Soft Detection of Features for Unsupervised Object Recognition

R. White

29 May 2002

Acknowledgements

I would like to thank my advisor Pietro Perona, who guided me through the process and helped solidify my work. Additionally, Rob Fergus spent many hours helping me with the math and directing my work. Without his help, this wouldn't have been possible.

I would also like to thank my second reader Glen George, the instructor for the class, Kent Potter and my roommate Jeff Sullivan who proofread my work. Finally, my parents have provided the mental and economic support to help me through college.

Abstract

An extension to an unsupervised object learning algorithm is presented that uses the feature quality to improve the accuracy of object detection. In previous attempts, models relied on thresholds to sort spurious detections from actual detections. However, by modeling and basing decisions on the probability distribution of the detector response the accuracy improves. This probability is modeled by using a set of N bins for the foreground distribution above a new threshold and another set of N bins for the background distribution. Experiments justify the validity of soft detection by indicating useful values for both the number of bins, N, and the new threshold.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Recognition

The task in object recognition is to identify objects within images. In research settings this task takes many forms: identifying a single object (ex: a twenty dollar bill), identifying classes of objects, identifying objects in clutter, and classifying variation within classes of objects (ex: identifying people at an airport). Within the scope of this paper, the focus will be on the task of identifying a class of objects within cluttered scenes. The data sets will include instances of objects of different forms in natural (or unnatural) scenery. Examples of these images include faces or cars shown in Figure 1.1 and the corresponding background images.

1.2 Applications

Object recognition has many uses, including both commercial and military applications. These applications seek to replace human observers, improve accuracy and process data faster than previously possible. Some of the most common applications include Internet image searching, security, manufacturing and medical imaging. In one project, the Computer Vision Group at Caltech seeks to recognize other cars on the road in order to provide emergency automatic response to traffic conditions. There are currently several small companies attempting vision problems in several arenas, including robotics, eye scanning for identification purposes and medical analysis of urine image data.

These companies have narrowed the scope of the problem to work in highly specialized fields. While in many cases they have solved their problems successfully, these companies and applications represent only the tip of the iceberg of object recognition and computer vision.

1.3 Related Work

The task of object recognition seems like a simple one to a human observer, but just as there are many different ways to look at the task of object recognition there are many ways to look at computational solutions. In this paper the focus proposed by M. Weber [6] in the Computer Vision Group at Caltech. While this method doesn't have some of the high performance numbers and speed of learning that other

Figure 1.1: An example of two recognition tasks. To a human observer, these identification tasks are very simple. However, to a computer this task presents a challenge.

methods boast, it is a far more general method able to learn on many object classes with minimal human involvement.

1.4 Outline

The structure of this paper reflects the structure of the research. Because work started by analyzing the work of Markus Weber, Chapter 2 provides a short summary of Markus Weber's Ph.D. thesis [6]. The summary is short with all mathematical derivations removed. Chapter 3 moves on to the new work in soft detection and provides a mathematical and theoretical framework for soft detection. Chapter 4 covers a series of experiments that classify the success of soft detection and fine tune the algorithm. Finally, some brief conclusions are presented in Chapter 5.

Chapter 2

Background

2.1 Introduction

This section presents a background on one type of unsupervised object recognition developed by Weber, Burl, Leung and Perona [2], [3], [4], [6]. The fundamental goal of their work is to separate images based on predominant objects in the image. In order to simplify the classification task, they examine the existence decision: "Is an object of a given class in the image under inspection?" To answer this question, they implemented two stages: learning the object class followed by detection in a given image.

The learning stage builds a model of the information contained in the images in order to make a quick decision on new images. This model contains all of the relevant information and is the only data passed on to the detection stage.

2.2 Model Structure

The model seeks to characterize the set of training images which define the object class. The approach taken here is a hierarchical one. At the lowest level, detectors search for small parts of the object within an image. Building on the detectors, a larger model keeps track of the relative locations of the detectors.

The detectors are small patches of pixels ("features") that correspond to important elements of an object. For example, when learning faces, typical features are eyes, corners of mouths, ears, et cetera. Because these small sections are specific to very localized regions, it is very unlikely that a left eye detector will effectively detect a right eye. Furthermore, a typical model has a very small number of detectors because of the learning complexity associated with additional detectors. For most of the models studied in this paper, the number of detectors is typically two or three.

The next level in the hierarchy is the constellation model that keeps track of the geometry of the detectors statistically. The hierarchical model aims to keep track of the geometry in a robust way, and therefore records the mean and covariance of the detector locations in the training images. In order to keep models translationally invariant, relative locations are used instead of recording absolute positions.

