far from some of the region pixels. Most drastically, for non-convex regions such as those on the front wheel, the centroid may be outside the region altogether.

requires a small set of object poses to keep the shape vocabulary tractable. Ramanan [91] learns shape models by a 2-cut of extended object detection bounding boxes based on color histograms. This approach works best for objects which fill the majority of their bounding boxes (results are presented on pedestrians, faces and cars), and have color histograms distinctive from their backgrounds. However, it tends to lose object parts whose color is more similar to the background than the overall object histogram. Marszalek and Schmid [73] use SIFT [69] vocabulary words to boost the strength of features on an object mask, and in [74] they use SIFT features to match and align full object masks. Shotten et al. [106] use the relative position between image structures and feature patches. All of these approaches require fully supervised training data to learn the shape templates/masks and implicitly assume that the objects are fairly rigid.

A second family of descriptors model the interior contents of a region. In [94], Russell et al. represent a region by a histogram of the contained quantized SIFT descriptors. This representation adequately models the information within a region, but it does not consider the information near a region. Since they make an assumption that they can create a region which contains their entire object of interest, perhaps information outside of the region would be superfluous. We, however, believe that the assumption of one region per object is overly optimistic (see Chapter 7), and hence information from the area surrounding a region could be useful.

This becomes even more important given that the boundaries of a segmentation region are unlikely to accurately denote the boundaries of an object or part. Additionally, SIFT descriptors are extremely discriminative, which makes them useful for matching objects with distinct parts, but too specific to match objects with repeated textures.

The Blobworld system [18,19] represents regions by both color histograms and the mean texture contrast and anisotropy. Since the segmentation method is EM, using a mean to represent texture is natural. However, the regions in the Blobworld system are only used for matching and database retrieval, not for object class recognition or localization. There is also user in the loop who decides which features are important to their particular query, so a relative feature weighting need not be learned by the system.

Tu et al. [111] attempt to perform image parsing by combining specific object models for text and faces with more generic image segmentation regions. For the faces, a model of the boundary pixels is used, and for text a boundary spline model is used. These models, however, are only feasible with prior knowledge about the object class and fully supervised training data, which we do not have. Generic segmentation regions are modeled using a Gaussian shading assumption, an intensity histogram, and a generic shading model using a quadratic form. The use of these generic models is for segmentation only, not for recognition, and hence they do not need to be discriminative.

Hoiem et al. [52] describe the interior contents of a region using a slew of features including, but not limited to, image position, region area, the mean response over the region to a subset of the texton filterbank that we also use, as well as a histogram of the filters eliciting maximal response over the region.

Cao and Fei-Fei [16] describe the interior of a region by the average color and texture, as well as a list of quantized SIFT descriptors of interest points. Since their interest points are sparse, they cannot guarantee that a region will contain any. Once again, they only model information within the region.

Although not applied to regions, the work of Savarese et al. [96] introduces the concept of correlatons which have some similarity to the RCF descriptors to be introduced in this section. Earlier color correlograms [54] produced histograms of the correlation between colors as a function of distance. Savarese et al. replace color with visual words and quantize the resulting correlograms to produce correlatons. These correlatons are used in conjunction with a bag-of-words representation for image classification. The similarity with our approach is in the distance-dependent histograms of visual words.

2. Texture descriptors

Inspired by the texton histogram representation [71], our first approach to region representation is to model a region's interior texture. The texture of a region can be an excellent classification cue for objects with repetitive, distinctive textures such as zebras of cheetahs. We will examine two ways of representing texture with subtle, but as our experiments will show in Chapter 6, relevant differences. We will also show that repetitive texture is complementary to our novel feature, the RCF.

2.1. TM: Mode of the texton histograms in a region

Consider an image region extracted through mean shift-based segmentation [25] performed using features which include a 30-dimensional texton histogram [71], as we described in Chapter 2. Mean shift filtering outputs the mode of the features within each region, thus a natural texture representation is the 30-dimensional texton histogram part of the region mode, (TM) [86, 88]. If a different segmentation algorithm were to be used, say normalized cuts [104], the mean of the texton histograms within a region could be substituted for the mode. In order to improve generalization, the texton modes from all of the training regions are clustered using K-means clustering into a vocabulary of size $K_{TM} \in \{50, 300\}$. Each mode is assigned to the cluster with the closest centroid resulting in the texton mode feature, TMF. Figure 5.4 shows examples of regions which are clustered together into the same TMF by this process. The image on the left shows regions from potentially different images containing 'spotted cat' textures. Note that all of the textures are

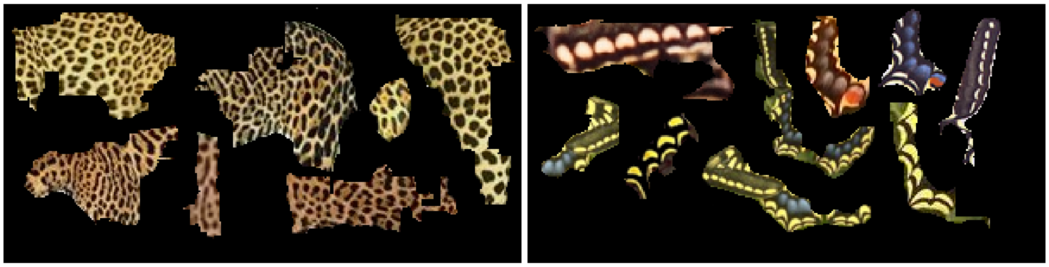


FIGURE 5.4. Examples of image regions within discriminative texture clusters. The feature cluster on the left is one of the top (best ranked) Spotted Cat texture features, while the one on the right is one of the top Machaon Butterfly texture features.

roughly the same scale. The image on the right shows regions from images of machaon butterflies with similar textures. In Chapter 6, we will discuss how to score these texton mode clusters according to their discriminative power, and it will turn out that the two clusters shown here are among the most discriminative for their object classes.

2.2. TR: Histogram of the textons in a region

When performing segmentation, we need to use a low-dimensional texton space (30 dimensions) to facilitate smooth grouping. However, for region classification a more discriminative texton vocabulary could potentially lead to more discriminative descriptors of region texture. The second texture representation, then, requires computing a new texton vocabulary whose size, $K_{TEX} \in \{30, 200, 1000\}$, is independent of that used for segmentation. The texton words in a segmentation region are then accumulated to form a new texton histogram (TR). Note that this texton histogram is computed over the region, not over square windows as in the previous method. These histograms can be clustered to create a different vocabulary of region descriptors of size $K_{TR} \in \{50, 300\}$.

Both texture descriptors can adapt to the data-driven shape of a region. If a region is 'fat', its quadratic number of interior pixels will vastly outnumber its linear number of boundary pixels, and hence the texture mode will reflect the interior texture. If, on the other hand, the region has long and 'skinny' parts, the interior pixels will not outnumber the boundary pixels to such a large extent and the texture mode histogram will reflect the boundary texture. The spotted cat's body in Figure 5.2 has regions in which the interior pixels outnumber the exterior pixels, and hence the spotted texture is modeled. Conversely, the bike frame has long regions with a small number of interior pixels, and hence the linear texture is modeled. We will encounter this phenomenon in Chapter 6.

3. Region-based Context Features (RCF)

The data-driven shape of a segmentation region is useful for specifying the spatial support of texture features, but it can also work against us. Consider the images in Figure 5.2. The regions on the body of the cheetah will be easily detected using texture, but what about the regions on the head? These regions have no texture in their interior, however they do have discriminative features near their boundaries, such as the shape of the ear, the presence of an eye, or the shape of the mouth. Similarly for the bike, the angles of the frame are on the boundaries of segmentation regions, and since they are unique they will be obscured by the texture representations. For these sorts of features we need a descriptor that encapsulates their discriminative nature and a spatial support that ventures outside region boundaries. In addition to shape, our discussion in Chapter 3 concluded that context may be vital to properly identifying object regions. We found that variations in lighting and pose, as well as object inhomogeneity, can lead to objects being divided into multiple regions. Thus, information from neighboring regions can be helpful in obtaining a more holistic object view.

There has been a significant amount of work done on the inclusion of shape, spatial information and context information into object models. Shape information relates the various parts of the object to each other, and has been explored through the silhouette-based approaches mentioned in the related work section, and geometric relations in part-based models [37, 127]. Spatial and context models may also relate the object to the other parts of the scene, be they local or global, for example [83]. In our problem, the objects may be highly deformable or seen from highly varying angles, so it is beneficial not to limit the model to shape or spatial relationships which enforce a strict topology among the parts. However, it is useful to model a notion of proximity among image structures. This allows us to identify regions which may not have a discriminative texture themselves, but which have close proximity to discriminative structures. The regions in Figure 5.8 do not have a discriminative texture (in fact they have no texture at all), but they do lie near to the dots which line the edge of the butterfly's wing. To better describe such region categories, we have introduced the concept of Region-based Context Features (RCF) [86].

How far outside our region should we go to capture shape and context information? For a patch-based approach we could double the size of the patch, say, and capture all of the features in its larger spatial support. Regions pose a more difficult problem, however. As we saw in Figure 5.1, region boundaries result from a number of effects, leading to regions which may not fully capture object parts and whose sizes may not be repeatable from one object instance to the next. Thus simply increasing the area of a region is not a stable way to determine feature support.

Interest points, however, have been shown to have much more repeatable scale than regions. Using either an interest point detector, such as the Harris detector, and a scale selection method such as a Laplacian pyramid, or by using dense interest points on a fixed grid with a fixed set of scales per point, we can denote a patch of fixed shape and describe the image structure it encloses. The difficulty with using interest points and patches is that there is no clear way to determine the object mask. In Figure 5.5, the circular patches denote examples of interest points (at the patch centers) and their extent. The patches do capture image structure, but neither drawing a bounding box around the entire set as in [2,67,73], nor around each patch individually such as in [40,107], provides a satisfactory mask. Both methods include multiple non-object pixels while at the same time omitting other object pixels. Instead we choose to combine the spatial support of segmentation regions for mask-building and interest point patches for their discriminative power.

We describe local image patches using the popular 128-dimensional SIFT descriptor [69]. The locations of the patches may be sparse, as determined by a local

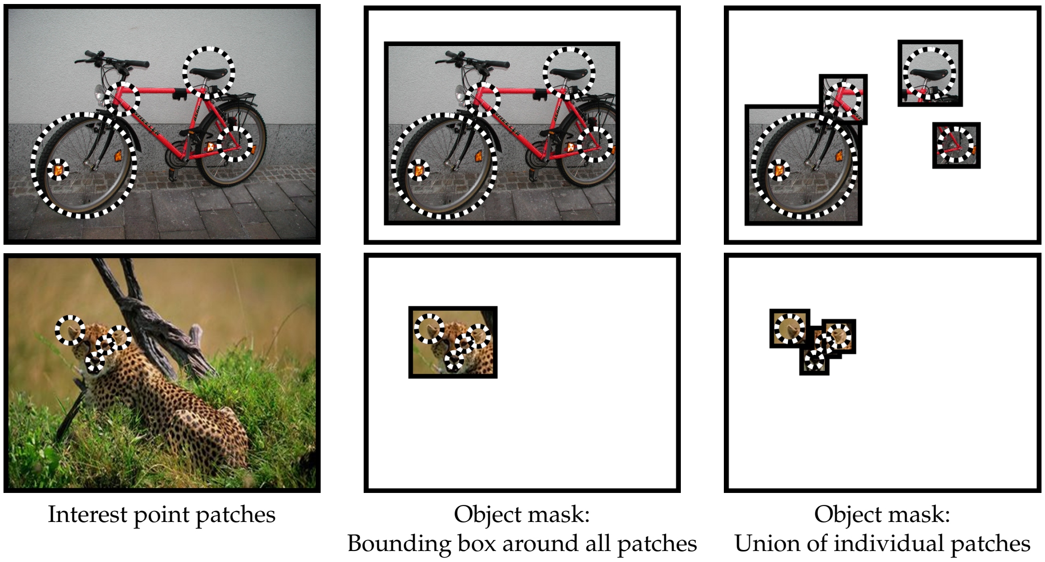


FIGURE 5.5. Ineffective methods for converting interest point patches into object masks. The left column shows some possible discriminative interest point patches. The middle column shows the results of generating an object mask by taking the bounding box around all of the interest point patches, while the right column shows the results of generating the mask by taking the union of all the separate patches. Neither method provides accurate object masks.

interest point operator and scale selection, or densely arranged over the image and in scale space. Let the set of points (patch centers) in one image be $\mathcal{P} = \{p_i\}_{i=1}^{N_P}$, with scales $\{\sigma_i\}$ and local descriptors $\{d_i\}$. Clustering the set of descriptors from all of the training images produces a vocabulary of local descriptor words $\mathcal{W}$ of size N_W. Let w_i be the nearest neighbor word to descriptor d_i. We use the scales $\{\sigma_i\}$ of the patches to define proximity to regions. The idea is to create a histogram for each region of the local words whose centers are at most $k\sigma_i$ pixels away from the region. Then we can append these k-histograms together, weighted inversely proportionally to their k values, to create a set of Region-based Context Histograms (RCH).

More specifically, let r be a region in the image, and p_j pixels within the region. Let h_k be the k-histogram for the region r with N_W bins, one for each word. We build the histograms as follows:

$$h_k(w) = |\{i \mid w = w_i, (k-1)\sigma_i < \min_{p_j \in r}\{d(p_j(r), p_i)\} \leq k\sigma_i\}|. \tag{5.1}$$

where $d(*, *)$ is the Euclidean distance between the pixel locations. In our implementation, $k \in \{1, 2\}$ and the histogram for $k = 1$ includes the points with distance $k = 0$ (points which are within the region itself). Each h_k is then normalized. For larger k, the histogram contains points which are farther away from, and less related to, the region. These distant points are accumulated over an area which grows quadratically with k (while k grows linearly). The h_k are weighted inversely proportionally to k, which is reminiscent of the scaling in the pyramid kernel introduced in [49]. For our experiments, we use weights of $0.5^{(k-1)}$. The (weighted) $\{h_k\}$ are concatenated to get a final feature $RCH = [h_1, h_2, \ldots, h_K]$.

The method for building an RCH is summarized in Figure 5.6. The figure shows a region with homogeneous texture, in red, on an image of a zebra. There are several interest points centered at each circle or pair of concentric circles. The descriptors for these interest points are quantized into three SIFT 'words', denoted as yellow, green and blue circles. For each interest point, the number of concentric circles around it denotes k, and the radius of the inner circle denotes σ_i. In other words, one concentric circle implies the point is $1\sigma_i$ from the region, while two concentric circles imply the distance is $2\sigma_i$. The top histogram shows the un-normalized RCH, while the bottom one shows the final normalized and reweighted RCH.

Figure 5.7 shows the advantage of using the RCH approach to combining regions and interest points compared to an approach based on the region area. In image (a) in the figure, the interior red region is the same as the region in Figure 5.6. The colored dots show the locations of some of the interest points in Figure 5.6 without their spatial extents. If we increase the area of the region by a fixed percentage to the exterior red region, the shown interest points would be captured. Suppose that the region was instead subdivided into smaller regions, one of which is shown in (b). If the area of the region is increased by the same percentage, the interest points are no longer captured. Images (c) and (d) show the same region subdivision but with the RCH approach to combining regions and interest points. Since the RCH is based on the interest point scale and not the region area, many of the interest points are still captured with the smaller region. Given the instability of

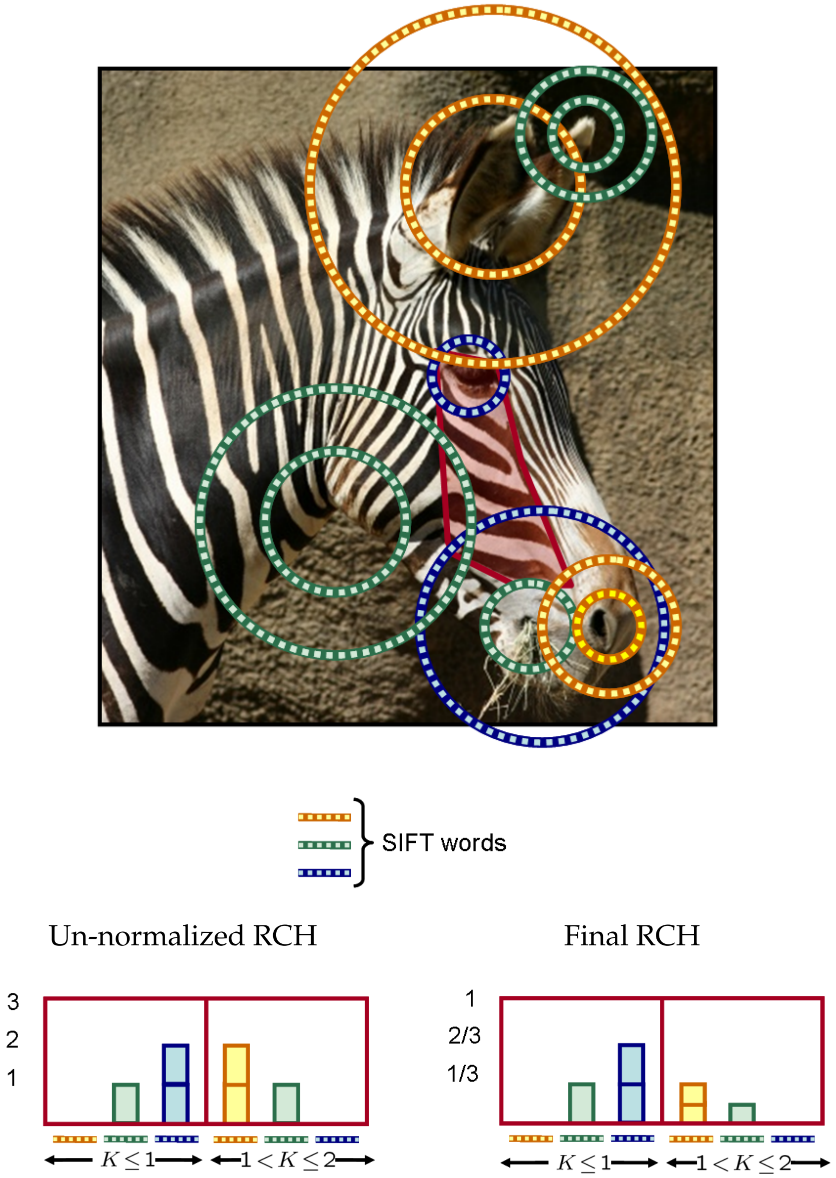


FIGURE 5.6. Illustration of the construction of an RCH. The top image shows a region with homogeneous texture, in red, on an image of a zebra. There are several interest points centered at each circle or pair of concentric circles. The SIFT descriptors for these interest points are quantized into three 'words', denoted as yellow, green and blue circles. For each interest point, the number of concentric circles around it denotes k, and the radius of the inner circle denotes σ_i. In other words, one concentric circle represents a distance of $1\sigma_i$ from the region, and two concentric circles represent a distance of $2\sigma_i$. The left histogram shows the un-normalized RCH, while the right one shows the final normalized and re-weighted RCH.

image segmentation to image and algorithm variations, the RCH provides robustness to possible region differences in instances of the same object class in different images or segmentations.

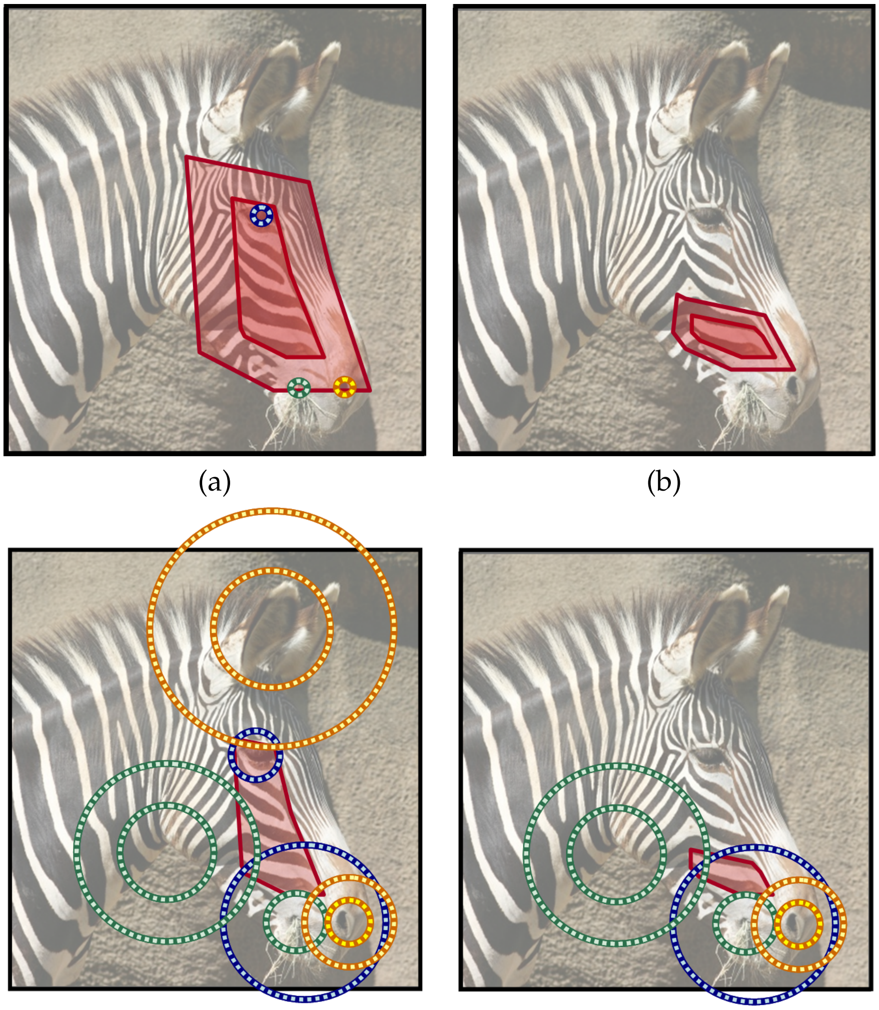
Once the RCHs have been extracted from each of the regions in the training data set, they are clustered (using K-means) into a vocabulary. The Region-based Context Features (RCFs) are the cluster centers, and regions are assigned the RCF which is the nearest-neighbor to their RCH. Figure 5.8 shows some of the regions within an RCF cluster that is discriminative for black swallowtail butterflies. The interior texture is not discriminative, however the pattern on the edges of the wings *near* these regions is discriminative. This is the power of the Region-based Context Feature.

4. Conclusions

In this section, we have presented two approaches to describing the image structure in and around regions. The texture-based representations model repetitive texture contained within a region. The novel Region-based Context Feature better models unique image structure and includes information from outside a region in a principled manner. In the next chapter we will discuss a classification scheme for discriminating between features belonging to different objects and use the scheme to evaluate the region representations for the object recognition task. Our experiments will show that the descriptors are effective for object discrimination, and are in fact complementary, improving performance when used together.

5. Contributions

- We have introduced a new feature, the Region-based Context Feature (RCF), for describing regions. The RCF is both repeatable and discriminative through its use of the SIFT descriptor, but also has a data-driven spatial support obtained from an unsupervised segmentation region.



(a) (b)

(c) (d)

FIGURE 5.7. Illustration of the possible effects of region instability on region descriptors. In image (a), the interior red region is the same as the region in Figure 5.6. The colored dots show the locations of some of the interest points in Figure 5.6 without their spatial extents. If we increase the area of the region by a fixed percentage to the exterior red region, the shown interest points would be captured. If image segmentation instead subdivided the region into smaller regions, one of which is shown in (b), increasing the area by the same percentage would not capture any interest points. Images (c) and (d) show the same region subdivision but with the RCH approach to combining regions and interest points. Since the RCH is based on the interest point scale and not the region area, many of the interest points are still captured with the smaller region, increasing robustness.

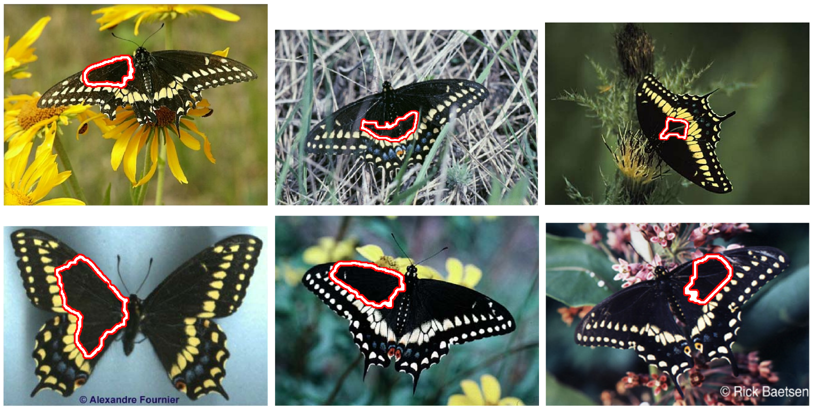


FIGURE 5.8. Examples of image regions within a discriminative RCF cluster for black swallowtail butterflies. The red and white outlines denote the regions. These regions would not be discriminative based on texture alone.

CHAPTER 6

CLASSIFYING REGIONS TO OBTAIN OBJECT MAPS

Up to this point, we have discussed the bottom-up tasks of unsupervised image segmentation and region description which have not relied on any object knowledge. We are now prepared to inject top-down object information into the process to provide a full object recognition and object segmentation system. Given a set of training regions and their descriptors, we need to learn to classify regions that are part of an object versus those which are background.

Most existing systems which attempt to precisely delineate object masks require fully supervised training data in which objects are either accurately hand-segmented or extremely obvious [15,34,51,53,57,67,68,70,80,96,99,106,122,129]. Unfortunately, generating training data with pixel-level object masks is extremely tedious and expensive, and as a consequence many fully-labeled data sets are actually quite small. There are current internet-based efforts to generate larger labeled data sets. The LabelMe initiative [95] has users voluntarily denote objects by clicking on boundary points which the program joins into a polygonal object mask. This data set has two main problems. First, the mask quality is highly variable with user accuracy and patience. Second, there is no way to determine whether every instance of an object class has been labeled in a given image. Another online attempt is being led by the Peekaboom game [124], where one player must mark object regions to show his partner so that his partner can guess the object identity in the shortest possible time. We believe that there is a significant amount of information regarding the saliency of object patches and human object recognition to be derived from this data set, however the fact that the uncovered patches are all of a fixed shape (circular), plus the fact that the whole object need not be unmasked, prevents us from using this data set.

At the other end of the spectrum, there exist methods that attempt to use completely unlabeled images as training data, such as in [16, 94]. Gathering unlabeled images is certainly cheap, however learning is extremely difficult and offers no guarantees. Given a lack of human information to help answer the question 'What is an object?', unintuitive or unimportant image structure might be learned. Why should an automatic system learn to detect human faces instead of noses, or patches of the color pink, edges in an arc, half a head plus an arm, or the floor of a room? Additional constraints can be imposed, for example that an object of interest be exactly denoted by one region [94] or that video data be available [60], but this is, respectively, incorrect and inconvenient. In [16], both supervised and unsupervised learning are performed using an LDA-like [10] framework. However, the unsupervised training set is extremely contrived, containing few object classes with little variation. Also, LDA requires that a fixed number of topics be chosen before training.

The compromise we choose is to use weakly labeled data in which images are labeled with the objects of interest that they contain, but without the object locations or masks. This type of data is far cheaper to obtain, and has been used in numerous image classification and object recognition tasks that utilize interest points (sparse or dense) or fixed-shape image patches, such as [85]. Very few systems [7, 17, 29] other than ours, however, have attempted to use weakly supervised training data and regions to learn object segmentations.

One caveat when building a weakly supervised training set is that care must be taken when choosing images for the 'object' and 'background', or 'positive' and 'negative' image sets. Without object masks, we must rely on the relative statistics of feature occurrence between the 'object' and 'background' images to learn which features indicate the object class. In order to discount non-object pixels, the 'background' images must be representative of the non-object pixels in the 'object' images. This inconvenience, however, has not prevented the creation of the multiple weakly supervised data sets that we discussed previously.

In this chapter, we will describe our method for classifying region features to create object masks. We will then apply our classifier to rigorously evaluate and compare the region features we presented in the previous chapter. We will show that both the texture and RCF region representations are in fact useful and complementary, and that each feature's effectiveness is class-dependent. Feature selection need not be done by hand, however, as our classifier can indeed accommodate multiple features of varying significance at once.

1. Classifier

Let F be a generic feature, either one of our texture-based features or the RCF. We require a method to 'score' how well F is able to discriminate between an object class and background. Since we are using a weakly-supervised framework in which we only know image labels, our score must reflect three situations:

1. Features which predominantly appear in images which contain the object are indicative of the object.
2. Features which predominantly appear in images which do not contain the object are indicative of the background.
3. Features which occur equally in object images and in background images are likely background features. Of course, there are many regions which are part of the object but which are textureless (have uniform color) or have frequently occurring image structure, such as the black portion of the butterfly wings in Figure 5.8. However, since the RCFs contain information from both inside and *around* a region, they are likely to contain information which will help discriminate these object regions from the

background. In Chapter 8, we consider how to explicitly incorporate information from neighboring regions to avoid mislabeling these kinds of regions.

The likelihood criterion presented in [28, 98] to select discriminative interest point clusters satisfies our criteria when applied to region features [86]. Let $P(F|O)$ be the conditional probability that a feature in an image containing the object is assigned to feature cluster F, and define $P(F|\bar{O})$ similarly for non-object images. Then we can define R as the log likelihood ratio of the object's presence, and $\tilde{R}$ as our posterior belief in O given F (assuming that $P(O) = P(\bar{O})$):

$$R(F) = \log \frac{P(F|O)}{P(F|\bar{O})} \in (-\infty, \infty) \tag{6.1}$$

$$\tilde{R}(F) = P(O|F) = \frac{P(F|O)}{P(F|O) + P(F|\bar{O})} \in [0, 1] \tag{6.2}$$

Note that $\tilde{R}$ preserves the ordering of R but rescales the scores to lie in $[0, 1]$. For R, a score approaching negative infinity implies that F indicates a negative image, a score approaching infinity implies that F indicates a positive image, and a score of 0 indicates that F is uninformative for either class. For $\tilde{R}$, a score of 0 implies a negative image, a score of 1 implies a positive image, and a score of 0.5 is uninformative. Laplace smoothing is performed during numerical evaluation of the probabilities for robustness. Examples of some discriminative feature clusters were given in the previous chapter in Figure 5.4 and Figure 5.8. This entire learning scheme can also be applied in a fully supervised setting by simply replacing positive and negative images with positive and negative regions.

The score $\tilde{R}$ only considers one feature, however it might be beneficial to combine region features to obtain even more powerful scores. To facilitate the combination of texture features and RCFs, one possible simplifying assumption that could be made is feature independence. Although a texture representation of a region and an RCF are clearly not truly independent, this naïve Bayes assumption allows us to estimate their joint probability from relatively little data. We define the independent score for a feature pair to be:

$$\tilde{R}\,(T, RCF) \propto \tilde{R}\,(T)\,\tilde{R}\,(RCF) \tag{6.3}$$

Where T is any one of the texture features, and RCF is a Region-based Context Feature. In this formulation, if the texture features cannot discriminate between a certain class and its background, they will all be near 0.5 and will have little to no effect on the RCFs preference for the object or background, and vice versa. If, on the other hand, both texture features and RCFs have a wide range of scores, they can reinforce each other when both agree, or cancel each other out when they disagree. Thus both feature sets can co-exist in one model.

If presented with sufficient training data, we could remove the independence assumption and model the complete joint probability of the texture features and RCFs. As in the single feature case, $\tilde{Rj}(T, RCF)$ can be modeled as:

$$Rj(T, RCF) = \log \frac{P(T, RCF|O)}{P(T, RCF|\bar{O})} \tag{6.4}$$

$$\tilde{Rj}(T, RCF) = \frac{P(T, RCF|O)}{P(T, RCF|O) + P(T, RCF|\bar{O})} \tag{6.5}$$

To summarize, the procedure for training the classifier is:

1. Generate a data set of images, with all images contain a specific object labeled as 'positive' and images not containing the object labeled as 'negative'.
2. Segment all of the training images using one of the unsupervised segmentation algorithms.
3. Extract a description of each region, using one or more of the texture descriptors TR and TM, or the RCH.
4. Cluster the region descriptions to generate the feature vocabularies of TRFs, TMFs or RCFs, respectively.
5. Assign each region to its closest cluster center in each vocabulary.
6. Compute one of the scores above for each region feature.

We are now ready to perform object recognition and object segmentation in a novel image. The procedure we follow begins much like that for training: the new image is segmented into regions and each region is described using one or more of the texture descriptors TR or TM, or the RCH. The region descriptors are each assigned to their closest center in the appropriate vocabulary, and one of the single-feature or combined features scores is computed for the region's features. Each pixel is assigned the score of its surrounding region, and the union of all pixels with scores above a given threshold forms the object mask.

The training and testing process has been formulated to use only one segmentation of each image. However, if we recall our discussion in Chapter 3, one segmentation algorithm and set of parameters rarely provides a good partition of every image in a data set. We require each image to be divided into regions which are neither too small to compute useful features, nor too large to capture all of the object boundaries. In this chapter, we take a basic approach to addressing this scale selection problem by generating three segmentations of each image for both training and testing. Specifically, we apply the mean shift segmentation algorithm with constant parameters on three different image sizes. For testing, let $s \in 1...N_s$ represent a single image segmentation, and m_s the image map of scores generated by our classifier using any one of the scores discussed above. Then the final pixel map of scores is:

$$M = \prod_{s=1}^{N_s} m_s \tag{6.6}$$

Any set of pixels which belong to the same regions in every segmentation will have the same final score. This is reasonable because pixels in the same region at all scales have a common texture and color at all scales. By assigning them the same score we achieve our original goal of classifying similar pixels in a consistent manner and avoiding pixel-level noise.

We choose to begin with the mean shift algorithm since our evaluation of segmentation algorithms in Chapter 3 showed it to be the most promising algorithm which still retains sufficient variation to generate significantly different segmentations at each scale. The image sizes are chosen such that we are confident that at least one of the segmentations is an over-segmentation (and hence contains all of the object boundaries as a subset of the region edges), and the regions from all of the training segmentations are used to generate the feature vocabularies and the feature scores. In Chapter 7, we address the problem of segmentation stability by generating even larger sets of image segmentations for each image.

We have described a method for creating one-vs-all classifiers for a data set. Some problems, however, include multiple classes, such as the Butterflies data set. We can create a multi-class classifier by running each of our one-vs-all classifiers over an image and assigning to each pixel the class corresponding to its maximum score, thresholding the responses to identify the background.

2. Region representation and classification experiments

To measure the effectiveness of our region representations, we apply them to the task of weakly-supervised learning, recognition and segmentation of the object classes described Chapter 4. Our goal here is to specify which pixels in a test image belong to an object class using single-region information alone. We begin by examining the performance of each representation individually, and then proceed to combine region representations to increase robustness. For each experiment, we provide precision and recall information about the pixel-wise classification performance, also described in Chapter 4. All classifiers were trained using the weakly supervised method introduced above.

2.1. Individual region representations

Our first set of experiments studies the performance of each representation individually using the scoring method in Equation 6.2. We experiment with model selection for region representation by choosing a representation type, TM, TR or RCF, a texton or SIFT dictionary size K_{TEX}, and a size for the feature, or histogram, dictionary K_{TM}, K_{TR} or K_{RCF}.

The first comparison we present is on the Spotted Cats data set, with results shown in Figure 6.1. Recall that the task is to differentiate between the cats and the background. The top table presents the average precision of pixel labeling over all recall values for each model choice. The top row gives the feature type, the second row gives the K_{TEX} used, and the left column gives the appropriate K_{TM}, K_{TR} or K_{RCF}. Full precision-recall curves are given for a selection of the model choices in the middle plot. Curves corresponding to the other model choices are omitted for clarity. While larger dictionaries seem intuitively more desirable, the results in Figure 6.1 disagree. In the table, we can see that the average precision does not increase monotonically with dictionary size. In the curves, we can see that for the TM representation on the Spotted Cats dataset, a dictionary size of 300 performs better for low recall values, a dictionary size of 50 performs better for medium recall values, and they perform similarly for high recall values. The same trend can also be seen for the TR curves. We must conclude that it is in fact possible to over-fit the model, and that model selection is important.

Figure 6.2 gives example images and results for the Spotted Cats. The first column contains the original images with hand-drawn ground truth for reference (although it was not used for either training or testing). The second column contains results obtained using the texture-based TM region features with $K_{TM} = 50$ alone. The third column contains results obtained using RCFs with $K_{RCF} = 50$ alone. Pixels which show image information were classified as being part of a cat, while red pixels were classified as background. For display purposes, detection thresholds were chosen to give equal error rates on the images. Notice that in the first and third rows, the texture-based classification misses parts of the cats' heads, while in the second example it misses the shadowed section under the chin. These are all low-texture regions. The RCF-based classification is able to properly label these regions, but has trouble with the tails which are mainly surrounded by background. The fourth row shows an all-around poor result where background texture is too dominant and a large part of the object is too shadowed for proper classification. These results support our hypothesis that texture features and RCFs are complementary.

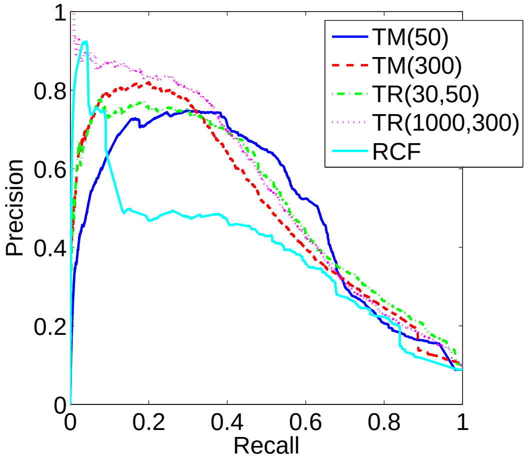
We perform the same comparison on the PASCAL Cars data set, with results shown in Figure 6.3. The task here is to differentiate between cars and the background. The table gives the average pixel-wise precision for each model choice, and the middle plot shows full precision-recall curves for a subset of the model choices, as in the Spotted Cats experiments. As for the Spotted Cats, we can see that average precision does not increase monotonically with dictionary size, so model choice is important. Also as for the Spotted Cats, the choice between the TM or the

Results for the Spotted Cats Data Set

Average Precision for All Individual Region Representations

	TM **30**	**TR** **30**	**TR** **200**	**TR** **1000**	**RCF** **50**
50	0.493	0.513	0.504	0.542	0.392
300	0.498	0.493	0.533	0.545	-

Precision-Recall Curves for Selected Individual Region Representations



Precision-Recall Curves for a Combination of Region Representations

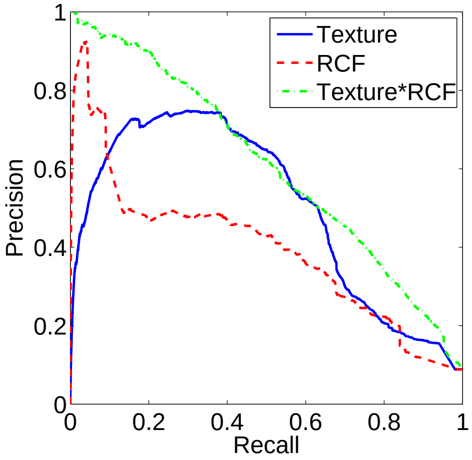


FIGURE 6.1. Comparison of region representations on the Spotted Cats data set. The notation follows that in Chapter 5. In the table, the first row indicates the region representation used, the second row indicates the size of the texton (or SIFT) dictionary, and the left column indicates the size of the feature dictionary. The table entries give the average precision for each feature variation. Note that the average precisions do not increase monotonically with dictionary size. The first plot shows the full precision-recall curves for a selection of the individual feature variations. The second plot shows the full precision-recall curve for combining the features assuming independence as in Equation 6.3. Two of the individual representations are also copied from the first plot as a comparison.

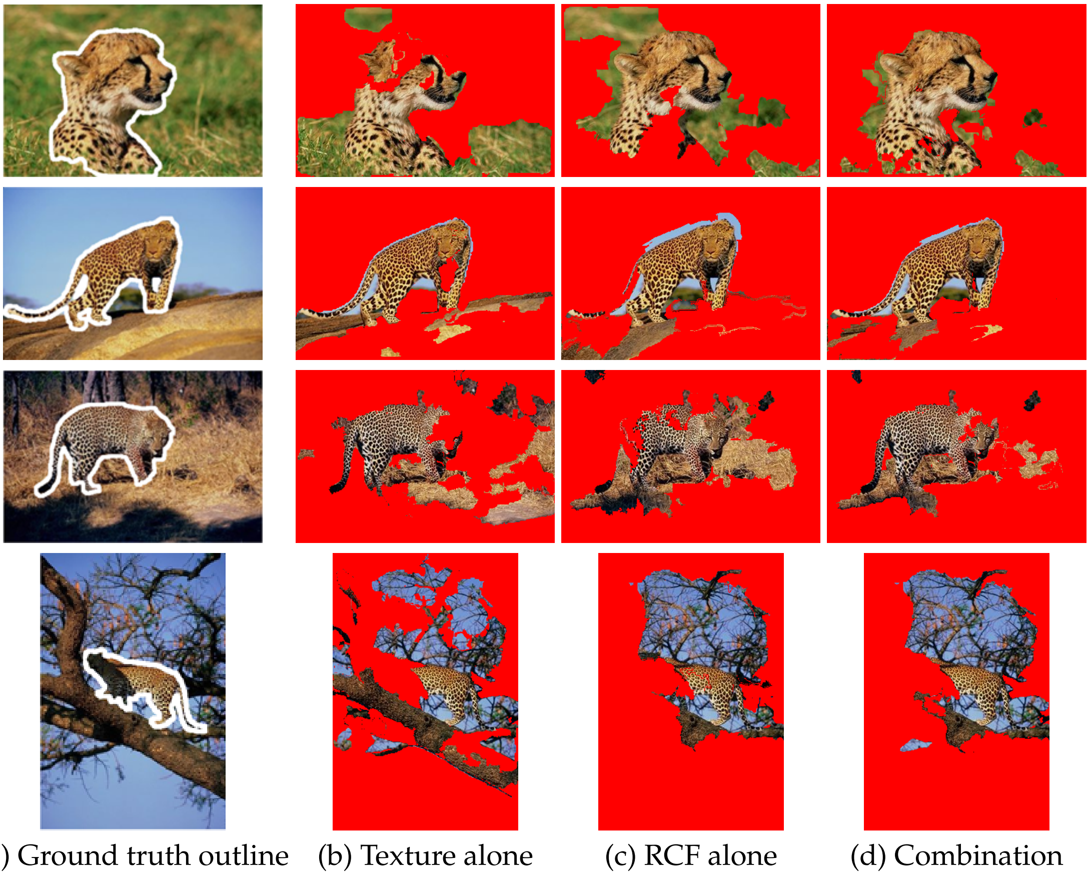


FIGURE 6.2. Object maps for the Spotted Cats dataset generated using various features. Image pixels were classified as being part of a cat, while red pixels were classified as background. All of the white outlines represent ground truth, not results. Column (a) contains the original images, column (b) contains results from texture-only classification using TM features with a 50-word dictionary, column (c) contains results from RCF-only classification, and column (d) contains results from texture and RCF classification combined using the independence assumption in Equation 6.3. The first three rows show good results, while the fourth result is poor due to a highly textured background, small object scale, and strong shading. Combining features increases labeling accuracy.

TR representation is dependent on the desired precision-recall trade-off. However, the relative performance of the texture representations to the RCFs on the Cars data set is in stark contrast to the relative performance on the Spotted Cats data set. The regular texture on the bodies of the Spotted Cats led to the texture descriptors generally outperforming the RCFs. For the PASCAL Cars, the opposite is true. Since much of the surface of a car is textureless, the texture descriptors all do very poorly. Cars do, however, have discriminative unique image structure, such as the wheels or the corner of the windshield. The RCFs capture this structure and perform quite well. If we are to choose only one region representation for our recognition task, it seems that the choice must be object dependent.

The images in the Graz02 Bikes data set provide yet another scenario. Consider the results shown in Figure 6.4, specifically the results of using the texture TM representation with $K_{TM} = 50$, and those of using the RCFs. Given that bikes do not have any obvious repetitive texture, but do have interesting structure such as the wheels and frame angles, we would expect the RCFs to perform much better than the TMFs. Although the RCFs do in fact outperform the TMFs, the difference is not as large as predicted. We hypothesize that the texture descriptors are capturing shape information. Since the regions are long and thin, the pixels on the boundary of a region are a significant fraction of the pixels in the entire region. This implies that the 'texture', in this case straight edges, along the boundaries are not swamped by the texture (or lack thereof) in the region interior. Thus, the texture descriptors can in fact model the straight lines along the region boundaries.

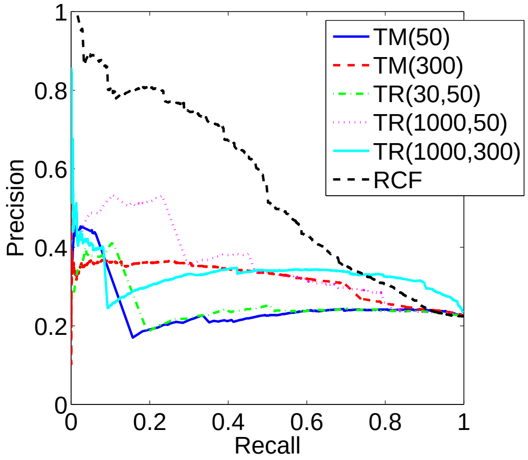
The final data set we use for our representation experiments is the Butterflies data set. This data set contains seven butterfly species, so there are in fact seven one-vs-all recognition problems to be explored. We consider seven separate tasks, each one consisting of differentiating between a specific butterfly species and a background class which includes non-butterfly pixels as well as all the other butterfly species. Figure 6.5 gives the pixel-level precision-recall curves for each of the one-vs-all classifiers. We focus on the results of using the texture representation TM with $K_{TM} = 50$ alone, and using the RCFs alone. The full range of performance can be seen in these experiments. The zebra and monarch (closed position) butterfly classifiers produce excellent results using either texture or RCFs due to the regular and distinctive texture on those species' wings. The machaon and monarch (open position) butterfly classifiers produce promising results using the RCFs, but less so using the texture representation. It is unclear exactly why the RCFs should outperform texture so much on these classes, although perhaps the variation in the data set and the backgrounds could be an explanation. The peacock, black swallowtail and admiral species are more challenging and strain the system. They all

Results for the PASCAL Cars Data Set

Average Precision for All Individual Region Representations

	TM	**TR**	**TR**	**TR**	**RCF**
	30	**30**	**200**	**1000**	**50**
50	0.246	0.253	0.302	0.353	-
300	0.316	0.261	0.346	0.329	0.548

Precision-Recall Curves for Selected Individual Region Representations



Precision-Recall Curves for Selected Combinations of Region Representations

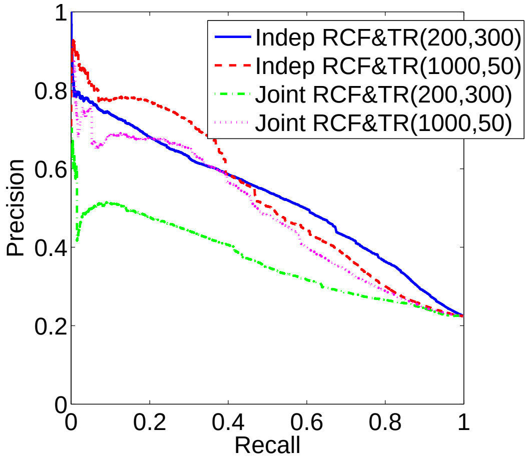


FIGURE 6.3. Comparison of region representations on the PASCAL Cars data set. The notation follows that in Chapter 5. In the table, the first row indicates the region representation used, the second row indicates the size of the texton (or SIFT) dictionary, and the left column indicates the size of the feature dictionary. The table entries give the average precision for each feature variation. Note that the average precisions do not increase monotonically with dictionary size. The first plot shows the full precision-recall curves for a selection of the individual feature variations. The second plot shows the full precision-recall curves for a selection of the feature combination methods. 'Indep' implies feature independence as in Equation 6.3; 'Joint' implies using the joint distribution as in Equation 6.5.

Results for the Graz02 Bikes Data Set
Precision-Recall Curves for Two Individual
and a Combination of Region Representations

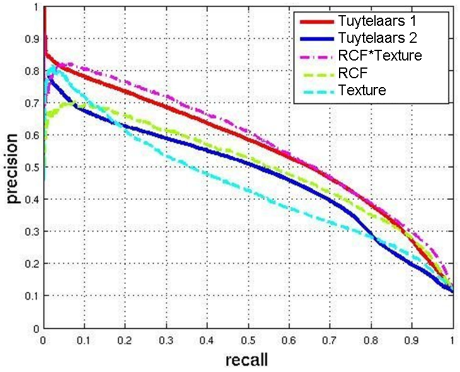


FIGURE 6.4. Precision-recall curves of results on the Graz02 data set. The turquoise dashed line represents classification using region texture TM(50) alone, the green dashed line using RCFs alone, and the purple dash-dotted line represents using the independent feature-combination method in Equation 6.3. The red and blue solid lines show results from [113].

lack discriminative repetitive texture, even the texture on the edges of the wings of the swallowtail can also be found on the wings of the monarch classes. In addition, they all have large uniform regions. With these hurdles, the texture-based classifier is essentially useless, as we saw for the PASCAL Cars data set. In these cases the texture-based classifier is essentially useless, severely handicapping the system. The RCFs are able to classify a portion of the pixels on these species correctly.