Additionally, the model keeps track of the probability that a feature isn't found. This could occur for several reasons: the portion of the image that contains the feature is occluded, or the particular object of the class lacks the feature or the feature exists but the detector response is not strong enough to correctly identify it.

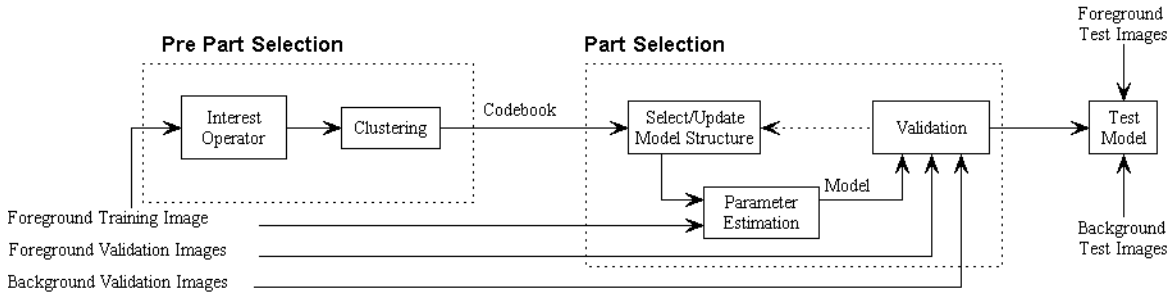

Figure 2.1: Flow graph of learning process. (image copied from Weber [6]).

2.3 Learning

The learning stage functions on two sets of input images: those labeled to contain the object, and those labeled to not contain the object. The data set is further broken into training, validation and test. Training data is used to pick possible detectors by identifying features that tend to occur in images containing examples of the object. Training images are also used for model learning. Validation is used for determining which learned models are better and test data is used to return performance numbers to the user.

The flow-graph shown in Figure 2.1 breaks the learning into two main stages: finding candidate parts (known as detectors) and then selecting these parts and learning the statistical model. The learning can be further broken down into the following components:

- **Interest Operator** Searches for interesting locations in an image. (Works with foreground training images.) These interesting points are extremely numerous($\sim 10,000$), and represent possible features to be used as detectors.
- **Clustering** Group interesting points obtained from previous stage based on similarity. This stage generates a small (~ 150) set of features that will later be used as candidate detectors. As sample set of features is shown in Figure 2.2 Many of these features represent legitimate features found mostly on the object, while others represent common features to all images (simple gradients are common).
- **Select Features** In the first iteration this step builds a model by selecting a small set of detectors. In later iterations, the detector that provided the worse performance is swapped out for a new detector. Learning concludes when the set of features doesn't change as the entire codebook is tested.
- **Parameter Estimation** The training set and the detectors selected in the previous stage are used to estimate the parameters in the model. These parameters include spatial and statistical information.
- **Validation** For the purposes of selecting the best model, the candidate models are tested on a new data set, the validation set. By looking at how well the model divides the data set into foreground and background, it is assigned a score which is used to decide which detector to replace.

Figure 2.2: A sample codebook used for many of the tests performed. Here each possible detector is made up of a eleven pixel by eleven pixel image created in the clustering of interest points. The numbers represent how many interest points created the detector.

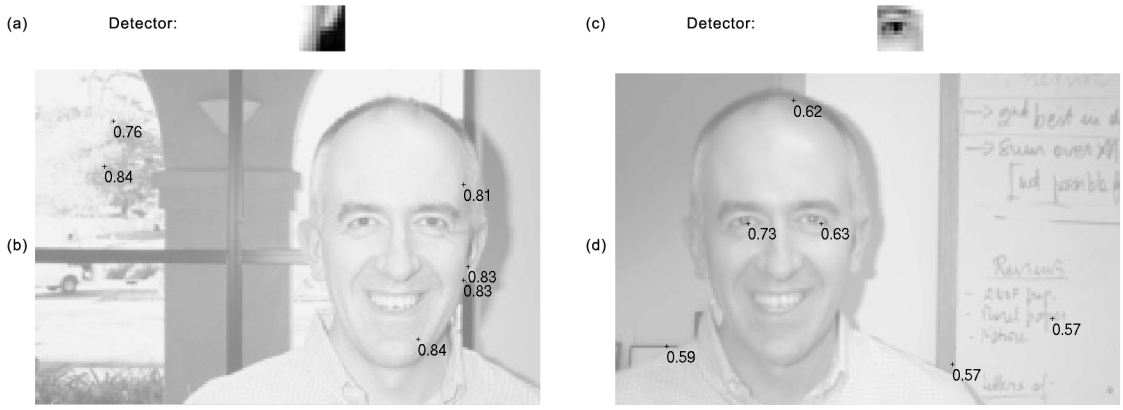

Figure 2.3: An example detector (a),(c) and a number of local maxima detections in a typical image (b),(d). The values at the labeled points represent the normalized responses which have values ranging from -1 to 1.