Figure 6.6 shows example images and results for each of the butterfly species, again classified with one-vs-all classifiers. Notice that the texture features TM(50), with results shown in column (b), are able to extract most of the regular textures on the butterflies, while the RCF results in column (c) capture some of the nearby texture-less regions. It once again appears that the two feature types are complementary.

From our quantitative experiments using individual features, we can conclude that on many data sets we can achieve promising results with our features. The issue, however, is that different data sets seem to require different representations.

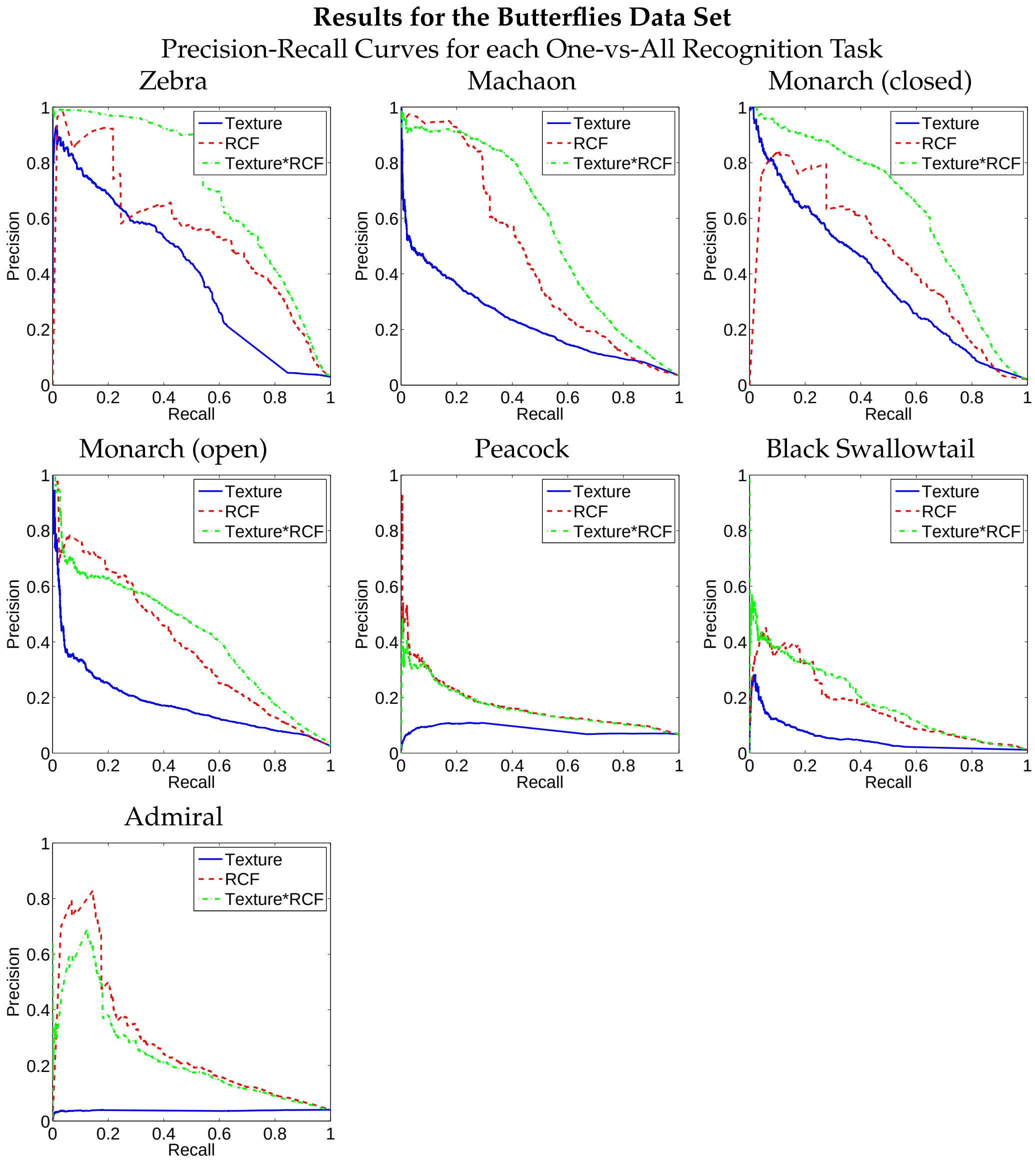
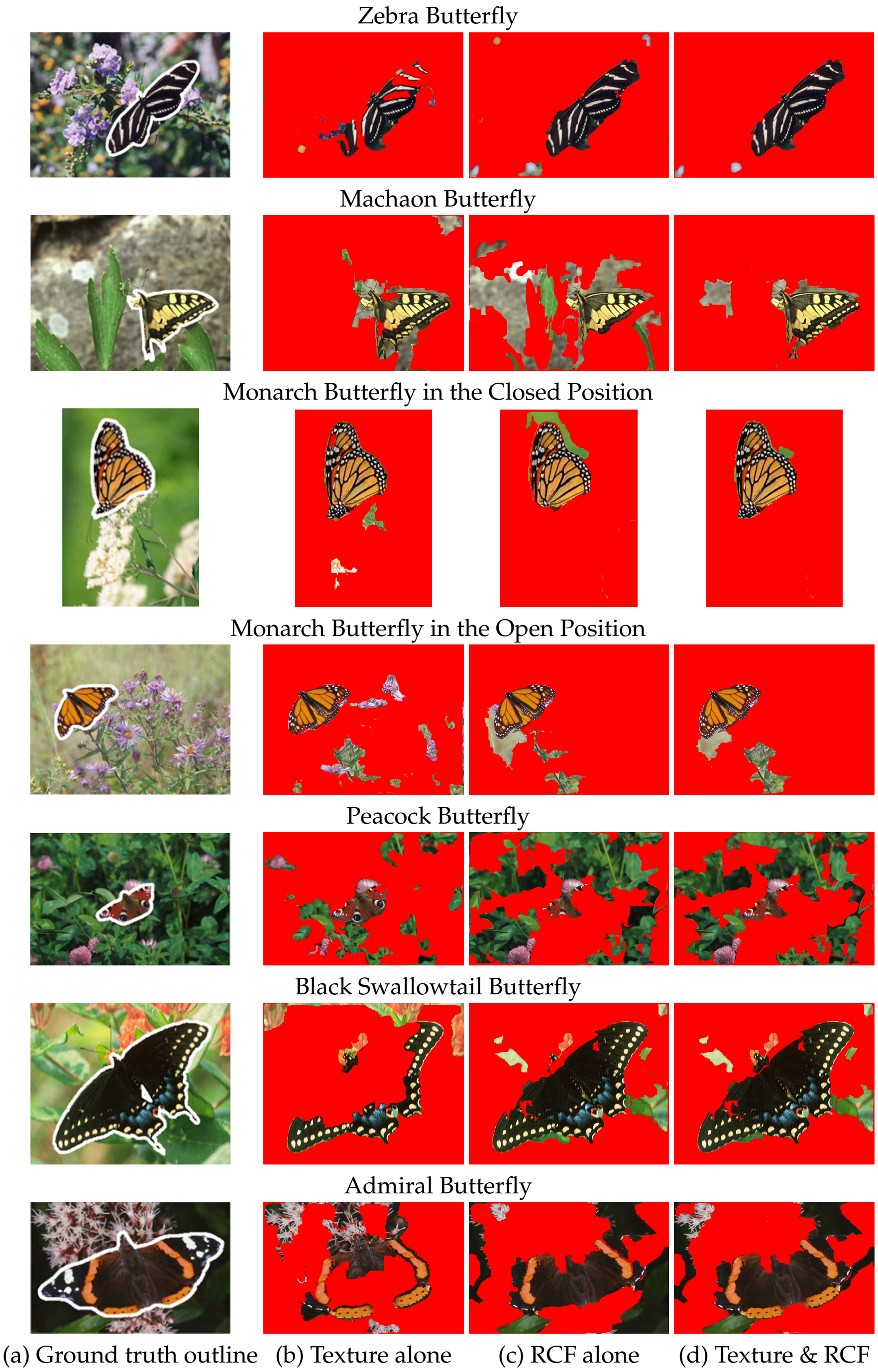


FIGURE 6.5. Precision-recall curves of results on the Butterflies data set for each one-vs-all recognition task where a specific butterfly species must be recognized, and the 'background' class contains not only non-butterfly pixels, but also all of the other butterfly classes. The blue solid line represents classifying using region texture TM(50) alone, the red dashed line using RCFs alone, and the green dash-dotted line represents using the independent feature-combination method in Equation 6.3.

For example, the Spotted Cats were best-recognized using texture, while the PASCAL Cars responded best to the RCFs. If we do indeed need to perform feature selection by hand for each data set, then this framework is impractical.



Zebra Butterfly

Machaon Butterfly

Monarch Butterfly in the Closed Position

Monarch Butterfly in the Open Position

Peacock Butterfly

Black Swallowtail Butterfly

Admiral Butterfly

(a) Ground truth outline (b) Texture alone (c) RCF alone (d) Texture & RCF

FIGURE 6.6. Object maps for the Butterflies data set generated using various features. For each row, image pixels were classified as being part of a specific butterfly species, while red pixels were classified as background or a different butterfly species. Column (a) contains the images, column (b) the results of using texture TM features alone with a 50-word dictionary, column (c) the results of using RCFs alone, and column (d) the results of combining TMFs and RCFs as in Equation 6.3. The texture and RCF representations are often complementary; the TMFs find regular textures while the RCFs also find textureless regions in close proximity to structure. Recognition can fail such as in row 5, due to very small scale, low object texture, and high intraclass variability.

Our goal now must be to eliminate the need to choose between representations. In the next set of experiments, we look at combining the region representations using the two methods in Equations 6.3 and 6.5. If either of those two combination methods consistently gives results at least as good as those of any one feature alone, we can give all of the features for each region to our classifier and avoid hand-picking a representation.

2.2. Combined region representations

We once again look to the results on the Spotted Cats data set in Figure 6.1 for our first foray into combining representations. The bottom plot shows two of the curves from the individual representations, those of using texture TM(50) and RCFs. This plot also shows the effects of combining the two individual representations using Equation 6.3 which assumes feature independence. The combined representation does indeed show higher precision for most recall values than either individual representation. The qualitative results shown in the fourth column of Figure 6.2 reinforce these results. The combined features are able to capture more of the pixels on the cats, while eliminating background noise.

The results on the Graz02 Bikes in Figure 6.4 lead to the same conclusion. By combining the two representations together, we obtain higher performance than either representation alone. The plot contains two additional curves showing the results obtained by Tuytelaars and Schmid [113]. The curves our algorithm produce are higher despite the fact that our algorithm is older. To compare our performance with the original results on this data set by Opelt et al. [85], we adjusted our algorithm to output object centers. For the positive (bike) images, we used as our localization the center of the largest connected component of values greater than 0.6 in the likelihood map M. Setting this threshold is equivalent to setting the number of clusters k in [85], which is also set by hand. Since this form of localization is not the main focus of our work, we did not perform a thorough search for the optimal threshold. Using the comparison method from [2] used by Opelt et al., a localization is considered correct if it falls within the ellipse inscribed in the bounding box surrounding the ground truth. In the case of multiple ground truth objects, finding any one is considered correct. Within this framework, we are able to localize 131/150 of the bikes, compared to 115/150 for Opelt et al.

The effects of combining features are most evident on the Butterflies data set in Figure 6.5. In the top three plots, we can see that combining representations has led to dramatic improvements over either representation alone. In the middle three plots, the combination of features performs as well as the top individual feature, which is sufficient to attain our goal of not choosing between feature representations. Only the Admiral species in the last plot suffers slightly from the combination, which we believe is due to the unfortunately poor performance of the texture descriptor. The qualitative evaluation on the Butterflies data set in Figure 6.6 once again confirms our results. Notice that in all examples but the Peacock butterfly, the results of combining representations in the fourth column show more of the butterfly pixels correctly classified, as well as less background noise.

The Butterflies data set allows us to attempt the more complex task of multi-class recognition. Instead of simply asking whether each butterfly species is more likely to be present than the background, we can instead ask for the overall most likely species at each pixel. We convert our method into a multi-class classifier in the following manner. Let the maximum one-vs-all classifier value at pixel p_i be $c^* = \text{argmax}_{c=1}^{C} M_c(p_i)$, where M_c is the map for butterfly species class c, computed as in Equation 6.6. Then the overall class at pixel p_i is:

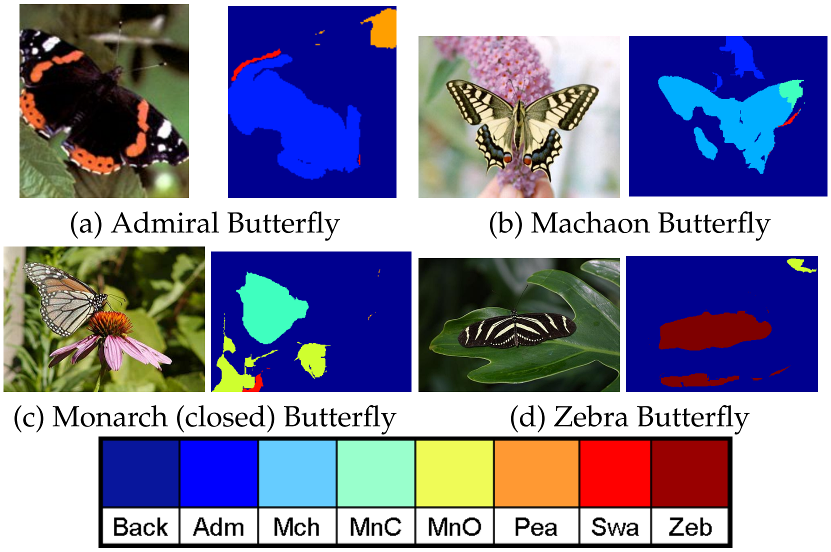
$$C(p_i) = \begin{cases} c^* & M_{c^*}(p_i) > 1.5(1 - M_{c^*}(p_i)) \\ background & otherwise \end{cases}$$

In Table 6.7 we can see the pixel-wise classification rates for the butterfly classes. The only two low scores came from the Swallowtail and Peacock classes which performed poorly in the one-vs-all tasks as well. Figure 6.8 shows some examples of classifications. The multi-class framework seems to have improved classification. One possible explanation for this is that a consensus between 7 classifiers is required to label a region as background, increasing the precision.

Thus far we have combined our feature representations as in Equation 6.3, using the assumption of feature independence. Can we improve our results by instead modeling the full joint distribution of the features as in Equation 6.5? We

Butterfly	Classif.	Butterfly	Classif.
Admiral	76.09%	Swallowtail	13.57%
Machaon	53.58%	Zebra	74.94%
Monarch-closed	92.67%	Pixel avg	53.84%
Monarch-open	66.85%	Class avg	57.59%
Peacock	25.63%		

FIGURE 6.7. Classification rates for the multi-class problem on the Butterflies dataset.



(a) Admiral Butterfly

(b) Machaon Butterfly

(c) Monarch (closed) Butterfly

(d) Zebra Butterfly

FIGURE 6.8. Examples of multi-class classification on the Butterflies data set. Each pair of images shows the original image input and the classified output. Each color in the output represents a butterfly class, with dark blue the background, as given in the color chart.

performed experiments on the PASCAL Cars to test this hypothesis. This data set was chosen since it is the largest used in these experiments, with 1062 training images, 553 of which are cars, and so has the best hope of modeling the complexity of the joint distribution. In Figure 6.3 we compare the effects of using the two feature combination models. For clarity, only two representative curves per model are plotted. The combination of RCFs with the TR texture representation with $(K_{TEX}, K_{TR}) = (1000, 50)$ provided the best results for both combination methods, surpassing even $(K_{TEX}, K_{TR}) = (1000, 300)$. Despite a large amount of training data, the best results are not always obtained with the largest dictionaries. Another counter-intuitive result is that combining the features with the independence assumption performs better than modeling the joint distribution despite the large data set. Finally, although the combination $(K_{TEX}, K_{TR}) = (200, 300)$ performs very poorly in the joint distribution representation, it outperforms $(K_{TEX}, K_{TR}) = (1000, 50)$ for high recall values in the independent distribution representation. This indicates that performance under one type of distribution does not mirror performance under the other. This result can be applied to other object recognition systems, suggesting that many common assumptions in system implementations should in fact be verified.

To conclude, we have shown that by combining multiple features in a straight-forward manner which assumes independence our system performs as well as, or better than, using individual features alone. This allows us to execute our system without a user in the loop to choose a region representation.

3. Conclusions

In this chapter, we have presented a region classifier which requires only weakly-supervised training. Through experiments on a number of data sets, we have shown that this classifier can obtain promising performance. Although using individual features is sufficient in many cases, by combining features we can improve performance and eliminate the need for class-dependent feature engineering.

4. Contributions

- Extension of a weakly-supervised classification scheme to our region-based framework, which results in object recognition and segmentation.
- Extensive testing of the region features introduced in Chapter 5.
- Demonstration that the features can be combined in a straight-forward manner to improve performance and eliminate the need for class-dependent selection.

CHAPTER 7

INTEGRATING INFORMATION FROM MULTIPLE IMAGE SEGMENTATIONS

So far, we have demonstrated that by segmenting an image, carefully describing the image information associated with each segmentation region, and using a weakly-trained classifier, we can obtain promising results for pixel-level object recognition and object segmentation. In Chapters 2 and 3, we discussed multiple segmentation algorithms and their performance. Our experiments led us to believe that among the algorithms we tested, mean shift-based segmentation produced the most promising segmentations. However, none of the algorithms, including mean shift, was perfect. In fact, the performance of each algorithm was variable with respect to the parameters used and the images considered. These conclusions resulted in our use of mean shift segmentation for three different scales of each image, giving the segmentation algorithm increased opportunity to capture useful regions which separated objects from background and were still large enough to compute meaningful statistics. In addition, this range of segmentation scales allowed us to capture local features in the smaller regions, and contextual features in the larger regions.

The obvious question that arises is: can we do even better? Looking back at the results in Chapters 2 and 3, it seems that each segmentation algorithm has its own strengths and weaknesses. The mean shift-based algorithm captures object boundaries well but is susceptible to over-segmenting the image. The graph-based segmentation algorithm is well-suited to capturing long and thin, or 'wiry', object parts, but also hallucinates thin regions where they do not exist. The normalized cuts algorithm based on boundary information (Pb) can capture larger regions, but is biased in favor of creating regions which have equal sizes, often leading to unnecessarily dividing or joining regions. We noticed that, in fact, even humans could not agree on a 'correct' segmentation for an image without prior object information. All of this variation and instability has led to advocacy in the computer vision community for using multiple segmentations of each image [6, 11, 52, 86, 94, 111].

In this chapter we examine the issue of using multiple segmentations per image, such as those in Figure 7.2. Specifically, we explore the question of how to best leverage the information from a number of bottom-up segmentations created using multiple algorithms, parameters, image scales and features. By integrating all of the image partitioning hypotheses in an intuitive combined top-down and bottom-up approach, we provide our recognition algorithm with multiple choices for object and feature support. We will show how the use of all the bottom-up segmentations in concert, along with established and straight-forward learning algorithms, leads to improved object segmentation performance and increases robustness to incorrect image segmentations. We will begin with a supervised approach, but we will also explore extensions toward using weakly supervised training data for efficiently increasing the training set size.

Throughout this chapter, we will present results on the MSRC 21-class data set, as well as the challenging data set used in the PASCAL VOC2007 segmentation competition, both of which were described in Chapter 4. For the MSRC 21-class data set, we compare our results to the Textonboost [106] approach, which was the original source of the data set and used exactly the same split between training and test sets as we do. We will also compare to Verbeek and Triggs [119], although they use a different split of the data. For the PASCAL VOC2007 segmentation challenge data set, we compare against the Oxford Brookes entry into the competition [64]. This was the only entry into the segmentation challenge itself and hence the only entry to use the small subset of the training data with segmentation masks available. We also show results for the TKK [120] entry which was the won the segmentation challenge. However, the TKK entry was actually entered into the detection

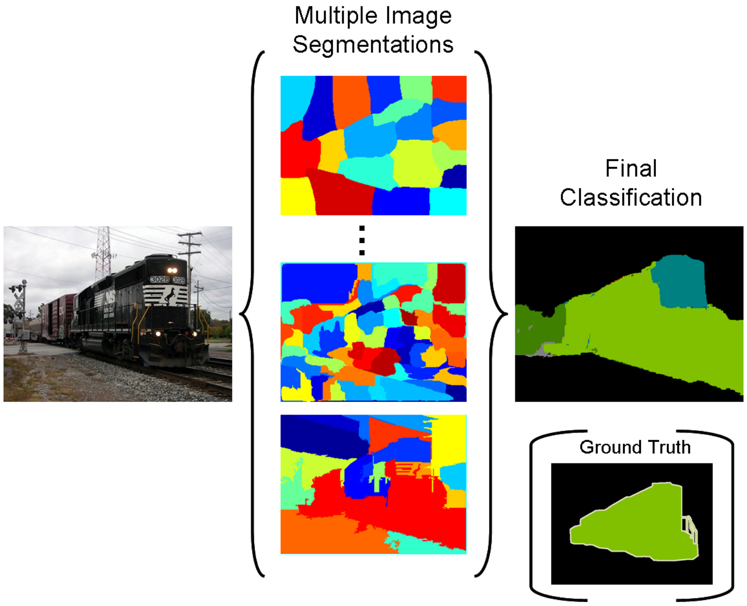


FIGURE 7.1. Our approach combines multiple bottom-up image segmentations in a principled and intuitive manner to produce robust object recognition and object segmentation results.

challenge (the object segmentation results were created by treating the bounding boxes as object masks), and used a much larger training data set consisting of thousands of images, thus it is not directly comparable to our own results.

1. Related work

Multiple image segmentations were used by Russell et al. [94] to create a pool of regions, each one of which was assigned a probability of belonging to an object class. The region with the highest probability in each image was identified as containing the object. This approach recognizes the need for multiple segmentations to obtain good object support, however it makes the often incorrect assumption that one region will perfectly define the object.

The methods introduced by Tu et al. [111,112] for incorporating multiple object partition hypotheses into one framework involve both object models and generic models of image regions. Their goal is not to recognize objects (except faces and text), but to partition the image in a logical manner. While their results are promising, their technique is quite complex and unintuitive. Sharon et al. [102, 103], and

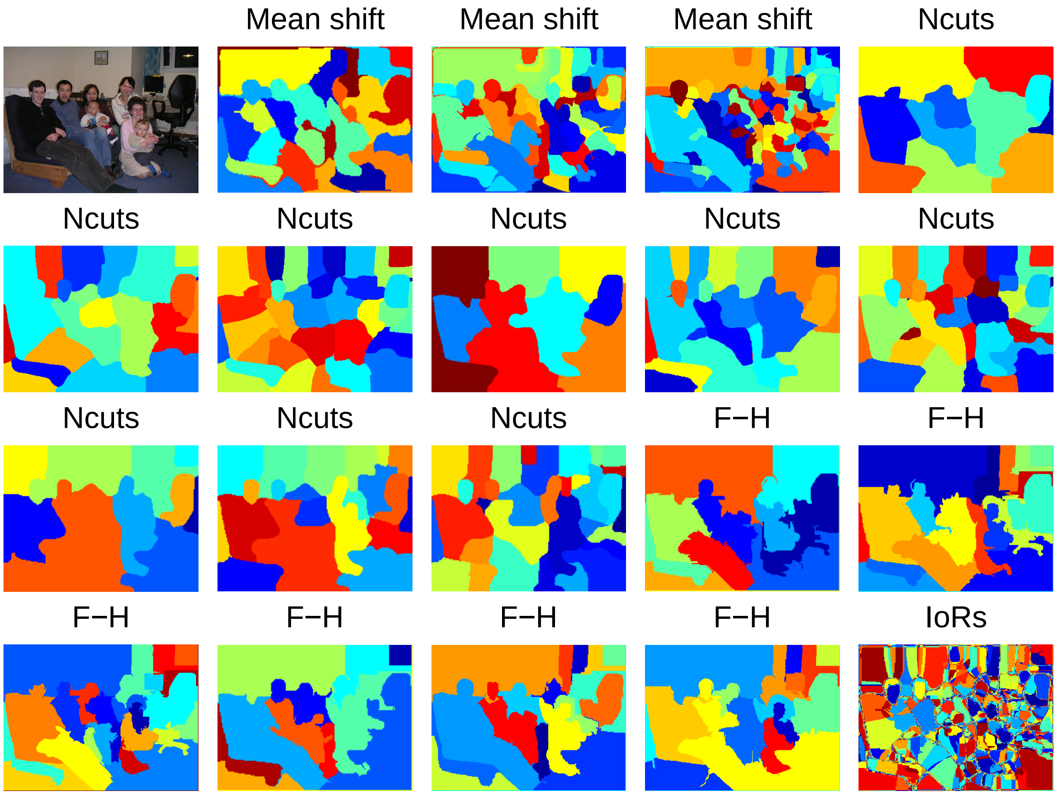


FIGURE 7.2. An example of the 18 segmentations we use, 3 from mean shift, 9 from Ncuts, and 6 from the F-H method. The variation in segmentations generated with different parameters and algorithms is evident. The final image shows the image partitioned into intersections of regions (IofRs), sets of pixels which belong to the same region in every segmentation.

Ahuja [4], also present methods for multi-scale segmentation and use boundary information to partition the image, but they do not use any object information. Rabinovitch et al. [90] also create multiple segmentations and perform cue combination, but their approach likewise does not use object information.

Borenstein et al. [11] use a hierarchy of segmentations to aid in recognition. However, their object recognition and segmentation is driven by choosing regions which fit a shape model. This assumes a strong prior on semi-rigid shapes, in their case side views of running horses and running people.

Hoiem et al. [52] and Saxena et al. [97] utilize multiple image segmentations to provide different feature supports in systems which categorize regions into 3-D geometric classes, such as vertical or ground. Aspects of Hoiem's work influence our own, and we will discuss it further below.

2. Generating multiple segmentations

Our goal in generating multiple segmentations of each image is to capture as much of the variation in color, edge contrast, texture, scale, noise, as well as artifacts inherent in every image segmentation algorithm's output. We need to ensure that every object boundary is captured in at least one of the segmentations, and that every pixel belongs to at least one region which is large enough for meaningful feature computation, but does not cross object boundaries. To give us the best possible chance of generating a set of segmentations that satisfy these criteria, we generate our segmentations using a number of the segmentation algorithms discussed in Chapter 2 with varying parameters and features. Any other method for generating multiple segmentations which meets our criteria could also be used.

For our implementation, we use 18 segmentations. The first three segmentations are generated as in Chapter 6 by using the mean shift-based segmentation algorithm [25] with pixel position, color (in the L*u*v color space), and a histogram of texton words as features [86]. We perform the segmentation over versions of each image whose dimensions have been scaled to 0.5, 0.75 and 1 times their original length.

The second set consists of nine segmentations generated using the normalized cuts algorithm with the 'probability of boundary' features as described in [44, 76]. We vary the image size as for the mean shift segmentations, as well as generating segmentations with 9, 21 and 33 regions (as suggested in [72]) for each image size.

The final set of six segmentations is generated using the graph-based segmentation method by Felzenszwalb and Huttenlocher (FH) [36]. The same three image sizes are used, and for each image size we use two different values for the parameter $k = \{200, 500\}$ which affects the scale of the final regions by controlling the propensity for regions to merge.

The EM segmentation algorithm is not used because it produces unsatisfactory results. The hybrid segmentation algorithm is also not used, but for different reasons. The accuracy of the hybrid algorithm is only minimally better than that of the mean shift algorithm, and still far from perfect. However, the hybrid algorithm is more stable. This is a desirable trait when trying to generate a single accurate segmentation, but counterproductive when trying to generate multiple different segmentations. Finally, the shapes of the hybrid segmentation regions are similar to the mean shift regions, so they would be redundant.

Examples of the segmentations we generate can be seen in Figure 7.2. This figure highlights the possible variation in segmentation granularity. Also, notice the different characteristics of the segmentation algorithms: the mean shift segmentation regions are slightly rounded (due to the texture features) and smaller, the normalized cuts regions are also rounded and tend to be of similar size even if homogeneous regions need to be broken, and the FH method more easily captures corners and long and thin object parts.

3. Describing and classifying regions from a single segmentation

Our approach to using multiple segmentations begins by classifying the regions in each segmentation independently. Given that we will perform our initial evaluation in a fully supervised framework, we will alter our previous method for classifying single regions somewhat.

Our region descriptors will consist of three feature types. The first two dimensions are the coordinates of the centroid of the region normalized by the image dimensions. The next 100 dimensions are composed of a color histogram of the quantized hue features by van de Weijer et al. [118], accumulated over the region. Image structure within and near a region is represented by a 300-dimensional version of our Region-based Context Histogram, the RCH. This gives a 402-dimensional region-specific feature. Given the difficulty of the data sets used for evaluation in this chapter, we also include overall image context to provide a prior on the image

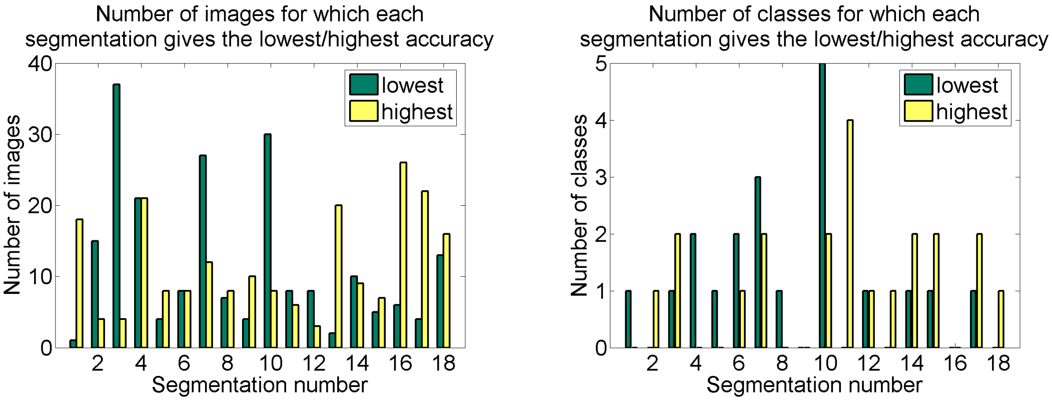


FIGURE 7.3. Histograms of the number of PASCAL 2007 images (left) or object classes (right) for which each single segmentation provides the best or worst pixel accuracy. Each segmentation is the best or worst on at least one image, and most are the best or worst on at least one object class. This suggests that all of the segmentations are useful, and none should be discarded nor used exclusively.

classification. Image context is provided by accumulating the color and RCHs over the entire image, to produce a final feature vector of 802 dimensions.

There are numerous classification algorithms available for use in a fully supervised framework. For our experiments, the support vector machine implementation provided in LIBSVM [20] has worked well. Using the method in [130], LIBSVM provides the probability that the label of a region c_r is k, $P(c_r = k|r)$, and our classification of the region is then $\text{argmax}_k P(c_r = k|r)$. As future work, it would be interesting to compare multiple classification algorithms, but a preliminary comparison with boosted decision trees has shown little difference in performance.

As a baseline, we experiment with this single-segmentation framework on the MSRC 21-class data set and the PASCAL VOC2007 segmentation challenge data set. Our hypothesis is that the object recognition and object segmentation results will vary for different segmentations of the same image. This hypothesis is confirmed by the results in Figures 7.5, 7.6, 7.7 and 7.8. The accuracy of the object masks seems to vary not only with the difficulty of the object, but also with the quality of the bottom-up segmentation.

Tables 7.1 and 7.2 show that the best results achieved by a single segmentation are competitive with the state-of-the-art. However, the variability of the results is problematic. In Table 7.2, we can see that on the PASCAL VOC2007 data set the performance of the best and worst single segmentations, averaged over all classes, differs by 5.5%. The situation is even worse for the MSRC 21-class data set shown in Table 7.1, where the top-left column shows the gap to be 10.2%.

How can we choose which segmentation algorithm to use? The obvious answer is to hold out a validation set and choose the segmentation algorithm and parameters which perform best on that set. However, this not an ideal solution. In Chapter 3, we asked the question: Can one segmentation algorithm and parameter choice provide the 'best' segmentations for all of the images in a data set? Unfortunately, the answer was 'No.' The choice of segmentation algorithm and parameters changed for every image. Even more problematic was that the same segmentation could produce mixed results within the *same* image. These results are reflected in our experiments on the Pascal VOC2007 data set as well. Figure 7.3 shows the performance resulting from using each segmentation in two ways. The left subfigure shows the number of images for which each segmentation leads to the best or worst results. We can see that every segmentation is the best on at least one image, and every segmentation is the worst on at least one image. The right subfigure shows the number of classes for which each segmentation leads to the best or worst results. Most of the segmentations are the best or worst on at least one class. In addition, image performance and class performance are not always correlated. For example, segmentation 11 is gives the highest performance on more classes than any other algorithm, but its image-based performance is unremarkable. These results suggest that none of the segmentations should be discarded nor used exclusively, so we propose to integrate the information from all of the segmentations.

4. Integrating multiple segmentations

In Chapter 6, we discussed the fact that pixels which belong to the same region in every segmentation of an image share a significant number of features, and so should be assigned to the same object label. We make that concept concrete here by defining the term 'Intersection of Regions' (IofR) to refer to the set of pixels which belong to the same region in every segmentation, as in the last image in Figure 7.2

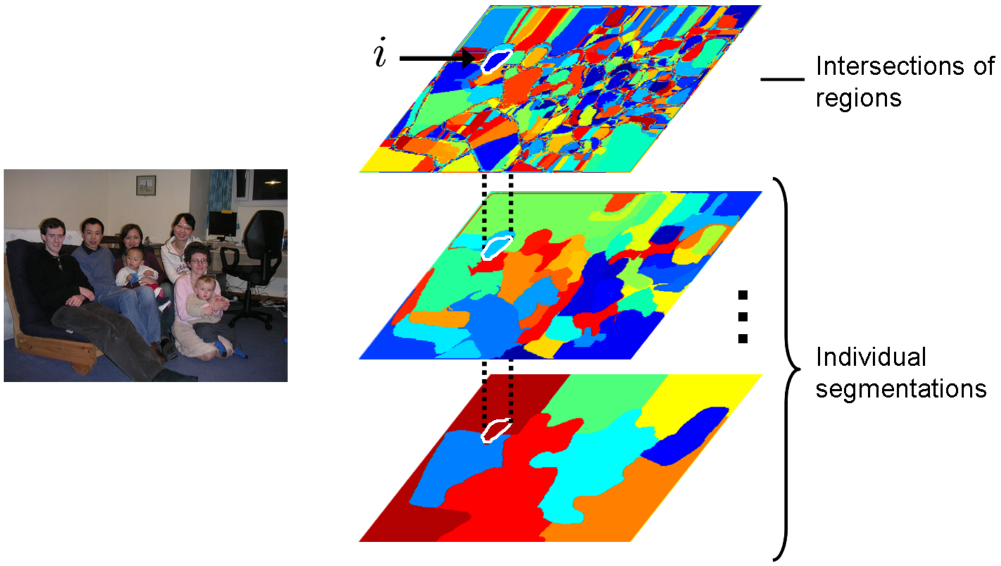


FIGURE 7.4. Example of an intersection of regions i and its parent regions in two of the eighteen segmentations. In the top individual segmentation, i's parent region (in blue) contains the man's head, while in the bottom individual segmentation i's parent region (in dark red) contains both the head and parts of the wall and floor. The quality of the individual segmentation regions varies, but the intersections of regions are almost always contained in one object.

and Figure 7.4. These are the 'basic units' of our approach, and every result we present will show the same class label for all the pixels in each IofR. IofRs could be likened to superpixels [93], however they are constructed by intersecting larger regions and not by performing image segmentation with small kernel bandwidths or many regions. Thus intersections of regions may in fact be quite large in sections of the image that are homogeneous, such as the wall in the Figure 7.2, or quite small in sections with large variation, such as the people. The key to our approach is that features and classification probabilities are never computed on the IofRs. IofRs often have very small spatial support, so feature computation would be unstable. Instead, we compute features on the original segmentation regions, which have sufficient spatial support. By using the region features, we can provide more reliable region classification than would be possible on the IofRs. In this section, we concentrate on how to combine the classification information from each of the segmentations into predictions for the IofRs.

Our first method for integrating multiple segmentations will consist of marginalizing over each of the individual segmentations. Specifically, to classify a given IofR i, the results from the individual segmentations can be combined by marginalizing over the regions r_i^s that contain i in each segmentation s. Let c_i be the class label of IofR i and k a specific class label. We use I to represent general image data instead of our previous use of F to represent features to simplify future notation. We define our first method for integrating information from multiple segmentations to be:

$$P(c_i = k|I) \propto \sum_s P(c_i^s = k|r_i^s, I) \tag{7.1}$$

The above formulation assumes that all of the segmentations should have an equal vote in the final classification. This seems reasonable if the object labeling created by every segmentation is equally good. It can also work if the cumulative accuracy of a subset of the segmentations outweighs the cumulative inaccuracy in the remainder, as we saw while using only three segmentations in Chapter 6. However, as we have discussed, segmentations can vary widely in quality.

Instead of considering all regions equal, another approach is to evaluate the 'goodness' of a segmentation region. In Chapter 3, we evaluated segmentations in comparison to ground truth, however we do not have ground truth on novel images. Instead, we must define a concept of region correctness for which a classifier can be trained. We choose to use the definition of region goodness and method of classification proposed by Hoiem et al. [52]. This approach assumes that if most of the pixels in a specific region are known to belong to one object class, the region's classification score should be trusted more than that of a region which overlaps multiple objects. Hoiem et al. suggest learning a classifier to predict the 'homogeneity' of a region with respect to the class labels. If we consider region homogeneity to be a measure of the likelihood of a particular region, $P(r_i^s|I)$, then we can define segmentation integration method 2 as:

$$P(c_i = k|I) \propto \sum_s P(r_i^s|I) P(c_i^s = k|r_i^s, I) \tag{7.2}$$

To compute region homogeneity $P(r_i^s|I)$, we train a classifier consisting of boosted decision trees using the logistic formulation of AdaBoost [24, 47]. We use 20 trees with 16 leaf nodes each to avoid over fitting. As in our object classifier, we use region features of normalized average position (2D), a color histogram (100D) and the RCH (300D). In addition, we add the region size divided by the image size (1D), and the number of IofRs the region contains (1D). This gives us a 404-dimensional feature vector to classify.

In summary, the object recognition and segmentation procedure is as follows:

1. Extract regions using multiple image segmentation algorithms per image.
2. Extract features from each region.
3. Learn which region features predict each object.
4. Optionally, learn which regions are reliable (homogeneous).
5. On a test image: combine the region information for each IofR using either Equation 7.1 or Equation 7.2 to compute $P(c_i = k|I)$ for each class k.
6. Classify each IofR as the most likely object, $k^* = \mathrm{argmax}_k P(c_i = k|I)$.

The following results show that this simple approach in fact produces excellent results, outperforming the more complex Textonboost approach [106] on the MSRC 21-class data set, and performing comparably to the more complex approach by Verbeek and Triggs [119]. By adapting the components of our system to the strengths and weaknesses of image segmentation we have reduced the need for algorithmic complexity.

The first results presented are of quantitative performance, given in the bottom two rows of Tables 7.1 and 7.2. Table 7.1 contains results for the MSRC 21-class data set. The top section of the table presents results for the entire data set in the form of pixel accuracy averaged over the object classes and pixel accuracy averaged over all pixels. Our approach outperforms the Textonboost approach [106] in both overall and class-averaged accuracy. The approach of Verbeek and Triggs [119] gives better performance with respect to the class-averaged pixel accuracy, however our approach outperforms theirs with respect to overall pixel-averaged accuracy. (Note, however, that Verbeek and Triggs use a different split of the data set into training and test groups.) The bottom section of the table presents the pixel accuracy on each object class. The performance rank of each approach is class-dependent; each approach is superior on at least one class.

Table 7.2 presents results on the PASCAL VOC2007 segmentation challenge data set. The measure used to judge the challenge was class-averaged pixel accuracy. The only entry into the segmentation challenge was the Oxford Brookes entry [64], which we surpass in class-averaged pixel accuracy by 11%. Other entries into the challenge, including the winning TKK entry [120], were actually entries into the object *detection* challenge, and as such used a much larger training set of thousands of images hand-labeled with bounding boxes around objects. These results are not directly comparable to our own, but are included in Table 7.2 for reference. Notice that despite the higher class-averaged performance of the TKK algorithm, our approach is superior on a subset of the object classes.

We present the class-averaged accuracy as it was the official measure used by the PASCAL VOC2007 segmentation challenge. However, it can also be instructive to look at the overall pixel accuracy, especially since the background in this data set is considered a separate class. We can see a common trade-off between performance on specific object classes and performance on a very large background class. The Oxford Brookes approach has 77.7% accuracy on the background at the expense of other object classes, lowering their class-averaged accuracy. The overall pixel accuracy of this approach is much higher than the class-averaged accuracy at 58.4%. Our approach compromises at 59.2% accuracy on the background but shows higher accuracy on the object classes, increasing our class-averaged accuracy. However, our overall pixel accuracy is 50.1%. The TKK approach shows poor differentiation between the classes and the background with only 23% accuracy on the background, leading to an overall pixel accuracy of only 24.4%. The overall pixel-accuracy rankings are directly opposite to the class-averaged accuracy rankings of the approaches. Care should be taken in future challenges to properly evaluate performance.

Qualitative results are presented in Figure 7.5 on a sample of images from the MSRC 21-class data set, and in Figure 7.6 on a sample of images from the PASCAL

VOC2007 segmentation challenge data set. Both figures show examples of successes, partial successes, and failures. The first left-most columns contain the original images, and the second columns the ground truth segmentations with their original color coding. The third columns present maps of the most likely class label at each pixel (generated using the best combination method for that data set) and the fourth column in Figure 7.5 presents the probability of the most likely class at each pixel, $P(c_i = k^*|I)$. For comparison, the right two columns present results using single segmentations only. The second column from the right shows one of the better results from the single segmentations, while the right-most column shows one of the worst results from the single segmentations. Additional results for the PASCAL VOC2007 data set showing all 18 object maps resulting from the individual segmentations in addition to the map generated by integrating multiple segmentations can be found in Figures 7.7 and 7.8.

Our key goal was to create robustness to poor segmentations, and both combination methods achieve this on both data sets. Their class averaged accuracies are higher than the best individual segmentation. This can also be confirmed qualitatively. In addition, both methods outperform the best individual segmentations on the majority of object classes. Most importantly, they are robust to the inclusion of poorly-performing segmentations. By combining multiple segmentations in a straight-forward manner, we are able to produce state-of-the-art performance on difficult data sets which would have otherwise required a careful choice of segmentation algorithm and parameters for each image.

Up to this point, we have not explicitly introduced any models which take advantage of information from neighboring regions. This may seem surprising given our discussion in Chapter 3 regarding the inability of a single region to completely define an object, and the resulting need for information from multiple regions. Indeed, in Chapter 8, we do enforce spatial consistency using a random field formulation. Why, then, do we obtain state-of-the-art results in this chapter without neighborhood information? The answer lies in the *implicit* neighborhood information in our features and framework. Firstly, the RCH features consider information in and *around* a region, which results in neighboring regions' features having some similarity. Secondly, the multiple segmentation framework in this chapter includes regions created at different scales and using different features. If neighboring IofRs come from the same object or part, they are more likely to belong to the same region in at least one of the segmentations, a region which affects both of their classification scores and smooths the final object classification map.

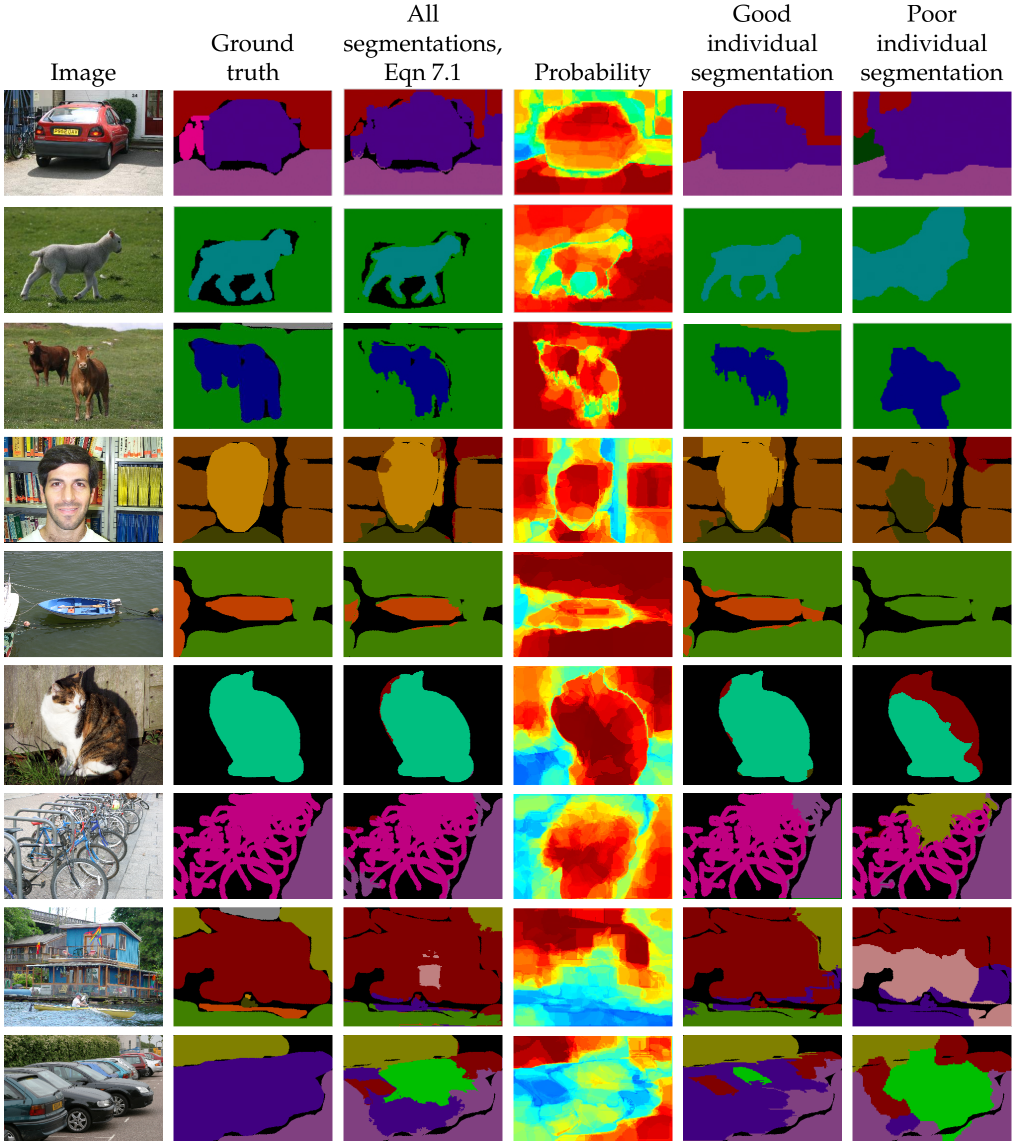


Image
Ground truth
All segmentations, Eqn 7.1
Probability
Good individual segmentation
Poor individual segmentation

FIGURE 7.5. Examples of object labeling results on the MSRC 21-class data set generated using single and multiple image segmentations. The left column shows the original image, while the second-from-left shows the ground truth with classes labels. The third column shows the most likely class using all of the segmentations combined with Equation 7.1. The black pixels are denoted 'void' in the ground truth, they are *not* generated by our method. The fourth column shows the probability of the classifications in the third column. The fifth column contains one of the better results from the 18 individual segmentations, and the sixth column shows a poor result from an individual segmentation. For clarity, we omit the other 16 individual segmentations. The top seven rows show promising results, while the bottom two rows show failures.

	class avg	**pixel avg**
Textonboost [106]	**57.7**	**72.2**
Verbeek [119]	**64.0**	**73.5**
Worst single seg	**49.6**	**63.3**
Best single seg	**59.8**	**72.2**
All segs, Eqn 7.1	**60.3**	**74.28**
All segs, Eqn 7.2	**59.9**	**74.15**

	building	grass	tree	cow	sheep	sky	aeroplane	water	face	car	bike	flower	sign	bird	book	chair	road	cat	dog	body	boat
Textonboost [106]	61.6	97.6	86.3	58.3	50.4	82.6	59.6	52.9	73.5	62.5	74.5	62.8	35.1	19.4	91.9	15.4	86.0	53.6	19.2	62.1	6.6
Verbeek [119]	52	87	68	73	84	94	88	73	70	68	74	89	33	19	78	34	89	46	49	54	31
Worst single seg	48.3	80.2	69.4	50.7	60.8	86.9	72.8	70.8	57.4	47.4	55.9	33.9	27.5	15.3	75.1	15.8	76.3	28.4	16.5	40.1	11.2
Best single seg	61.4	88.8	79.0	56.6	65.8	91.5	80.9	79.9	67.1	62.9	66.1	51.9	30.6	25.6	88.1	21.6	79.5	51.6	31.8	45.1	30.1
All segs, Eqn 7.1	67.8	92.1	81.4	57.7	64.7	94.6	83.6	80.9	74.9	65.1	68.4	52.8	34.5	23.4	84.5	15.8	83.0	47.6	29.2	48.2	14.9
All segs, Eqn 7.2	67.9	92.2	81.1	57.2	63.2	94.6	82.2	80.9	75.6	64.8	66.7	53.5	33.7	23.4	84.4	16.1	82.9	47.2	30.0	46.2	13.5

TABLE 7.1. Pixel accuracy results on the MSRC 21-class data set. For each class we show the pixel accuracy of the object segmentations. In the top section, the first column gives the class-averaged accuracy and the second column gives the overall pixel-averaged accuracy. We compare our approach with that of Textonboost [106] and of Verbeek and Triggs [119]. (Note, however, that Verbeek and Triggs [119] use a different split of the data into training and test sets.) Our approach gives superior performance to Textonboost, and comparable performance to Verbeek and Triggs. There is a large discrepancy between the worst and best individual segmentations in the top section, rows two and three. The combined segmentation methods in rows four and five manage to ignore the incorrect single segmentations and perform slightly better than any one segmentation alone. The bottom section shows the average pixel accuracy on each object class.