- **Testing** The testing stage provides no additional learning, but provides performance numbers similar to the Validation step that can be used to test the quality of a learned model.

2.4 Feature Selection and the Greedy Search

The iterated learning selects a small number of features ($\mathcal{F}+1$) from the codebook to use as detectors. For this small set of features, an even smaller set is selected ($\mathcal{F}$) for model learning. Every possible permutation of the ($\mathcal{F}+1$) features are combined into a model for learning. Of these models, the best one is selected, the least useful feature is removed and replaced by another feature from the codebook. This process repeats until no new feature is selected over the entire codebook.

While this process is not guaranteed to converge on the optimal model, it works well, and usually produces very good models. From a complexity standpoint, it reduces the number of models learned from $\mathcal{O}(|Codebook|^{\mathcal{F}})$ to $\mathcal{O}(|Codebook| * \mathcal{F})$.

2.5 Detectors

For the purposes of this paper, the most important portion of the object recognition method is the functionality of the detectors. As mentioned before, these detectors are composed of a map of pixels, an output of clustering. Each map of pixels, or detector, is compared against every possible contiguous patch of image in the data set, producing a normalized response score. By normalizing the two patches to be compared, these detectors are somewhat robust to local lighting conditions, and reveal more about the orientation and make-up of the pixels than the scene specifics. This score ranges from -1 (an inverted feature) through 0 (no correlation) to +1 (a perfect match). Because of the large number of detections in an image, this set of detector scores is reduced significantly by choosing only local maxima and by ruling out low scores. Specifically, the algorithm employs a hard threshold: a minimum usable response score.

Chapter 3

Soft Detection

3.1 Introduction

Using the detectors described in the previous section, the output can be utilized in several ways. As mentioned, hard thresholding removes all values below a certain score. Soft detection also removes all responses below a score, but instead preserves information about the score which can be used to classify. As the following section demonstrates, true detections will generally have higher scores than false detections. Therefore, it is possible to compute a probability that any given detection corresponds to a foreground detection. Intuitively, a higher detector score is more likely to correspond to a foreground detection.

3.2 Motivations

In order to assess how much information is contained in the detector response of feature matching, tests were performed comparing ground truth detections (obtained by a human operator hand clicking on features) with background responses. Figure 3.1 shows results of for a given eye feature taken from an existing codebook. The background response is significantly smaller than foreground response because it is symmetric around 0, hiding half of the results.

To ensure that the maximum response of the detector, after a human operator clicked on the eye, a ± 2 pixel region was searched for the maximum response. To obtain unbiased background results, locations were chosen at random in images that did not contain the object.

Because the foreground and background distributions have distinctly different means, significant useful information lies in the detector response. In previous work, this difference in responses was used to threshold the data. However, because the data above threshold still holds significant information, the detector response can be further used to make better predictions about the existence of an object.

3.3 Mathematical Notation

In this chapter, a mathematical derivation of the decision criteria is derived. Before this derivation, it is necessary to re-examine the learning process and the mathematical quantities that will be used.

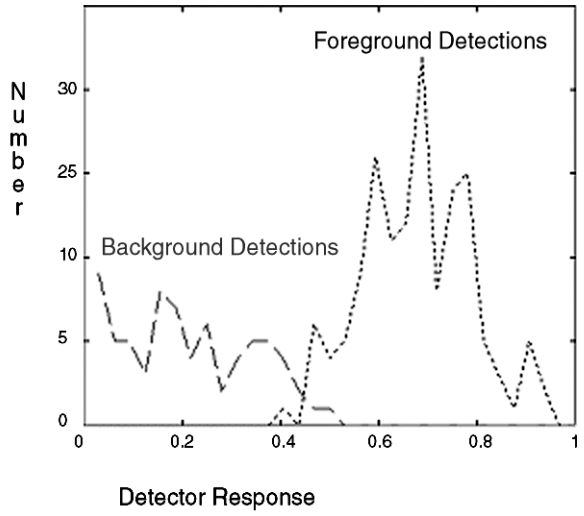

Figure 3.1: Histogram of results obtained by (a) hand-clicking on foreground images and (b) randomly selecting locations from background images. The x-axis is the detector response (which ranges from -1 to 1, while only 0 to 1 is shown). The y-axis is number of responses in a given detector response range.