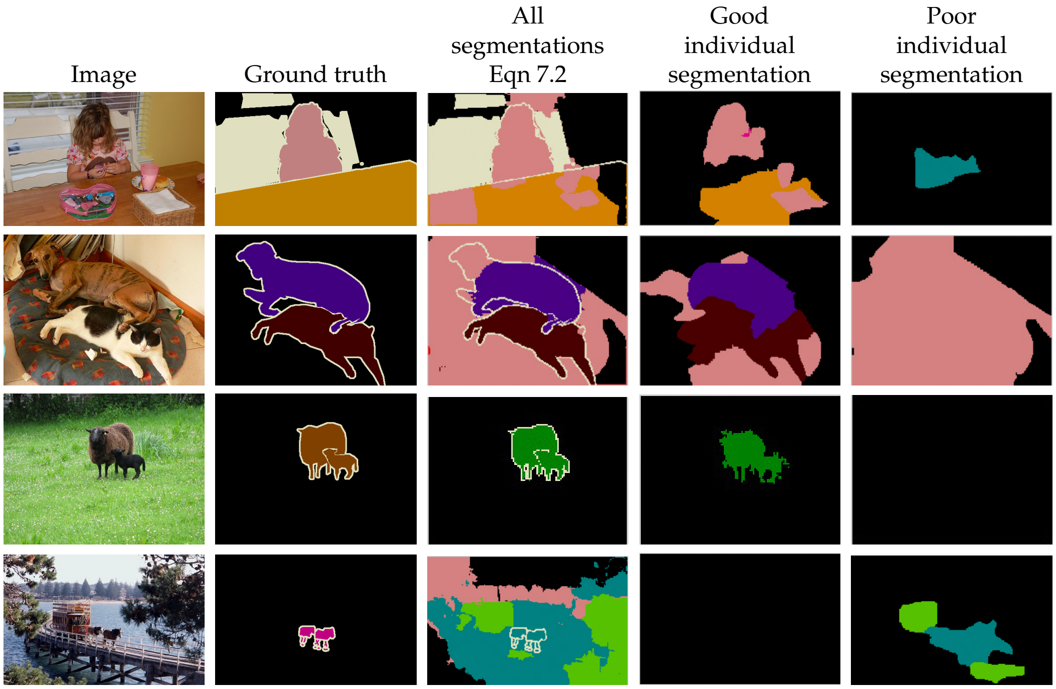


Image
Ground truth
All segmentations Eqn 7.2
Good individual segmentation
Poor individual segmentation

FIGURE 7.6. Examples of object labeling results on the PASCAL VOC2007 data set generated using single and multiple image segmentations. The first column on the left shows the original image, and the second column shows the ground truth labeling. The third column shows the most likely class using all of the segmentations combined with Equation 7.2. The beige pixels are denoted 'void' in the ground truth, they are *not* generated by our method. The fourth column contains one of the better results from the 18 individual segmentations, and the fifth column shows a poor result from an individual segmentation. For clarity, we omit the other 16 individual segmentations. The first result is promising, with the girl and most of the table correctly labeled. The second row also shows a good result on the very difficult dog and cat data sets. However, the background is classified as 'person' instead of 'background'. The third row shows a perfect segmentation but incorrect classification as 'cow', likely due to the relative scarcity of brown sheep [1]. The final row shows a complete failure.

	class avg	background	aeroplane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	dining table	dog	horse	motorbike	person	potted plant	sheep	sofa	train	tv/monitor
Oxford Brookes [64]	**8.5**	77.7	5.5	0.0	0.4	0.4	0.0	8.6	5.2	9.6	1.4	1.7	10.6	0.3	5.9	6.1	28.8	2.3	2.3	0.3	10.6	0.7
TKK [120]	**30.4**	22.9	18.8	20.7	5.2	16.1	3.1	1.2	78.3	1.1	2.5	0.8	23.4	69.4	44.4	42.1	0.0	64.7	30.2	34.6	89.3	70.6
Worst single seg	**12.7**	70.9	10.1	7.2	0.2	0.9	8.1	29.4	1.5	13.7	2.7	0.1	6.5	0.0	12.6	20.2	50.3	0.0	4.6	8.7	10.8	8.0
Best single seg	**18.2**	60.1	15.2	0.6	11.7	0.2	2.2	29.4	11.1	17.5	3.5	4.3	27.6	7.3	23.4	12.5	79.0	7.5	15.5	1.3	21.4	31.8
All segs, Eqn 7.1	**19.1**	55.3	27.9	0.4	8.5	1.8	1.2	32.9	13.3	16.9	3.0	8.4	31.0	8.5	23.2	16.4	80.1	7.8	18.7	1.4	28.1	17.4
All segs, Eqn 7.2	**19.6**	59.2	26.7	0.6	7.9	1.6	1.2	32.2	13.5	13.7	3.5	8.3	32.2	8.5	24.3	14.9	80.7	10.5	26.0	1.3	28.3	16.5

TABLE 7.2. Pixel accuracy of our object segmentation results on the PASCAL VOC2007 segmentation challenge data set. Evaluation was performed using the same criteria used in the PASCAL VOC2007 segmentation challenge. The first column gives the class-averaged accuracy, with the other columns showing results for the individual classes. We compare our approach with that of the Oxford Brookes entry into the competition [64], which used only the segmentation challenge training data set with the provided object masks. All of the other entries into the competition used the entire detection training set of thousands of images. Our overall accuracy is much higher than that of Brookes. Also, as in the MSRC data set, note the large variation between the worst and best individual segmentations. The overall accuracies of the combined segmentation methods outperform all of the individual segmentations.

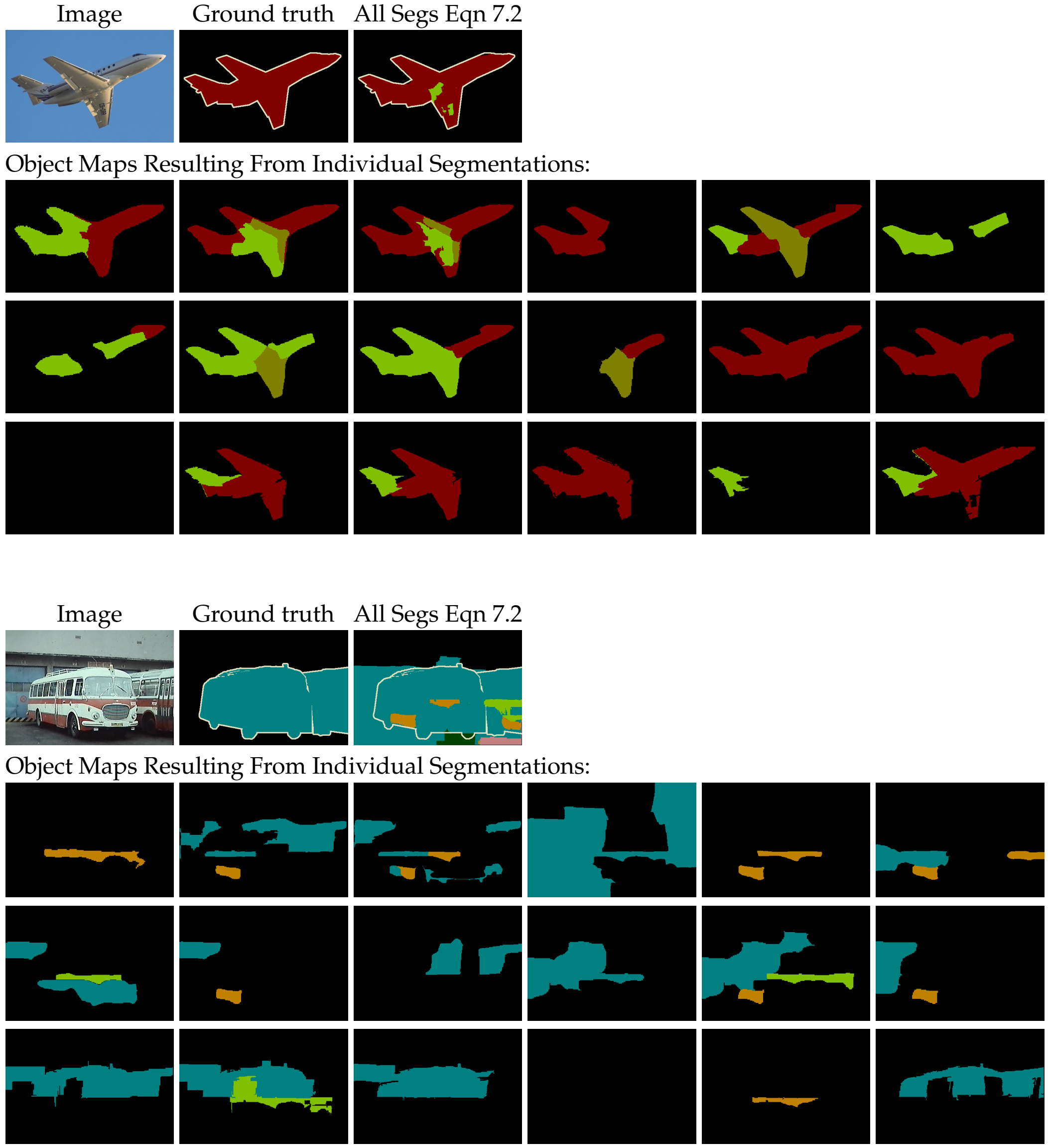


Image
Ground truth
All Segs Eqn 7.2
Object Maps Resulting From Individual Segmentations:
Image
Ground truth
All Segs Eqn 7.2
Object Maps Resulting From Individual Segmentations:

FIGURE 7.7. Examples of object segmentation results on the PASCAL VOC2007 data set generated using single and multiple image segmentations. For each example, the top-left image is the original, the top-middle is the ground truth labeling and the top-right shows the most likely class using all of the segmentations combined with Equation 7.2. The beige pixels are denoted 'void' in the ground truth, they are *not* generated by our method. The following three rows show the resulting object labeling for each individual segmentation alone.

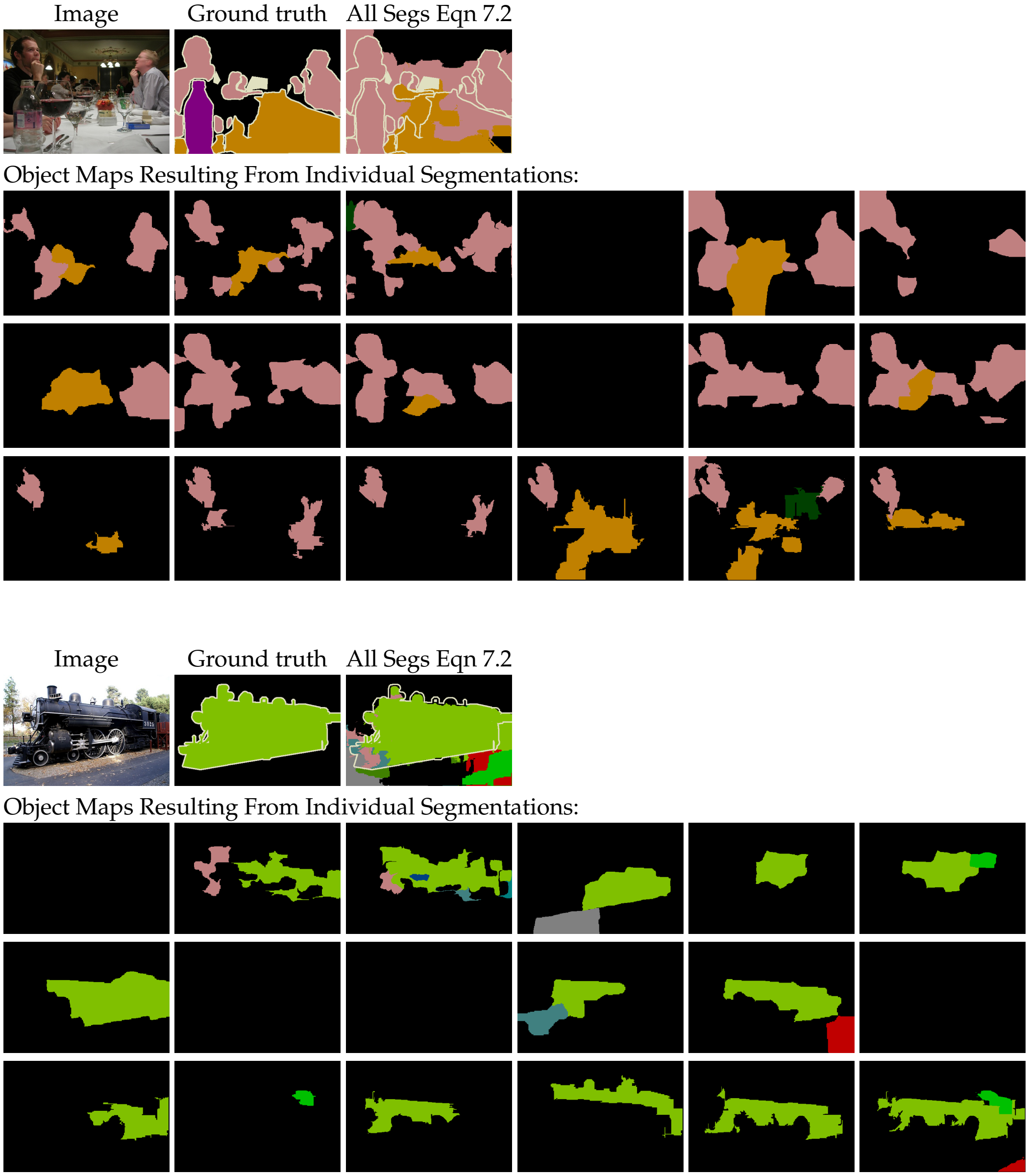


Image
Ground truth
All Segs Eqn 7.2
Object Maps Resulting From Individual Segmentations:
Image
Ground truth
All Segs Eqn 7.2
Object Maps Resulting From Individual Segmentations:

FIGURE 7.8. Examples of object segmentation results on the PASCAL VOC2007 data set generated using single and multiple image segmentations. For each example, the top-left image is the original, the top-middle is the ground truth labeling and the top-right shows the most likely class using all of the segmentations combined with Equation 7.2. The beige pixels are denoted 'void' in the ground truth, they are *not* generated by our method. The following three rows show the resulting object labeling for each individual segmentation alone.

5. Incorporating weakly labeled data

So far our discussion of using multiple segmentations has assumed fully labeled training data which contains a precise object mask for each object class in each image. However, as we have previously discussed, full supervision is expensive and hence not scalable to large data sets. Human impatience and inaccuracy can even be seen in the two data sets used in this chapter which both contain 'void' labels around many object boundaries, indicating that precise labeling was too burdensome. To increase the amount of available data, we look once again to a weakly labeled data set. The PASCAL VOC2007 data set contains many images not in the segmentation challenge which are labeled with bounding boxes around objects, but not precise object masks. In this section, we will explore two ideas for incorporating some of these additional weakly-labeled bounding boxes.

5.1. Using object detection to guide object segmentation

Training data sets containing ground truth bounding boxes or image-level object labels are well-suited to training weak classifiers such as bounding box object detectors or image classifiers. Since a significant amount of work has been done in the areas of object detection and image classification, it is worthwhile to consider whether this type of information could improve our own object recognition and segmentation results. As a preliminary study of this issue, we ask whether *perfect* bounding-box object detection could be used to filter our object segmentation results by disallowing object detections outside the bounding boxes. If layering object segmentation atop perfect object detection produces more accurate object masks than perfect object detection alone, then there is hope that object detection results from actual systems could be useful. In addition, showing that the hybrid system provides more accurate masks confirms that our object segmentation approach is reasonable.

Formally, the hybrid object masks are formed as follows. Let W_k be a map of the pixels in an image that are within a ground truth bounding box for object k. This is a map of perfect bounding box detection results, and can be thought of as a binary prior on the location of the object. Our confidence in class k at pixel q is:

$$T(c_q = k|I) = W_k(q)P(c_{i(q)} = k|I) \tag{7.3}$$

where $i(q)$ is the IofR that contains q. The ground truth bounding boxes are included in the PASCAL 2007 test data. As before, a multi-class object map is created by taking the most likely object class at each pixel, $K^*(q) = \mathrm{argmax}_k T(c_q = k|I)$. Since the bounding boxes in the PASCAL human annotations do not always correspond perfectly to the ground truth segmentation masks, our baseline bounding boxes have each dimension increased by 10% over the ground truth, and extra boxes around any remaining uncovered object pixels.

Perfect bounding boxes are unlikely to be attained by a real object detector, so we simulate the common practice of increasing the size of imprecise bounding box detections to increase the probability of finding an object. The experiment is repeated with increasingly larger bounding boxes of sizes 2.25, 4, 6.25, and 9 times the original sizes. In addition, bounding boxes filling the entire image are used, which is equivalent to image classification.

If the object masks K^* are generated using the baseline ground truth bounding boxes to create W_k, the class-averaged accuracy for the PASCAL VOC2007 segmentation challenge test set increases to 79.4%. As the sizes of the bounding boxes used to create W_k increase, the class-averaged accuracy monotonically decreases, finally resulting in accuracy of 58.9% when the bounding boxes fill the entire image (image classification). This is the best performance we can expect with perfect object detection or image classification.

We next examine whether our object segmentation provides more accurate object masks than the bounding boxes alone. Unfortunately, the bounding boxes cannot be converted into a single object map as multiple bounding boxes of different classes often overlap and there is no natural way to order them. So the multi-class object map K^* generated by our algorithm must be compared to each object classes' bounding box map separately. Consider one object class k. In K^*, some of the pixels labeled 'object' in W_k have been changed to a different object or background. To measure this change, we compute the following ratio. As the numerator, we compute the fraction of the pixels incorrectly labeled as 'object' in W_k that are correctly relabeled to a different class in K^*. As the denominator, we compute the fraction of the pixels correctly labeled as 'object' in W_k that are incorrectly relabeled to a difference class in K^*. Pixels labeled 'void' are ignored. This measure has two important qualities. First, since objects can take up variable amounts of the bounding boxes (consider thin cat tails versus box-like monitors), the measure considers the number of pixels which change labels from 'object' to 'non-object' as a fraction of the total number of object or non-object pixels in the box. Second, the measure provides a clear point of improved performance: if the ratio is above 1, then a higher fraction of the non-object pixels are correctly relabeled than object pixels incorrectly relabeled, and the resulting object masks are more accurate than the bounding boxes alone.

The experiment was performed for each bounding box size and the results are presented in Figure 7.9. The behavior of the measure broadly fell into three categories. The first few objects, for example the sheep class, saw increased improvement with increased size of the bounding boxes. Larger bounding boxes contain a greater number of non-object pixels, therefore it can be expected that object segmentation will relabel an overall larger percentage of non-object pixels in these larger boxes. For the final few objects, like the cat class, however, the ratio actually decreased with increasing bounding box size. We speculate that this is due to confusion between classes; as the bounding boxes of other classes grow, they encroach on the true cat pixels, and the cat is misclassified as another object. The middle objects, such as the tv/monitor class, consist of more difficult objects which show little change between bounding box sizes. Most importantly, all of the bars on every plot reached above 1, showing that our method always improves object segmentation compared to bounding boxes. The patterns we have shown are interesting in themselves as they suggest that providing a weaker prior, such as image classification, can still provide large overall improvement, but it may come at the expense of certain individual classes. As improved image classification and object detection systems emerge, it will be important to revisit this issue and compare future results to the 'ideal' situation presented here.

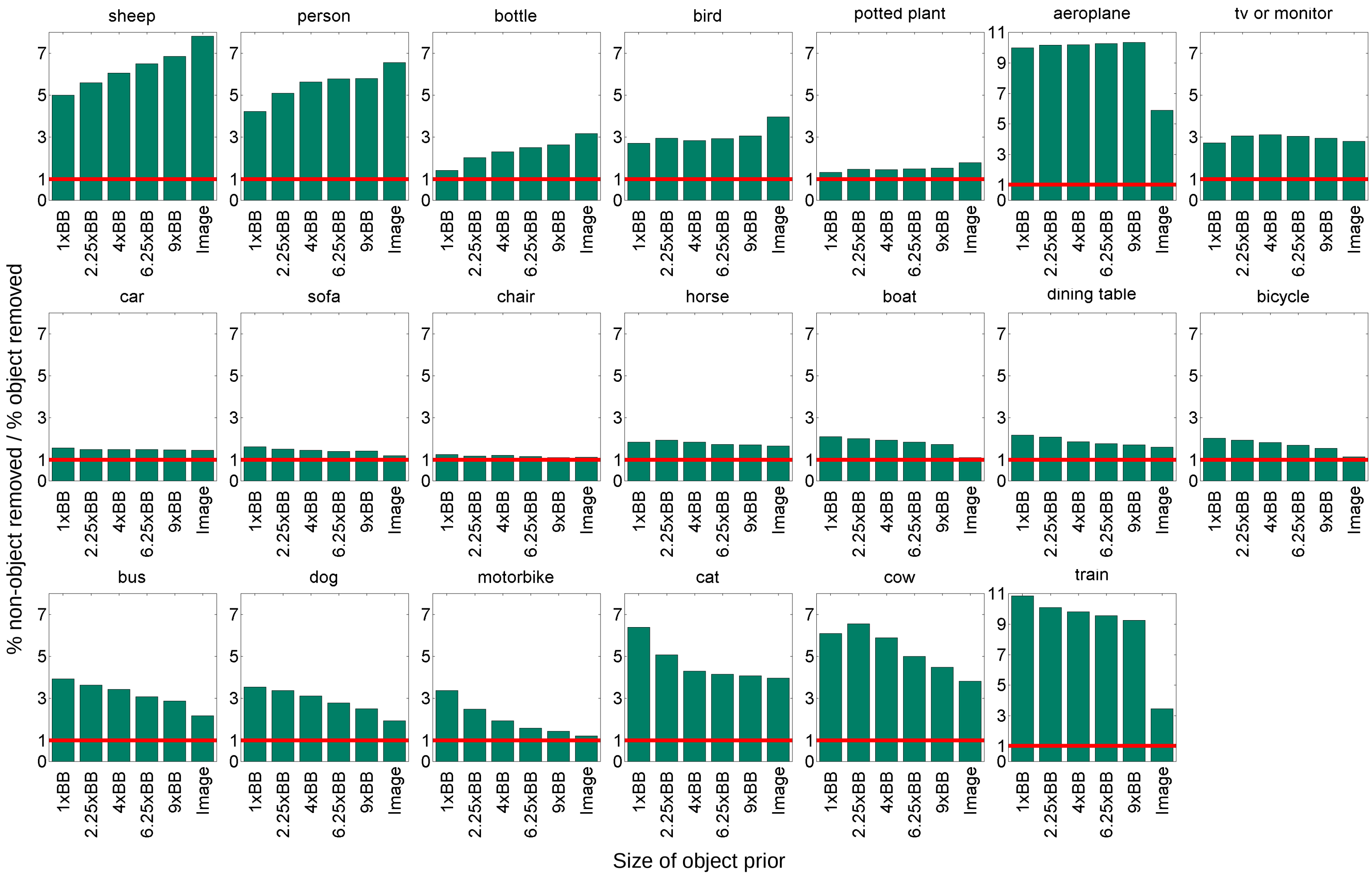


FIGURE 7.9. The effects of using various sized bounding boxes as priors for object segmentation on the PASCAL2007 data set. The bars show the improvement in object mask accuracy using our approach versus bounding boxes alone. The x-axis represents the bounding box size, from the ground truth size to the entire image. The y-axis represents the percent of non-object pixels in the boxes correctly relabeled from 'object' to 'non-object' by object segmentation, divided by the percent of object pixels that are incorrectly relabeled. All of the bars are above 1, showing that object segmentation relabeling of bounding box pixels from 'object' to 'non-object' does improve object masks.

5.2. Weak labels as noisy labels

Weakly labeled data can also be incorporated into our multi-segmentation approach by treating the weak labels as noisy labels. For the PASCAL VOC2007 data set, we augmented the segmentation challenge training set by adding 400 randomly selected training images from the detection challenge, increasing the total training set to 822 images. These additional images did not have ground truth segmentation masks, but they did have ground truth bounding boxes surrounding each object. We performed two sets of experiments, one using the bounding boxes as weak training labels, and another using only image-level object labels without localization (mirroring weak training in previous chapters of this document.)

The new training set was used to learn the individual region classification probabilities. We did not use the extra images to relearn the homogeneity measure since the noise in the bounding box labels lies around the object outlines, exactly where regions may encompass more than one object, making the data useless for training a classifier whose sole purpose is to model such heterogeneous regions. To make the results comparable to the fully supervised case, the rest of our training procedure, and our testing procedure, was kept the same.

Two improved qualitative results from using the training set augmented with bounding box-labeled images can be seen in Figure 7.10. The fully supervised but smaller training set resulted in complete misclassifications of the boat and train in the images. The augmented data set, although noisy, contained more examples of each class and led to correct object identification.

Quantitative results are presented in Figure 7.11. The three sets of bars show the class-averaged accuracy results for, from left to right, the worst single segmentation performance, the best single-segmentation performance, and the performance of segmentation combination 2. The dark green bars represent the original training set, the light green bars the extended set with image-level weak labels, and the yellow bars represent the augmented set with bounding box weak labels. We can see that both the performance of the best segmentation and the combination of segmentations increases by 3-5% when using either augmented set. We can conclude that increasing the size of the training set is indeed useful. In addition, weak image-level labels for the extra data seem to be sufficient.

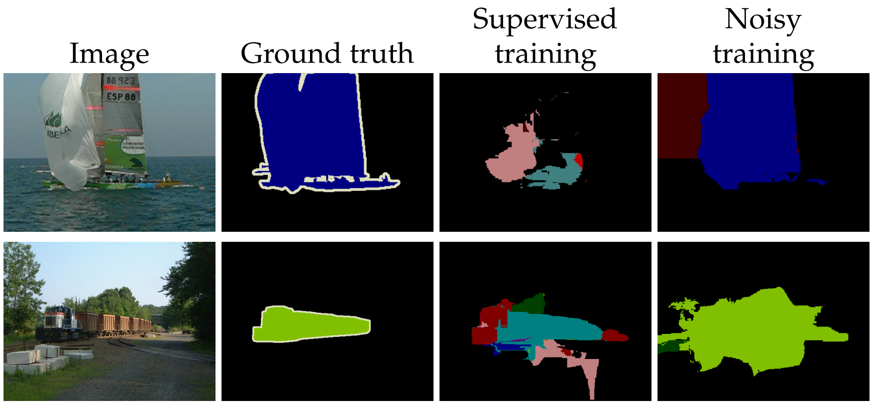


FIGURE 7.10. Examples from the PASCAL VOC2007 data set of improved object recognition using an augmented training set of 422 fully segmented images and 400 weakly segmented images (bounding boxes) as in Subsection 5.2. The classifier trained using the fully supervised training set alone does not correctly recognize any part of the objects (although it does recognize the background). On the other hand, the augmented classifier does label the objects correctly.

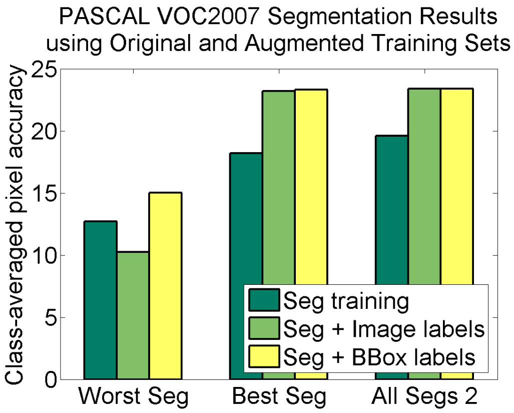


FIGURE 7.11. Class-averaged pixel accuracy on the PASCAL VOC2007 using the 422 fully segmented training images, and augmenting the training set with 400 images with weak image labels (no localization), and weak bounding box labels. Using a relatively small amount of additional, weakly labeled data, the results improve by almost 4%.

6. Conclusions

In this chapter, we have presented an intuitive method for integrating information from multiple image segmentations to produce object recognition and segmentation results. By utilizing multiple bottom-up image segmentations, we were able to use segmentation variability to our advantage and provide robustness to poor image segmentations. Experimental results were provided on the MSRC 21-class data set, as well as the PASCAL VOC2007 segmentation data set.

We have also suggested two paths for augmenting a training set using weakly supervised training data. The first method uses weakly labeled data to train an object detector or image classifier, whose output provides a prior for our object segmentation. We studied an ideal version of this method by using the ground truth bounding boxes on the test set as our object prior. The second method for augmenting the training set uses weakly labeled data and treats the labels as noisy object masks. We showed that augmenting the training set with images containing either bounding box labels or image-level labels provides a substantial improvement in performance.

7. Contributions

- An intuitive method for combining multiple segmentations into our object recognition approach.
- Experimental validation that multiple segmentations used in concert can improve results over individual segmentations and increase robustness to poor outlier segmentations.
- Two proposed methods for increasing the training set size by adding weakly labeled images.

CHAPTER 8

INCORPORATING SPATIAL INFORMATION

We have made progress toward recognizing and segmenting objects from their background, however we can still make mistakes. One type of error is the misclassification of a region despite the fact that all of the regions in its vicinity are properly classified, such as in column (b) of Figure 8.1. This phenomenon points to a lack of spatial consistency in our region classification. If we could model the spatial relationships between the regions in an object, perhaps we could smooth the object maps to provide improved masks as given in Figure 8.1 column (c). Indeed, the cumulative information from multiple regions could improve the classification of all the regions.

In previous chapters, our approach to accumulating information over multiple regions was implicit in nature. In Chapter 5, our approach to accumulating spatial information was through the RCFs, which include information from interest points not only inside but *around* a region. The information they capture, however, is still relatively local and does not explicitly enforce any spatial smoothness in the final labels. In Chapter 7, we incorporated information from multiple segmentations computed with different algorithms, parameters, and on different image scales. The different spatial extents of the regions served as a method of smoothing the labeling, the information from larger regions would smooth the information from the smaller regions they encompassed. However, heterogeneous object parts such as a person's shirt and pants in different colors would likely belong to separate

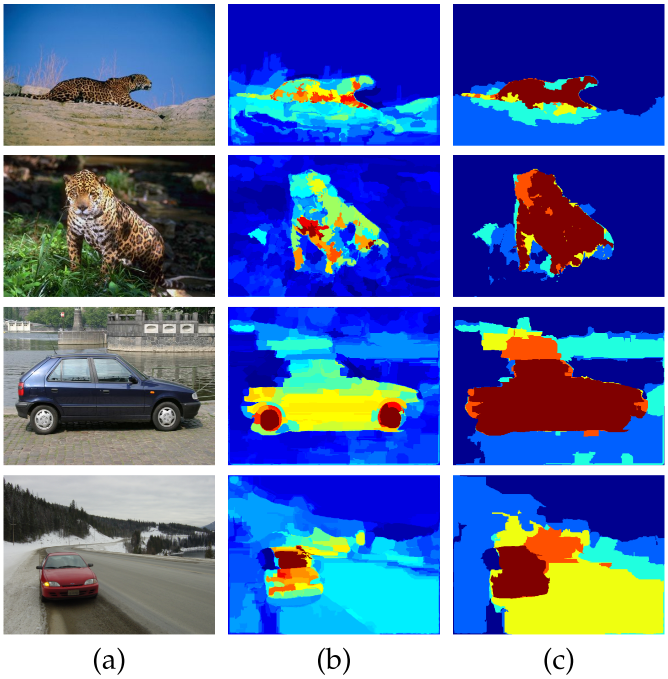


FIGURE 8.1. Qualitative results of enforcing spatial consistency on the Spotted Cats and PASCAL VOC2006 Cars data sets. Column (a) shows the original images, (b) the results of using single region classification with feature TR(1000,300) for the Spotted Cats and RCFs for the Cars, and (c) shows the results of incorporating neighborhood information.

regions in every segmentation, so this smoothing effect would not take place. In this chapter, we will look at *explicitly* combining information from multiple regions.

In order to gather information from multiple regions, we first need to decide which regions to consider. In other words, we need a model of the spatial relationships between an object's parts or features. For some object classes, the degree of intra-class variability in the spatial organization of their parts can be quite small, for example in the grid-like texture of a chess board or the parts of a hammer. However, the variability in spatial organization can also be very high, such as in deformable objects like Transformer toys, objects from visually ill-defined classes such as chairs, or objects which are both deformable and have an ill-defined visual class description like cats. In order to cover the range of deformations, we require a spatial model which is flexible and yet informative.

Another difficulty in defining spatial relationships is the instability of region shapes. As previously discussed, region centroids are unstable and an unnatural measure of region position. In addition, the subdivision of an object into regions may change drastically between segmentations. Finally, we would like to choose a model which could be trained using only weakly labeled data. To satisfy all of these constraints, the spatial relationship we choose to model is region adjacency, and our framework for smoothing region information will be a pairwise random field on the regions.

1. Related Work

Before describing our methods for incorporating spatial information into our classification model, we describe previous work in this area. We first mention some of the more prominent approaches to spatial modeling using points or patches, and explain why they cannot be extended to the region framework. We then discuss current approaches to using sets of regions, or smoothing segmentations.

The constellation of parts approach [14, 38] provides a flexible model of the relationship between object parts. However, it is expensive and can only realistically model a few parts. In addition, it requires knowledge of the part centers to define a spatial relationship. Using K-fans [26] to model spatial relationships also requires knowledge of the patch center, and requires K to be high to model highly deformable objects. Pictorial structures [37] suffer from many of the same problems as the previous methods.

The Textonboost approach in [106] requires that image structure have a repeatable location within an object patch, and requires fully supervised training data.

Learning an object shape for which interest points or object parts can vote, as in [3, 67], is impractical in a weakly supervised framework since we do not know the shape of the training objects. The LOCUS framework [128] combines low-level appearance cues with top-down object shape information from edges making it susceptible to occlusion issues and requires that objects be somewhat rigid.

In another approach, both Ferrari et al. [40] and Lazebnik et al. [65] define sets of patches by the consistency of their affine transformations with an exemplar set.

It is unclear, however, how to measure the affine consistency of irregular segmentation regions. Ferrari et al. [40] also introduce the notion of image exploration, or finding strong features and then searching for weaker ones given an exemplar set, but the resulting object maps are still noisy. Yu and Shi [131] attempt to combine knowledge from square image patches with edge information, however they require manually labeled ground truth data and can only identify 15 specific objects, not object classes.

Originally, approaches which grouped regions focused on region similarity-based grouping and edge-based separation [71,102,104]. Hoiem et al. [53] segment an image into superpixels and then create multiple superpixel sets by randomly seeding each set with a superpixel and then adding similar superpixels. These methods are useful for producing cleaner segmentations, however they do not incorporate class-specific top-down knowledge and hence cannot group visually different parts into a whole object. The approach by Borenstein and Ullman [12] takes the opposite approach, using only top-down image patches from an object class and trying to piece them together over the image to obtain an object outline, thereby avoiding the unsupervised segmentation stage altogether. They require fully-supervised training data, however, and the use of image patches hinders generalization.

More recently, methods have started to appear which combine image regions into larger object parts or the entire object using top-down information. One class of approaches uses the learned shape of the object or object parts as a cue for grouping by biasing the final segmentation to follow object or part boundaries [11,60,68]. However, learning these shape models requires segmentations of the training data, derived either manually [11, 68] or through additional cues (i.e. rigid motion in video) [60].

In [51], He et al. use image context to build a mixture of Conditional Random Fields (CRF) model with superpixels as nodes. Their pairwise potentials model context-specific label compatibility, as well as the strength of the edge between the two regions. Their approach shows promise in a multi-class environment, however they still require fully labeled data. Also, the use of edge information in the pairwise potentials prevents visually different parts of the same object from clustering together, and hence sample results are shown on mainly distant or homogeneous objects. Kumar et al. [62] also make use of a CRF over image regions. We join these approaches in using a random field over regions.

2. Spatial consistency within a weakly supervised framework

The first component of our study in using multi-region information will be to build on the weakly supervised framework first introduced in Chapter 5 applied to two-class problems (one object class plus a background class). To model the information between adjacent pairs of regions while enforcing global labeling consistency, convert our two-class region classification problem into an energy minimization and look to a random field formulation with region pair cliques. Let $c_i \in \{0, 1\}$ be the label of region i, with label 0 representing the background and 1 the object. Then $\mathbf{C} = [c_1..c_N]$ is the vector of label assignments to every region in an image. Let F_i be the feature cluster for region i, and $\mathbf{F} = [F_1..F_N]$ be the vector of region features over the entire image. Here F can be any of the features we have discussed, such as an RCF. We define the energy of an assignment of labels C to an image to be:

$$E(\mathbf{C}, \mathbf{F}) = \sum_i U(c_i, F_i) + \alpha \sum_i \sum_j B(c_i, c_j, F_i, F_j) \tag{8.1}$$

Provided that the pairwise term $B(c_i, c_j, F_i, F_j)$ is associative, we can use graph cuts to minimize the energy exactly and obtain a region labeling [59].

In the energy function, the unary term serves to classify single regions. By using a generative model, we can define the unary energy term as:

$$-\log P(F_i|c_i) = -\log P(F_i|c_i{=}0)^{(1-c_i)} P(F_i|c_i{=}1)^{c_i} \tag{8.2}$$

$$= -c_i \log \frac{P(F_i|c_i{=}1)}{P(F_i|c_i{=}0)} - \log P(F_i|c_i{=}0) \tag{8.3}$$

Since $\log P(F_i|c_i = 0)$ is constant with respect to the label assigned to c_i we can ignore it in the energy minimization. Also, as in the single-region classifier, we can use the bounded posterior $\tilde{R}(F_i)$ instead of the unbounded log likelihood ratio and approximate $\tilde{R}(F_i)$ using the *image* labels instead of the *region* labels. Finally, energy minimization via graph cuts produces a binary labeling of the data, but it does not produce marginals. To compute different precision-recall values, we use a penalty term β which provides a simple method for adjusting the threshold between declaring a region part of an object or the background. A more involved method for achieving this goal is presented in [58]. Our final unary term is:

$$U(c_i, F_i) = -c_i \left(\tilde{R}(F_i) - \beta \right)$$

We now need to define the binary energy term. In most energy minimizations, the binary energy term is a classifier between the four possible assignments $(c_i, c_j) \in \{(0,0), (0,1), (1,0), (1,1)\}$ trained using fully labeled data. Since we only have image-level labels, defining and training the binary energy term is a more difficult process. We choose to define $B(c_i, c_j, F_i, F_j)$ as a smoothing term which differentiates between two regions having the same label or different labels. We approximate the probability of neighboring regions belonging to the same class by assuming that regions which are frequently seen as neighbors in the training data are more likely to come from the same class (object or background) than features which are more often seen apart. In the notation of our energy minimization, this binary potential relationship can be written as:

$$B(c_i, c_j, F_i, F_j) \quad = \quad \begin{cases} \log\left(1 + \frac{P(F_p = F_i \wedge F_q = F_j | p \bowtie q)}{P(F_p = F_i \vee F_q = F_j | p \bowtie q)}\right) & c_i \neq c_j \\ 0 & c_i = c_j \end{cases} \tag{8.4}$$

Where $p \bowtie q$ means that region p is adjacent to region q. The $1+$ term ensures that $B >= 0$ and hence is associative. By this definition of B, we prefer to smooth region labels if the regions are frequently adjacent in the training data and hence are likely to belong to the same object.

We present experiments using this energy function on the Spotted Cats and PASCAL VOC2006 Cars data sets also used in Chapter 6. We chose the best-performing single features in Chapter 6 for each class: the TR(1000,300) texture features for the Spotted Cats and the RCFs for the Cars. Values of α of 0, 0.5, 1 and 5 were used. In Figure 8.2 we show the results of setting $\alpha = 0$ and $\alpha = 5$ for each data set. We omit the $\alpha = 0.5$ and $\alpha = 1$ settings for clarity since they lie strictly between the $\alpha = 0$ and $\alpha = 5$ curves. As in Chapter 6, the plots show the accuracy of

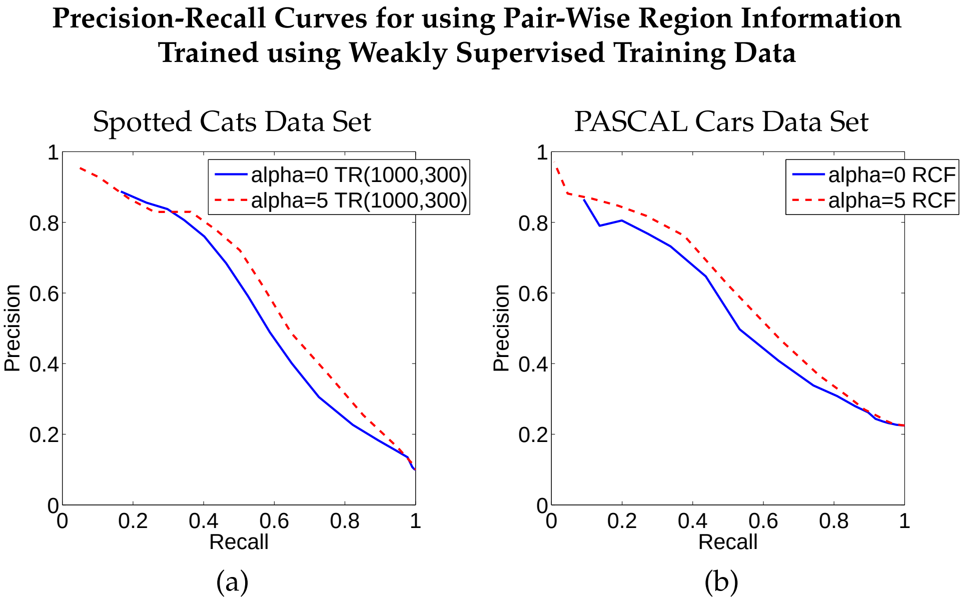


FIGURE 8.2. Pixel-level precision-recall curves for the (a) Spotted Cats and (b) Cars datasets using pairwise region information. The best-performing single-region representations were used, TR(1000,300) for the Spotted Cats and RCFs for the Cars. Each curve shows the results using a different α as in Equation 8.1. For $\alpha = 0$ the binary energy term is ignored, and the results mirror those of single regions in Figure 6.1. From the $\alpha = 5$ curve we see that incorporating neighborhood information can indeed improve precision by 5-10% for many of the recall values.

the pixel-level object masks. If $\alpha = 0$, the binary term in the energy function is ignored, and indeed these curves match their respective curves in Figures 6.1 and 6.3 in Chapter 6. The $\alpha = 5$ curve demonstrates that using this pair-wise smoothing approach does indeed improve precision by 5-10% for most recall values. The qualitative examples given in Figure 8.1 demonstrate this improved accuracy on specific images, showing the reduction in spurious region classifications as well as the increased margin between object and background scores.

3. Spatial consistency using multiple image segmentations

We have seen that enforcing spatial consistency between the labels of neighboring regions has positive effects on the two-class problem in our weakly supervised framework. We now study whether it has the same effect on the multi-class problem presented in Chapter 7 using fully supervised training, specifically for the PASCAL VOC2007 segmentation challenge data set.

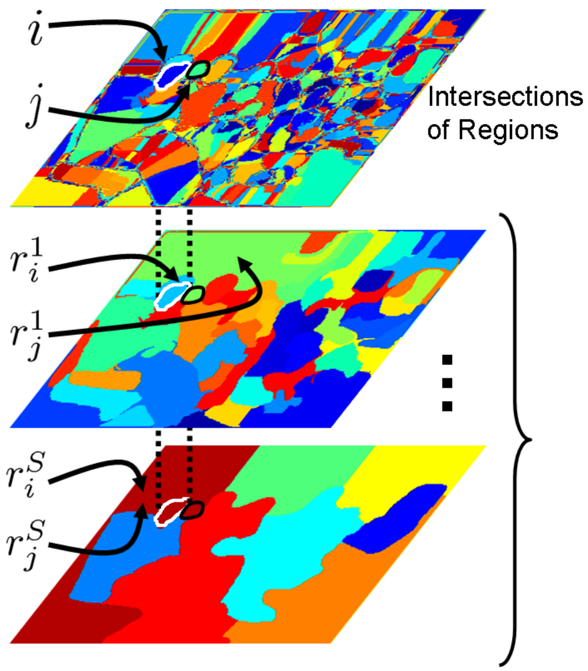


FIGURE 8.3. Example of a pair of intersections of regions i and j and their parent regions in two of the eighteen segmentations. In the first segmentation the IofRs come from different parent regions, but in the last segmentation they come from the same parent region.

Once again, let us define the energy function we wish to minimize using pairwise cliques as in Equation 8.1. In this case, however, the label c_i can take multiple values, and i and j are intersections of regions (IofRs) as in Figure 8.3. We redefine the functions U and B to take advantage of the fully supervised training. We define the unary potentials as $U(c_i, I) = -\log P(c_i|I)$ to penalize uncertainty in the label, which we can compute as in Chapter 7 using Equation 7.2. We use the notation I to denote image features, for example in the unary potential $I = F_i$. This avoids more cumbersome notation in the following equations.

The binary potentials are defined as follows to penalize discontinuity between adjacent labels [52]:

$$B(c_i, c_j, I) = \begin{cases} \alpha\,(\log p_{ij} - \log{(1 - p_{ij})}) & \text{if } c_i \neq c_j, \\ 0 & \text{otherwise.} \end{cases} \tag{8.5}$$

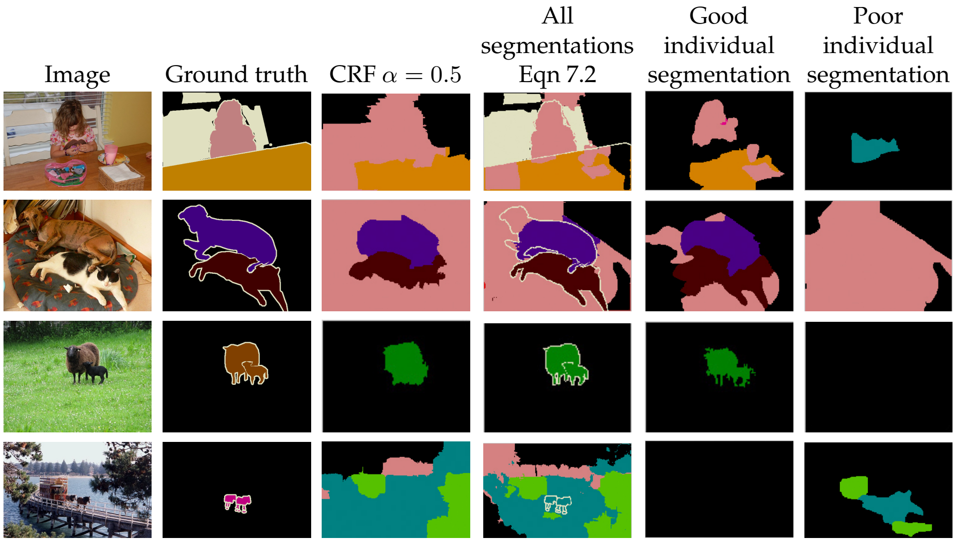


Image
Ground truth
CRF $\alpha = 0.5$
All segmentations Eqn 7.2
Good individual segmentation
Poor individual segmentation

FIGURE 8.4. Qualitative results of using region adjacency information on the PASCAL VOC2007 segmentation challenge data set. The first column gives the image, the second the human-labeled ground truth, and the third column gives the results of using the approach in this chapter. For comparison, columns four through six repeat the single region results presented in Chapter 7. Despite the small α used, the CRF seems to over-smooth consistently.

Where p_{ij} reflects the probability that two neighboring IofRs belong to the same class. We enforce that $B(c_i, c_j, I) \geq 0$ so that graph cuts with alpha-expansion can be used to minimize the energy [13].

The binary potentials between IofRs are computed by examining the likelihood that their parent regions in each segmentation belong to the same object. Once again, region features and relationships are computed using the parent segmentation regions, not the IofRs, to ensure adequate spatial support. As in the single-region case, we can combine the information from all of the segmentations. Following the notation in Chapter 7, we define p_{ij} to be:

$$p_{ij} = \frac{1}{S} \sum_s p_{ij}^s \tag{8.6}$$

$$p_{ij}^s = \begin{cases} 1 & \text{if } r_i^s = r_j^s \\ P\left(c_i^s = c_j^s | r_i^s, r_j^s, I\right) & \text{otherwise} \end{cases} \tag{8.7}$$

Like the homogeneity measure, the pairwise likelihoods $P\left(c_i^s = c_j^s | r_i^s, r_j^s, I\right)$ can be learned using logistic AdaBoost [52]. To learn the likelihood that two neighboring regions in a given segmentation come from the same object, we use the following features:

- To describe the union of the two regions we use:
  - the normalized average position of the union (2D),
  - the RCH of the union (300D), and
  - the color histogram of the union (100D).
- To compare the two regions we use:
  - the smaller region area divided by the larger region area (1D),
  - the symmetrical KL-divergence between the individual regions' RCFs scaled between 0 and 1 (1D),
  - the scaled symmetrical KL-divergence between the two color histograms (1D), and
  - the different in region position normalized by the image diagonal (1D).

Results of using this random field formulation on the PASCAL VOC2007 segmentation challenge can be seen qualitatively in Figure 8.4 and quantitatively in Table 8.1. For comparison, the other qualitative results in Figure 8.4 are repeated from Chapter 7 Figure 7.6, while other quantitative results in Table 8.1 are repeated from Chapter 7 Table 7.2. The efficacy of our formulation seems to be class-dependent. The accuracy on some of the classes, including birds, people and sheep, has improved. However, there seems to be undesired smoothing of thin, wiry objects. We hypothesize that the more accurate modeling from fully supervised training data, plus the implicit label smoothing from multiple segmentations, may have enforced much of the spatial consistency required.

	class avg	background	aeroplane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	dining table	dog	horse	motorbike	person	potted plant	sheep	sofa	train	tv/monitor
Oxford Brookes [64]	**8.5**	77.7	5.5	0.0	0.4	0.4	0.0	8.6	5.2	9.6	1.4	1.7	10.6	0.3	5.9	6.1	28.8	2.3	2.3	0.3	10.6	0.7
Worst single seg	**12.7**	70.9	10.1	7.2	0.2	0.9	8.1	29.4	1.5	13.7	2.7	0.1	6.5	0.0	12.6	20.2	50.3	0.0	4.6	8.7	10.8	8.0
Best single seg	**18.2**	60.1	15.2	0.6	11.7	0.2	2.2	29.4	11.1	17.5	3.5	4.3	27.6	7.3	23.4	12.5	79.0	7.5	15.5	1.3	21.4	31.8
All segs, Eqn 7.2	**19.6**	59.2	26.7	0.6	7.9	1.6	1.2	32.2	13.5	13.7	3.5	8.3	32.2	8.5	24.3	14.9	80.7	10.5	26.0	1.3	28.3	16.5
All segs, CRF	**19.3**	47.1	24.9	0.7	11.6	1.4	1.1	34.2	14.8	15.7	2.5	6.9	33.8	5.7	23.3	14.3	87.3	7.7	26.6	0.3	28.2	17.5

TABLE 8.1. Quantitative results of pixel accuracy on the PASCAL VOC2007 segmentation challenge data set. The first column gives the class-averaged accuracy. We compare our approach to enforcing spatial consistency using a random field formulation in the last row with single region methods from Chapter 7. The efficacy of the approach seems to be class-dependent.

4. Conclusions

In this chapter, we have presented a method for enforcing spatial consistency in our object masks. By modeling the relationships between pairs of adjacent regions, we can formulate our object recognition task as an energy minimization in a random field. Our formulation is applicable to both rigid and deformable objects, and can accept both fully and weakly supervised training data.

We saw promising improvements by using our formulation on the two-object-class problem trained with weakly supervised image labels. Spurious misclassified regions were indeed removed in many cases, and accumulated information from multiple regions increased the margin between object and background classification scores. Results were mixed, however, when using fully supervised training with multiple object classes and multiple segmentations. We speculate that the fully supervised training and multiple segmentations are sufficient to model many of the spatial relationships. In addition, the larger number of object classes may require a larger training set to learn useful region adjacency information.

5. Contributions

- A proposed framework for enforcing spatial consistency between region labels using region adjacency and a random field formulation. The framework is applicable to objects which are rigid or deformable, can utilize fully supervised or weakly supervised training data, and applies to two- or multi-class problems.

CHAPTER 9

FUTURE WORK: APPLICATIONS AND EXTENSIONS

In this chapter, we discuss two possible applications and extensions of our object recognition system which could be explored as future work.

The first extension involves the use of other sensors, such as laser range finders, to supplement the information in still images. We present preliminary work in this area showing that even a simple approach to sensor fusion can provide improvement in object recognition and localization.

The second extension discusses automatic methods for generating weakly labeled training data sets using the vast and partially labeled image collections now available. We propose that given a sufficiently large amount of data, training sets can be created on-demand, tailored to a particular query image. Such training sets could generate more discriminative classifiers, minimizing generalization error.

1. Fusing information from other sensor modalities

The work described in this thesis has been limited to utilizing information from still images. However, for many applications additional sensor data may exist. One notable example is 3D range data which is available in many robotic systems. In this section, we look at the scenario of outdoor urban robotics and the task of accurately recognizing and segmenting objects in the environment. As a first step, we describe a preliminary approach to combining 3D range data from LADAR with image data to extract pixel masks of building locations. This approach was developed with R. Unnikrishnan and published in [89]. We begin by discussing the extraction of scene information from each sensor modality separately, and then proceed to combine the predicted structures.

1.1. Extracting building regions from image data

We look for buildings in images in the same general fashion as our basic object recognition and object segmentation throughout this thesis: features are extracted from the image which are then used to segment the image into coherent regions, each of which is classified as to whether it belongs to a building to form a final object map.

The features we use for image segmentation are computed on non-overlapping blocks of 16x16 pixels. For each block, the center position (2D) and average color in L*u*v* space (3D) are computed. In addition, 'orientograms' are computed for each block. Orientograms were introduced by Kumar and Hebert [61] to help discriminate between man-made structures such as buildings and natural structures such as trees. Each orientogram is a histogram of the edge orientations within 1x1, 2x2 or 4x4 blocks. Fourteen different statistics are computed on the three orientograms to which a block belongs. Nine of the features are internal to each orientogram, including: the first and third heaved central-shift moments at each of the three scales, the dominant orientation at each scale, and the sine of the absolute difference between the two dominant orientations at each scale. To model the relationship between features in adjacent blocks, the cosines of the absolute differences of the dominant orientations between scales 1 and 2, and between scales 2 and 3, are computed. Examples of orientograms are given in Figure 9.1. Intuitively, the orientograms for blocks on buildings are unimodal or bimodal, and the difference between their dominant orientations is approximately $\frac{\pi}{2}$.

Unsupervised segmentation is performed using the mean shift algorithm with the 19 features described above for each pixel block. Examples of such segmentations can be seen in Figure 9.2(b).

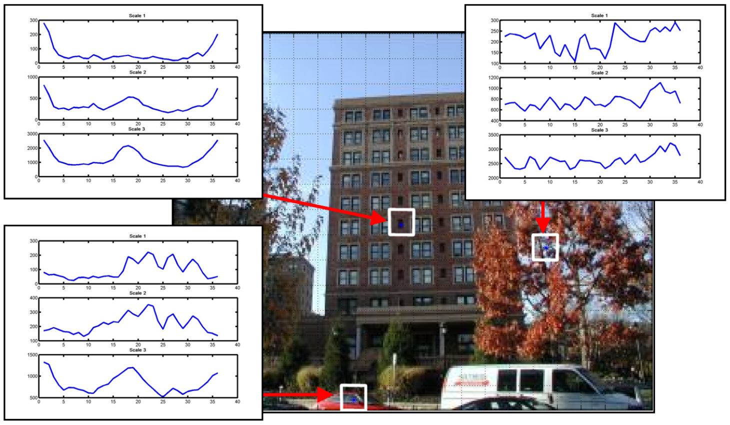


FIGURE 9.1. Examples of orientograms at each of three scales for blocks centered on a pixel belonging to a building, a pixel belonging to vegetation, and a pixel belonging to a car. For each histogram, the x-axis represents (quantized) edge orientation and the y-axis the total edge strength at that orientation. The differences between the orientograms of structured and unstructured objects are evident.

A segmentation-generated region is classified using the information of each encapsulated block. Specifically, if S represents the presence of building structure, and F_i represents the 14-dimensional feature vector derived from a block's orientograms, then the likelihood that a *block* belongs to a building is defined to be:

$$l_i = \frac{P(S|F_i)}{1 - P(S|F_i)} \tag{9.1}$$

Examples of block-level classification are given in Figure 9.2(c). Combining the likelihood from each block, the confidence that a region r belongs to a building is:

$$c_r = \frac{1}{n} \sum_{i \in r} w_i \frac{1}{1 + \exp^{-\alpha(l_i - 1)}} \tag{9.2}$$

The result of this process is a (block-level) image labeling such as in Figure 9.2(d) and (e). We can see that by using image segmentation, the noise in the classification of individual blocks is removed.

1.2. Extracting planar information from range data

Range data, often generated by laser range finders and shown as a cloud of 3D points such as in Figure 9.3(a), is a rich source of structural data. Since our task is to create object masks of buildings, we utilize proposals for planar structures in the 3D data, such as those in Figure 9.3(b). Information regarding the discovery of

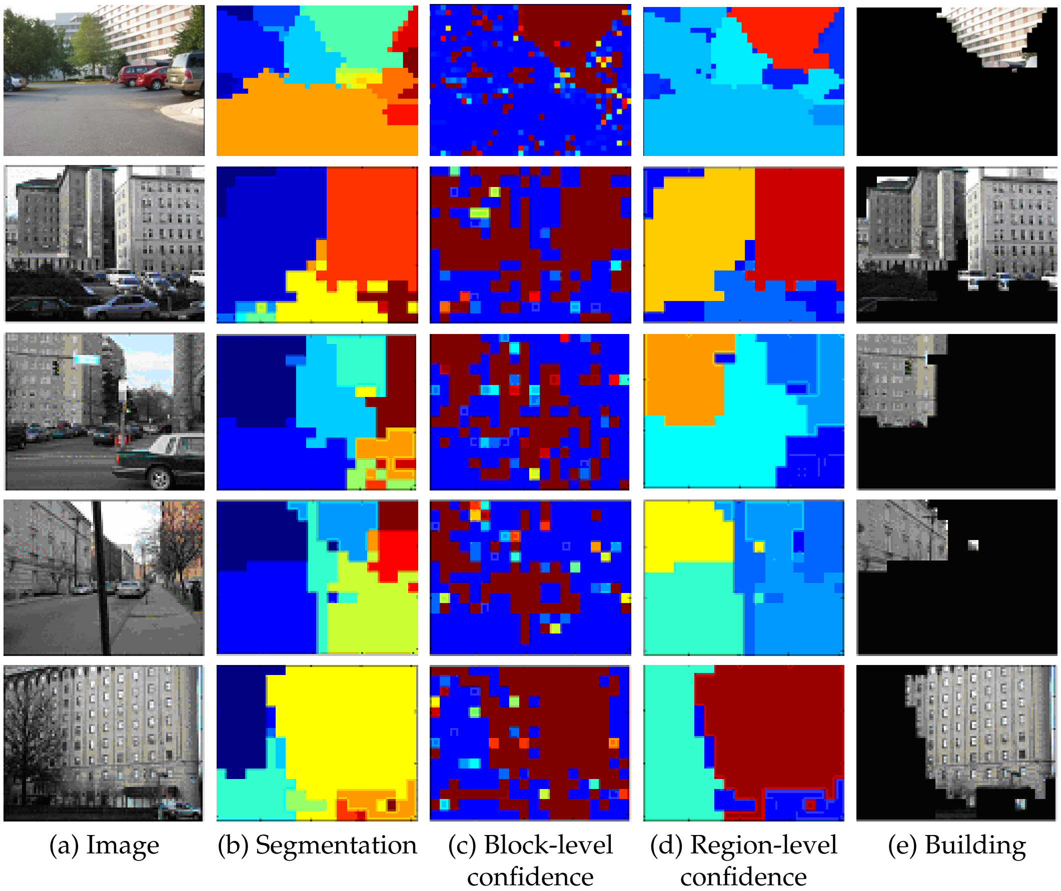
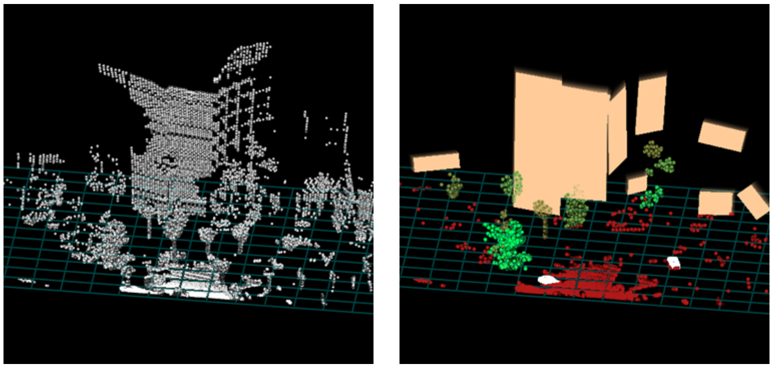


FIGURE 9.2. Building classification resulting from using image information. For each image, column (b) shows the unsupervised segmentation (colors are random), column (c) shows the classification of individual blocks (jet colormap), column (d) shows the classification of each region (jet colormap), and column (e) shows the final building localization.

planes in 3D point clouds can be found in [89] and [114]. All manipulation of 3D data in this work was performed by R. Unnikrishnan.

1.3. Combining information from multiple sensors

The image and 3D information can be registered into a colored 3D point cloud by calibrating the camera and LADAR to each other. The points in this colored cloud can now be classified using both image and range information. Let k_i be proportional to the likelihood that block i belongs to a plane, as recovered from the range data. As in Equation 1.1, let l_i be proportional to the likelihood that block i belongs to a building, as recovered from the image data. We combine these



(a) Point cloud

(b) Label structures

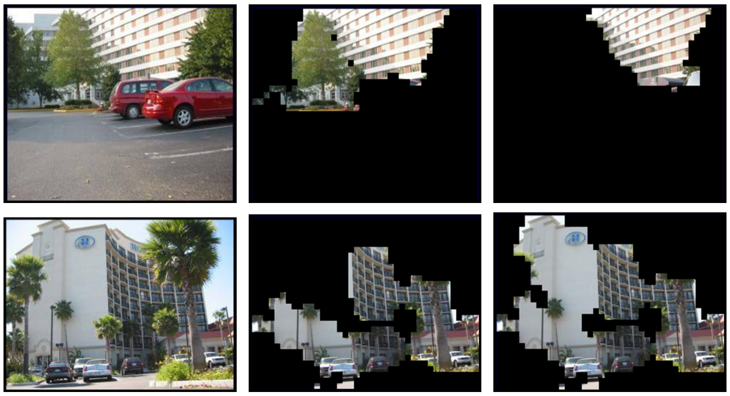
FIGURE 9.3. A 3D point cloud (a) and the derived structure (b). Peach indicates planar structures, green indicates vegetation, and red the ground plane. 3D structure courtesy of R. Unnikrishnan.

sources of information to compute the final confidence c_r that region r comes from a building as:

$$c_r = \frac{1}{n} \sum_{i \in r} k_i w_i \frac{1}{1 + \exp^{-\alpha(l_i - 1)}} \tag{9.3}$$

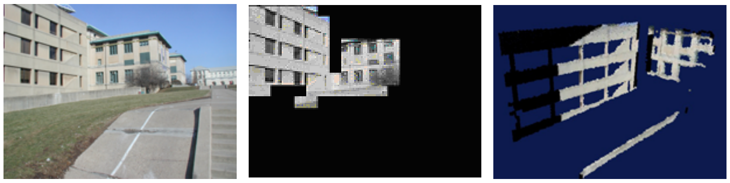
This method of fusing the image and range information corrects misclassifications from either individual sensor by: 1) increasing the weight of any block which is classified as belonging to a plane in the 3D data, and 2) allowing regions which are determined unlikely to belong to a building from the image data to remain classified as background.

Figure 9.4 shows the results of using such a fusion scheme. The image-based classifications in column (b) each contain errors, the top result incorrectly includes the tree, and the bottom result incorrectly excludes the texture-less wall. The results from combined classification in column (c) are more accurate, removing the tree and including the wall. These examples show two interesting scenarios where sensor information in addition to image data is particularly useful. The erroneous inclusion of the tree was likely due to the building structure which can be seen from behind the tree. In the range data, however, the tree appears as an unstructured cloud regardless of the limited structure behind it. In the second example, the wall of the hotel was excluded due to its lack of texture, however this does not affect the extraction of planes from range data.



(a) Image (b) Image-based classification (c) Combined classification

FIGURE 9.4. Building classification resulting from combining image and range data. Column (b) shows the classification results using the image data alone, which produces partially erroneous results. Column (c) shows the classification resulting from using both image and range data, which correctly removes the tree in the first image, and includes the textureless wall in the second image.



(a) Image (b) Image-based classification (c) Combined classification

FIGURE 9.5. Building classification resulting from combining image and range data. Image (b) shows the classification results using the image data alone which produces small errors near the leafless tree and along the bottom of the walls. Image (c) shows an oblique view of the classification using combined sensor information. The colored points correspond to pixels in the image, while the black points were only visible in the range data.

Figure 9.5 shows another example of combined classification. Again, image (b) shows the resulting classification using image information alone. Image (c) shows an oblique view of the 3D structure of the building classified using both sensors. Colored points correspond to pixels in the image, while black points were outside the camera's field of view. We can see that fusing the sources of information has removed the small tree against the far wall, while cleaning up the classification results along the bottom of the wall.

1.4. Conclusions

This section has shown a preliminary approach to combining information from cameras and laser range finders to produce more accurate masks of buildings in outdoor urban scenes. As future work, it would be interesting to extend this approach to use the more developed image segmentation-based approach for proposing object masks discussed in this thesis. In addition, other sensors (such as video) and other methods for combining the sensor modalities could be examined. While designing a method for fusing sensor data is difficult, the benefits of multiple independent and complementary sources of information are clear.

2. Improving the scalability of object recognition using large image sets

The field of object recognition has become preoccupied with the task of generalization. It is no longer sufficient to identify and locate a specific object, such as my green office chair, now we must find a class of objects, such as all possible chairs. As a consequence, we are encountering issues related to system scalability. These issues have crept into all three parts of the object recognition process: gathering image sets for training, building object class models, and querying the models regarding the presence and location of an object in a new image. Image sets used for training a system must contain examples of an object class that span all possible variations in pose, lighting, appearance, etc. This implies that the size of these data sets must be very large. The data sets used throughout this thesis, despite often being quite large (up to 500 positive and 500 negative images for one object class), have still proven inadequate. To accommodate the variation in object classes, object classifiers have become increasingly less specific, resulting in object models that are not well-separated from the background model. Finally, at query time, vague models lead to time-consuming searches for possible configurations. All of these problems are exacerbated as we increase the number of object classes.

The key to battling these issues of scalability may in fact lie in increasing the size of training image data sets even further. Recently, the number and availability of digital images has exploded, and the ability to access them in an organized manner has improved. In addition, images are increasingly being tagged with weak labels which reflect the image content (but not the location of objects within the images.) One particularly interesting image set is the ESP game data set, composed of a large number of internet images labeled by players of the online ESP game [123]. By having pairs of players compete together to name the same objects in an image (in a set amount of time), the ESP game ensures that both the most salient objects are labeled and that labels are correct. In this section, we discuss how using very large, weakly-labeled image sets can help us deal with scalability issues in object recognition algorithms. Using the technique described here could potentially improve the performance of weakly supervised object recognition and segmentation frameworks such as the one we discuss in this thesis.

How can an enormous, weakly labeled data set such as the ESP data set ameliorate scalability issues? By allowing us to stop generalizing about objects and scenes. Given a query image, consider retrieving images from an extremely large data base that have the same high-level similarity, such as the same 'gist' [84]. By using a low-dimensional image representation, we can make the search relatively efficient. If the data set is sufficiently large, this search should return a number of images with the same structure as the query. From the labels associated with the retrieved images, we can extract the most likely labels for the query image. This is a similar idea to [109], however we propose to use all of the labels associated with objects in an image, instead of only one label for the entire image. Using holistic scene characteristics for retrieval takes advantage of the image context in the query image to retrieve groups of object labels which co-occur under the same conditions. This is opposed to the piecewise approach of first classifying the objects in an image and then deciding whether those objects make sense together, which requires generalized object models. Of course, some of the retrieved images will have a similar gist but represent different actual scenes, so this object classification step simply proposes possible consistent object sets.

We also propose that the images and labels retrieved during classification can be used to create relevant, specific training data sets for object segmentation methods such as our own. By creating training image sets composed of images with similar context to the query, we encourage object exemplars which share characteristics such as lighting conditions, scale, and pose with the query image. In addition, we can learn which labels should co-occur. This implies a sort of on-line training, where new object models are created for a specific query image. The novelty lies in the use of very large data sets to avoid generalization; we are trying to segment specific objects as they look when present in a specific scene composition, and only together with other objects that can appear in such a scene.

2.1. Image classification

The first step in our image mining approach is image classification. For our purpose, classification refers to the retrieval of one or more sets of labels from the training data set which identify many or all of the objects in the query image. As observed in [109], it is highly likely that an image similar to the query will exist in a large enough image set, and the labels for the retrieved image will also apply to the query image. In contrast to [109], however, we are interested in extracting multiple (ideally all) object labels, not just one scene label.

An obvious approach, although not the most efficient, is to search for the (approximate) nearest neighbours to the query image. With a large enough data set and minor image blurring and illumination invariance, this may in fact retrieve a very similar image to the query and allow us to copy its labels. Although this approach may be overly simplistic and expensive, the sheer size of the data set makes success an interesting possibility, and experiments should be conducted to compute the probability of correct object identification.

A more efficient and general approach to retrieving similar images would be to match images by a lower-dimensional representation, such as their gist or by severely downsampled versions [109]. Work by Torralba et al. [84, 109] and Hays and Efros [50] has shown that the gist of an image can be successfully used to retrieve similar images from very large data sets. With precomputation of the gist for each training image, this lower-dimensional query would be much faster than a nearest-neighbor search. In addition, reducing the dimensionality would increase the generalization of the search, allowing multiple similar images to be returned. On the other hand, the low dimensionality would likely obscure smaller objects, which would then have to rely on being well-predicted by the image context. Of course, the same gist could match multiple scenes, for example a row of buildings could look very much like a row of books in a low-dimensional space. In this case, multiple sets of co-occurring objects would be proposed. For this approach, it would be interesting to compute how often the search returns only object labels which are present in the query image, versus additional object sets.

2.2. Creating weakly labeled training data sets for object segmentation

The images and labels retrieved during the classification task can also serve as weakly labeled training sets for object segmentation. By using this restricted training set of similar-context images, scalability issues can by avoided. By modeling which objects are likely to appear in a certain scene context, we can limit the search for objects in the query image. As opposed to [110] we propose to use the context information from all of the objects in an entire scene. Images with similar gist would share at least some of the same content, lighting, scale, pose, etc., all of which constrain the appearance of included objects. Due to the breadth of the image data set, training sets can be created on-line which are better-tuned to a query image. The cost of creating object models can be amortized by storing them for future use.

The union of the retrieved images' labels form the list of objects which might exist in the query image and hence for which we need models. We have mentioned that a single gist may correspond to multiple actual scenes (such as a bookshelf and a row of buildings), so the first step to creating training sets will be to separate the retrieved images into sets of mutually exclusive labels. The image can then be analysed separately for each label set and a maximum likelihood approach used to select between the sets. For clarity, the remainder of this discussion will assume that all of the retrieved images belong to the same scene type, such as wide-angle city scenes.

To create a training set for weakly supervised learning, the retrieved images need to be divided into those that contain an object (positive images) and those which do not (negative images). To partition the images and labels into positive and negative training sets for each object, we need to consider the relationships between the labels, and the interplay between labels and image context. Below, we divide the object labels into four categories.

The first type of object label is one that occurs in some but not all of the retrieved images. One possible example is cars in wide-angle city scenes, which are likely to be small and hence do not dominate the scene context. Weakly-trained classifiers will have difficulty separating small objects from their background unless the negative images are very similar to the background in the positive images. Thus, negative images must be chosen which share a large number of labels with the positive images. In addition, we can put a prior for small objects on the model. Finally, by learning models for the larger objects first, we can narrow down the possible locations of the smaller objects.

The second type of object label occurs in all of the retrieved images. These objects define the context of an image, such as the sky or a row of buildings. These objects pose two difficulties: finding a negative set, and separating them from each other since they always co-occur. To form the negative set and separate the objects, additional images will have to be retrieved from the data set which, ideally, contain all but one of the object labels. For example, for city scenes we can retrieve images that have building and road labels but not a sky label. Matching the context of partial images can increase the appropriateness of the additional data with respect to illumination, scale, etc.

The third object type is an object part, such as a wheel or a hand. For these objects we would need to rely on occlusion and possibly part relationship information from an outside source such as Wordnet [35].

The fourth and final object types are unlabeled objects. In the ESP game, for example, the goal is to name as many of the same objects as your partner in limited time, so insignificant or non-salient objects such as a wall or 'street furniture' may remain unlabeled. At this time, the most reasonable solution to this problem is a general 'clutter' class containing the unidentified objects in the image. Unlike past approaches in which 'clutter' has been a vague class whose model often obscures more specific object models, the clutter in a set of our retrieved images should have a much narrower and denser distribution due to the shared image characteristics.

2.3. Assumptions and limitations

- The image data set is assumed to be representative of any queried image by including sufficiently many images with similar gist which also represent that same scene.
- The labels for each image are assumed to be reasonably exhaustive except for clutter. This could be tested in the classification evaluation.
- Labels are assumed to be nouns. For future work, other parts of speech could provide additional context.
- Misspelled words, synonyms and 'type of' relationships are assumed to be mapped to one correct label. Some ways to accomplish this are user collaboration such as the ESP game, using Wordnet, or using Google's 'Did you mean?' functionality.

2.4. Related work and additional data sets

Limiting the scope of training data to model objects with multi-modal distributions was explored by Kumar et al. [63]. However, their generative model required fully supervised training data so they were only able to classify a few objects at once. In addition, they did not consider co-occurrence of object labels as a cue.

In [29], Duygulu et al. use weak training on a 4500-image subset of the Corel database to predict region labels. They show that abstracting together synonyms or words in 'type-of' relationships can improve recognition when the individual words cannot be discerned. They do not, however, consider co-occurrence or context information, so their method suffers from the scalability issues of other object recognition systems.

Recently, Schroff et al. [100] have proposed a method for downloading image databases from the web. As opposed to the gist-based method proposed here, their method relies on text-based searches such as Google and Google Image Search to download the images. This leads to a second task of filtering out images of incorrect objects (since only 30% of images returned are of the correct object) and inappropriate images such as sketches. They also only search for one object label at a time, thereby ignoring context.

There also exist additional data sets which may prove useful for this task. Some of these sets include:

- The most general and expensive approach is to use the entire set of internet images, using labels from Google image search for the object labels. Using the gist to filter retrieved images should increase the likelihood of correct text labels. This same approach could be applied to images and tags from web sites of photograph collections, such as flickr [42].
- The LabelMe project [95] also contains a large number of human-labeled images. Given that it requires the more time-consuming task of human object tracing, it is not as large as the ESP data set.
- The set of 32x32 pixel images collected by Torralba et al. [109] could also potentially be used, although full resolution images would need to be available for object segmentation and it would be necessary to experimentally confirm that such small images hold enough information for effective matching.
- The Corel data set contains a partial weak labeling of its images.

2.5. Conclusions

In this section, we have proposed a method for generating training sets on-line for our object recognition and segmentation framework. By taking advantage of large data sets and the power of context, scalability issues in creating object class models should be improved.

CHAPTER 10

CONCLUSIONS

Applications of object recognition in the foreseeable future will be extremely demanding. Precise robot manipulation of an object, image editing for artistic purposes, along with a slew of other applications will require exact pixel-accurate masks of an object's location. Unsupervised image segmentation is one possible data-driven mechanism for proposing the pixel support of an object, object part, or image feature. In this thesis, we have examined the issues related to using image segmentation in this manner.

We began by asking whether image segmentation could be used as a black-box pre-processing step, to propose masks of entire objects or object parts. To answer that question, we developed a novel framework for assessing and comparing segmentation algorithms. We identified three necessary criteria a black-box segmentation algorithm must satisfy: correctness, consistent performance on a given image for a wide range of parameters, and consistent performance across an image set for the same parameter setting. In addition, we outlined experiments and proposed a novel measure of segmentation correctness, the Normalized Probabilistic Rand (NPR) index, for evaluating these criteria. Multiple popular segmentation algorithms were evaluated within this framework, however none of the algorithms satisfied all of the criteria.

Despite not being an appropriate black-box pre-processing algorithm, we have shown that with careful application, image segmentation can be an effective mechanism for reducing the number of possible pixel sets that form the spatial support for an object. In fact, the lessons learned from the segmentation experiments motivated our approach to the problem.

The segmentation experiments showed that unsupervised segmentation is extremely unlikely to generate a segmentation of an entire image that corresponds to human partitioning of that image. This implies that segmentation-generated image regions do not correspond to parts or objects in an intuitive manner, often over-segmenting objects into smaller pieces. As a consequence, in order to be able to discriminate between an object and the background, the description of a region must include information from both inside a region and around a region, or in neighboring regions. In Chapter 5, regions were described in two manners. The repetitive texture inside a region was described using the established histogram of textons descriptor. In addition, we proposed a novel descriptor, the Region-based Context Feature (RCF), to describe the discriminative but non-repetitive structures both within and *around* a region in a principled manner. Numerous experiments were performed to compare different instantiations of each descriptor, as well as methods for combining them. The combination of these descriptors in a simple Naïve Bayes fashion facilitated object recognition and state-of-the-art pixel accuracy in the resulting object masks.

Another conclusion which arose from our evaluation of segmentation algorithms was that segmentation is sensitive to both the algorithm and parameters used. All of the segmentation algorithms tested produced segmentations of vastly different qualities depending on the parameters used. In addition, the same parameter choice did not produce segmentations of equal quality on all of the images. Experiments in Chapter 7 showed that in fact the overall pixel accuracy of our system on an entire image data set could decrease by 10% given a poor choice of segmentation algorithm and parameters. To increase the robustness of our system to poor segmentations, Chapter 7 proposed intergrating the information from multiple image segmentations to produce a final object mask. By labeling pixels which belonged to the same region in every segmentation (intersections of regions) as a unit, object masks could be created that captured the boundaries in any of the segmentations. The key, however, was that features and classification scores were computed from the regions in the individual segmentations, not the intersections of regions. These original regions had larger spatial support and thus produced more reliable and descriptive features. Experimental evidence confirmed that by using the information from all of the segmentations in concert, the pixel-level accuracy produced by our algorithm was robust to the existence of poor image segmentations.

Given that segmentation regions rarely correspond to single objects, we returned to the question of incorporating spatial information into our model in Chapter 8. Both describing regions using RCFs and generating multiple segmentations of different granularities allowed us to capture spatial and contextual information implicitly. In Chapter 8 we looked at capturing this information explicitly by learning the relationships between adjacent pairs of regions and enforcing global consistency through a random field formulation. Explicitly capturing spatial information showed significant benefit in weakly supervised training scenarios which did not use multiple segmentations. However, the use of full supervision during training, plus multiple segmentations, negated the benefit of additional spatial modeling. These experiments showed that each component of a system should be carefully evaluated as their efficacy can vary.

Throughout this thesis, we have carefully considered the nature of image segmentation and proposed methods for harnessing that nature. The result is a study of the power of image segmentation to provide accurate proposals for object and feature pixel supports, and how those proposals can facilitate state-of-the-art object recognition and object segmentation performance.

REFERENCES

[1] S. Adalsteinsson and H. Wardum. Frequency of color genes in Faeroe Islands sheep. *Journal of Heredity*, 69:259–262, 1978.

[2] S. Agarwal, A. Awan, and D. Roth. Learning to detect objects in images via a sparse, part-based representation. *PAMI*, 2004.

[3] S. Agarwal and D. Roth. Learning a sparse representation for object detection. In *ECCV*, 2002.

[4] N. Ahuja. A transform for multiscale image segmentation by integrated edge and region detection. *PAMI*, 1996.

[5] N. Ahuja and S. Todorovic. Learning the taxonomy and models of categories present in arbitrary images. In *ICCV*, 2007.

[6] A. Azran and Z. Ghahramani. Spectral methods for automatic multiscale data clustering. In *CVPR*, 2006.

[7] K. Barnard, P. Duygulu, R. Guru, P. Gabbur, and D. Forsyth. The effects of segmentation and feature choice in a translation model of object recognition. In *CVPR*, 2003.

[8] S. Belongie and J. Malik. Matching with shape contexts. In *Workshop on Content-Based Access of Image and Video Libraries*, June 2000.

[9] S. Belongie and J. Malik. Shape matching and object recognition using shape contexts. *PAMI*, 2002.

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

[11] E. Borenstein and J. Malik. Shape guided object segmentation. In *CVPR*, 2006.

[12] E. Borenstein and S. Ullman. Class-specific, top-down segmentation. In *ECCV*, 2002.

[13] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *PAMI*, 23:1222–1239, 2001.

[14] M. Burl, M. Weber, and P. Perona. A probabilistic approach to object recognition using local photometry and global geometry. In *ECCV*, 1998.

[15] N. W. Campbell, B. T. Thomas, and T. Troscianko. A two-stage process for accurate image segmentation. In *Int. Conf. on Image Processing and its Applications*, 1997.

[16] L. Cao and L. Fei-Fei. Spatially coherent latent topic model for concurrent segmentation and classification of objects and scenes. In *ICCV*, 2007.

[17] P. Carbonetto, N. de Freitas, and K. Barnard. A statistical model for general contextual object recognition. In *ECCV*, 2004.

[18] C. Carson, S. Belongie, H. Greenspan, and J. Malik. Region-based image querying. In *Workshop on Content-Based Access of Image and Video Libraries*, June 1997.

[19] C. Carson, S. Belongie, H. Greenspan, and J. Malik. Blobworld: Image segmentation using expectation-maximization and its application to image querying. *PAMI*, August 2002.

[20] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[21] H. I. Christensen and P. J. Phillips, editors. *Empirical Evaluation methods in Computer Vision*. World Scientific Publishing Company, July 2002.

[22] C. M. Christoudias, B. Georgescu, and P. Meer. Synergism in low level vision. In *ICPR*, 2002. http://www.caip.rutgers.edu/riul/research/code.html.

[23] J. Cohen. A coefficient of agreement for nominal scales. *Educ. and Psychological Measurement*, pages 37–46, 1960.

[24] M. Collins, R. Schapire, and Y. Singer. Logistic regression, Adaboost and Bregman distances. *Machine Learning*, 2002.

[25] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *PAMI*, 2002.

[26] D. Crandall, P. Felzenszwalb, and D. Huttenlocher. Spatial priors for part-based recognition using statistical models. In *CVPR*, 2005.

[27] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, pages 1–38, 1977.

[28] G. Dorkó and C. Schmid. Selection of scale-invariant parts for object class recognition. In *ICCV*, volume 1, pages 634–640, 2003.

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

[30] F. J. Estrada and A. D. Jepson. Quantitative evaluation of a novel image segmentation algorithm. In *CVPR*, 2005.

[31] M. Everingham, L. V. Gool, C. Williams, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2006 (VOC2006), 2006.

[32] M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman. The PASCAL VOC2007. http://www.pascal-network.org/challenges/VOC, 2007.

[33] M. R. Everingham, H. Muller, and B. Thomas. Evaluating image segmentation algorithms using the pareto front. In *European Conference on Computer Vision (ECCV)*, volume 4, pages 34–48, 2002.

[34] M. R. Everingham, B. T. Thomas, and T. Troscianko. Head-mounted mobility aid for low vision using scene classification techniques. *Int'l Journal of Virtual Reality*, 3, 1999.

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

[36] P. Felzenszwalb and D. Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 59(2):167–181, 2004.

[37] P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. *IJCV*, 61, 2005.

[38] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, volume 2, pages 264–271, June 2003.

[39] R. Fergus, P. Perona, and A. Zisserman. Weakly supervised scale-invariant learning of models for visual recognition. *IJCV*, 71(3):273–303, March 2007.

[40] V. Ferrari, T. Tuytelaars, and L. V. Gool. Simultaneous object recognition and segmentation by image exploration. In *ECCV*, May 2004.

[41] M. Fischler and R. Elschlager. The representation and matching of pictorial structures. *Computers, IEEE Transactions on*, C-22(1):67–92, 1973.

[42] flickr. www.flickr.com.

[43] C. Fowlkes, D. Martin, and J. Malik. The berkeley segmentation engine.

[44] C. Fowlkes, D. Martin, and J. Malik. Learning affinity functions for image segmentation: Combining patch-based and gradient-based approaches. In *CVPR*, 2003.

[45] E. B. Fowlkes and C. L. Mallows. A method for comparing two hierarchical clusterings. *Journal of the American Statistical Association*, 78(383):553–569, 1983.

[46] J. Freixenet, X. Munoz, D. Raba, J. Marti, and X. Cuff. Yet another survey on image segmentation: Region and boundary information integration. In *European Conference on Computer Vision (ECCV)*, pages 408–422, 2002.

[47] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 2000.

[48] F. Ge, S. Wang, and T. Liu. Image-segmentation evaluation from the perspective of salient object extraction. In *CVPR*, 2006.

[49] K. Grauman and T. Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *ICCV*, 2005.

[50] J. Hays and A. Efros. Scene completion using millions of photographs. In *ACM Transactions on Graphics (SIGGRAPH 2007)*, 2007.

[51] X. He, R. Zemel, and D. Ray. Learning and incorporating top-down cues in image segmentation. In *ECCV*, 2006.

[52] D. Hoiem, A. Efros, and M. Hebert. Recovering surface layout from an image. *IJCV*, 75, 2007.

[53] D. Hoiem, A. A. Efros, and M. Hebert. Geometric context from a single image. In *ICCV*, 2005.

[54] J. Huang, S. Kumar, M. Mitra, W. Zhu, and R. Zabih. Image indexing using color correlograms. *IJCV*, pages 245–268, 1999.

[55] Q. Huang and B. Dom. Quantitative methods of evaluating image segmentation. In *IEEE Intl. Conf. on Image Processing (ICIP)*, pages 53–56, 1995.

[56] L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, pages 193–218, 1985.

[57] I. Y. Kim and H. S. Yang. Efficient image understanding based on the markov random field model and error backpropagation network. *PAMI*, 1992.

[58] P. Kohli and P. Torr. Measuring uncertainty in graph cut solutions - efficiently computing min-marginal energies using dynamic graph cuts. In *ECCV*, 2006.

[59] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? In *ECCV*, 2002.

[60] M. Kumar, P. Torr, and A. Zisserman. Obj cut. In *CVPR*, 2005.

[61] S. Kumar and M. Hebert. Man-made structure detection in natural images using a causal multiscale random field. In *CVPR*, 2003.

[62] S. Kumar and M. Hebert. A hierarchical field framework for unified context-based classification. In *ICCV*, 2005.

[63] S. Kumar, A. C. Loui, and M. Hebert. An observation-constrained generative approach for probabilistic classification of image regions. *Image and Vision Computing*, 21:87–97, 2003.

[64] L. Ladicky, P. Kohli, and P. Torr. Oxford Brookes entry, PASCAL VOC2007 Segmentation Challenge. http://www.pascal-network.org/challenges/VOC, 2007.

[65] S. Lazebnik, C. Schmid, and J. Ponce. Semi-local affine parts for object recognition. In *BMVC*, 2004.

[66] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CPVR*, 2006.

[67] B. Leibe and B. Schiele. Interleaved object categorization and segmentation. In *BMVC*, 2003.

[68] A. Levin and Y. Weiss. Learning to combine bottom-up and top-down segmentation. In *ECCV*, 2006.

[69] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[70] W. P. J. Mackeown, P. Greenway, B. T. Thomas, and W. A. Wright. Labelling images with a neural network. In *ICANN*, 1993.

[71] J. Malik, S. Belongie, J. Shi, and T. Leung. Textons, contours and regions: Cue combination in image segmentation. In *ICCV*, 1999.

[72] T. Malisiewicz and A. A. Efros. Improving spatial support for objects via multiple segmentations. In *BMVC*, September 2007.

[73] M. Marszalek and C. Schmid. Spatial weighting for bag-of-features. In *CVPR*, 2006.

[74] M. Marszalek and C. Schmid. Accurate object localization with shape masks. In *CVPR*, 2007.

[75] D. Martin. *An Empirical Approach to Grouping and Segmentation*. PhD thesis, Univ. of California, Berkeley, 2002.

[76] D. Martin, C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color and texture cues. *PAMI*, 2003.

[77] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *ICCV*, 2001.

[78] M. Meila. Comparing clusterings by the variation of information. In *Conference on Learning Theory*, 2003.

[79] M. Meilă. Comparing clusterings: an axiomatic view. In *Proceedings of the 22nd Intl. Conf. on Machine Learning (ICML)*, pages 577–584, 2005.

[80] J. W. Modestino and J. Zhang. A markov random field model-based approach to image interpretation. *PAMI*, 14, 1992.

[81] G. Mori, S. Belongie, and J. Malik. Shape contexts enable efficient retrieval of similar shapes. In *CVPR*, 2001.

[82] G. Mori, X. Ren, A. A. Efros, and J. Malik. Recovering human body configurations: Combining segmentation and recognition. In *CVPR*, 2004. http://www.cs.sfu.ca/ mori/research/superpixels.

[83] K. Murphy, A. Torralba, and W. Freeman. Using the forest to see the trees:a graphical model relating features, objects and scenes. In *NIPS*, 2003.

[84] A. Oliva and A. Torralba. Building the gist of a scene: the role of global image features in recognition. *Visual Perception, Progress in Brain Research*, 155, 2006.

[85] A. Opelt and A. Pinz. Object localization with boosting and weak supervision for generic object recognition. In *SCIA*, 2005.

[86] C. Pantofaru, G. Dorkó, C. Schmid, and M. Hebert. Combining regions and patches for object class localization. In *Beyond Patches Workshop, CVPR*, 2006.

[87] C. Pantofaru and M. Hebert. A comparison of image segmentation algorithms. Technical Report CMU-RI-TR-05-40, Robotics Institute, Carnegie Mellon University, September 2005.

[88] C. Pantofaru and M. Hebert. A framework for learning to recognize and segment object classes using weakly supervised training data. In *BMVC*, September 2007.

[89] C. Pantofaru, R. Unnikrishnan, and M. Hebert. Toward generating labeled maps from color and range data for robot navigation. In *IROS*, 2003.

[90] A. Rabinovich, T. Lange, J. Buhmann, and S. Belongie. Model order selection and cue combination for image segmentation. In *CVPR*, 2006.

[91] D. Ramanan. Using segmentation to verify object hypotheses. In *CVPR*, 2007.

[92] W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

[93] X. Ren and J. Malik. Learning a classification model for segmentation. In *ICCV*, 2003.

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

[95] B. Russell, A. Torralba, K. Murphy, and W. T. Freeman. Labelme: a database and web-based tool for image annotation. *IJCV*, 2007.

[96] S. Savarese, J. Winn, and A. Criminisi. Discriminative object class models of appearance and shape by correlatons. In *CVPR*, 2006.

[97] A. Saxena, M. Sun, and A. Ng. Learning 3-d scene structure from a single image. In *ICCV workshop on 3D Representation for Recognition (3dRR-07)*, 2007.

[98] C. Schmid. Constructing models for content-based image retrieval. In *CVPR*, 2001.

[99] F. Schroff, A. Criminisi, and A. Zisserman. Single-histogram class models for image segmentation. In *ICVGIP*, 2006.

[100] F. Schroff, A. Criminisi, and A. Zisserman. Harvesting image databases from the web. In *ICCV*, 2007.

[101] C. W. Shaffrey, I. H. Jermyn, and N. G. Kingsbury. Psychovisual evaluation of image segmentation algorithms. In *Advanced Concepts for Intelligent Vision Systems*, 2002.

[102] E. Sharon, A. Brandt, and R. Basri. Fast multiscale image segmentation. In *CVPR*, 2000.

[103] E. Sharon, A. Brandt, and R. Basri. Segmentation and bound detection using multiscale intensity measurements. In *CVPR*, 2001.

[104] J. Shi and J. Malik. Normalized cuts and image segmentation. *PAMI*, 2000.

[105] J. Shotton, J. Winn, C. Rother, and A. Criminisi. The MSRC 21-class object recognition database, 2006.

[106] J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *ECCV*, 2006.

[107] J. Sivic, B. Russell, A. Efros, A. Zisserman, and W. Freeman. Discovering objects and their location in images. In *ICCV*, October 2005.

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

[109] A. Torralba, R. Fergus, and W. Freeman. Tiny images. Technical report, MIT, 2007.

[110] A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: a large dataset for non-parametric object and scene recognition. http://people.csail.mit.edu/torralba/tmp/tiny.pdf, 2007.

[111] Z. Tu, Z. Chen, A. L. Yuille, and S.-C. Zhu. Image parsing: Unifying segmentation, detection, and recognition. *IJCV*, 2005.

[112] Z. Tu and S.-C. Zhu. Image segmentation by data-driven markov chain monte carlo. *PAMI*, 24:657–673, 2002.

[113] T. Tuytelaars and C. Schmid. Vector quantizing feature space with a regular lattice. In *ICCV*, 2007.

[114] R. Unnikrishnan and M. Hebert. Robust extraction of multiple structures from non-uniformly sampled data. In *IROS*, 2003.

[115] R. Unnikrishnan and M. Hebert. Measures of similarity. In *IEEE Workshop on Applications of Computer Vision (WACV)*, pages 394–400, January 2005.

[116] R. Unnikrishnan, C. Pantofaru, and M. Hebert. A measure for objective evaluation of image segmentation algorithms. In *CVPR workshop on Empirical Evaluation Methods in Computer Vision*, 2005.

[117] R. Unnikrishnan, C. Pantofaru, and M. Hebert. Toward objective evaluation of image segmentation algorithms. *PAMI*, 29, 2007.

[118] J. van de Weijer and C. Schmid. Coloring local feature extraction. In *ECCV*, 2006.

[119] J. Verbeek and B. Triggs. Region classification with markov field aspect models. In *CVPR*, 2007.

[120] V. Viitaniemi. Helsinki University of Technology, PASCAL VOC2007 Challenge. http://www.pascal-network.org/challenges/VOC, 2007.

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

[122] F. Vivarelli and C. Williams. Using bayesian neural networks to classify segmented images. Technical report, Aston University, 1997.

[123] L. von Ahn and L. Dabbish. Labeling images with a computer game. In *Proc. ACM Conf. Hum. Factors Comp. Syst. (CHI)*, 2004.

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

[125] D. L. Wallace. Comments on “A method for comparing two hierarchical clusterings”. *Journal of the American Statistical Association*, 78(383):569–576, 1983.

[126] G. Wang, Y. Zhang, and L. Fei-Fei. Using dependent regions for object categorization in a generative framework. In *CVPR*, 2006.

[127] M. Weber, W. Einhuser, M. Welling, and P. Perona. Viewpoint-invariant learning and detection of human heads. In *Automatic Face and Gesture Recognition*, 2000.

[128] J. Winn and N. Jojic. Locus: Learning object classes with unsupervised segmentation. In *ICCV*, 2005.

[129] J. Winn and J. Shotton. The layout consistent random field for recognizing and segmenting partially occluded objects. In *CVPR*, 2006.

[130] T.-F. Wu, C.-J. Lin, and R. C. Weng. Probability estimates for multi-class classification by pairwise coupling. *Journal of Machine Learning Research*, 5:975–1005, 2004.

[131] S. Yu and J. Shi. Object-specific figure-ground segregation. In *CVPR*, 2003.