Variable	**Size**	**Explanation**
$\mathcal{O}$	binary	$\mathcal{O}_1$ corresponds to an existing object, $\mathcal{O}_0$ to a lacking object
χ	matrix	Locations of candidate detections
$\mathbf{R}$	matrix	Detector responses at locations in χ
$\mathbf{h}$	vector	Hypothesis selects values out of χ
$\mathbf{h}_0$	vector	The null hypothesis: no detections within the image
$\mathcal{B}$	integer	number of detector bins used
$\mathcal{F}$	integer	number of feature detectors used
$\mathbf{N}$	vector	N_f is the total number of candidates of type $f \in \mathcal{F}$

Table 3.1: Summary of notation.

After filtering, a list of the detector responses is stored for each location within the image. This list is thresholded, and combined across features to produce χ. Rows in χ are organized by feature type and distinct members of each row are distinct detections of the feature. Each element in χ contains the x and y locations of each detection. Because the threshold is an important quantity in learning, the related matrix $\mathbf{R}$, which holds the detectors responses, will be used. Each index into $\mathbf{R}$ is the detector response of the corresponding (x,y) entries in χ. Now as far as the recognition is concerned, χ and $\mathbf{R}$ completely represent the set of input images after filtering.

When computing probabilities, it will become convenient to sum over an unobserved variable, the hypothesis $\mathbf{h}$ in the set of all hypotheses $\mathcal{H}$. Each hypothesis is a vector of indices into the matrix χ, and therefore represents a potential observation of a model.

A summary of the mathematical notation is included in Table 3.1 .

3.4 Math for Soft Detection

In the context of finding objects, the goal is to determine if an object is in the image given the data. This can be computed using the following likelihood ratio and Bayes' rule to obtain class-conditional densities:

$$D = \frac{p(\mathcal{O}_1|\chi,\mathbf{R})}{p(\mathcal{O}_0|\chi,\mathbf{R})} = \frac{p(\mathbf{R},\chi|\mathcal{O}_1)p(\mathcal{O}_1)}{p(\mathbf{R},\chi|\mathcal{O}_0)p(\mathcal{O}_0)} \tag{3.1}$$

By summing over all hypotheses the decision criteria becomes:

$$D = \frac{\sum_{\mathbf{h}\in\mathcal{H}} p(\mathbf{R},\chi,\mathbf{h}|\mathcal{O}_1)p(\mathcal{O}_1)}{p(\mathbf{R},\chi,\mathbf{h}_0|\mathcal{O}_0)p(\mathcal{O}_0)}$$

$$D = \frac{\sum_{\mathbf{h}\in\mathcal{H}} p(\mathbf{R}|\chi,\mathbf{h},\mathcal{O}_1)p(\chi,\mathbf{h}|\mathcal{O}_1)p(\mathcal{O}_1)}{p(\mathbf{R}|\chi,\mathbf{h}_0,\mathcal{O}_0)p(\chi,\mathbf{h}_0|\mathcal{O}_0)p(\mathcal{O}_0)} \tag{3.2}$$

The terms $p(\chi,\mathbf{h}|\mathcal{O}_1)p(\mathcal{O}_1)$ and $p(\chi,\mathbf{h}_0|\mathcal{O}_0)p(\mathcal{O}_0)$ are identical to the term used by Weber et al [6]. Therefore, a modification to his work can be made by incorporating an additional factor: the sum of the ratio of probabilities of the detector response. Because the features are assumed to be independent at the detector level, this probability can be broken into a product over the features:

$$p(\mathbf{R}|\mathbf{h},\chi,\mathcal{O}_1) = \prod_{f=1}^{\mathcal{F}} p(R_f|\mathbf{h},\chi,\mathcal{O}_1) \tag{3.3}$$

$$p(\mathbf{R}|\mathbf{h}_0,\chi,\mathcal{O}_0) = \prod_{f=1}^{\mathcal{F}} p(R_f|\mathbf{h}_0,\chi,\mathcal{O}_0) \tag{3.4}$$

Here each hypothesis $\mathbf{h}$ has $\mathcal{F}$ features, and each feature has N_f detections. In this context, $\mathbf{R}$ is a matrix with element R_{ij} as the detector response of feature i, detection j. For foreground detections:

$$p(R_f|\mathbf{h},\chi,\mathcal{O}_1) = p(R_{fh_f}|\mathbf{h},\chi,\mathcal{O}_1) \prod_{i=1,i\neq h_f}^{N_f} p(R_{fi}|\mathbf{h},\chi,\mathcal{O}_0) \tag{3.5}$$

In the background (denominator):

$$p(R_f|\mathbf{h}_0,\chi,\mathcal{O}_0) = \prod_{i=1}^{N_f} p(R_{fi}|\mathbf{h}_0,\chi,\mathcal{O}_0) \tag{3.6}$$

In other words, the probability of the detector response is independent for each feature, and assuming that one detection is foreground necessarily assumes that all other detections are background. In the case where the hypothesis chooses that the feature does not exist in the image, but that the object is present, then all of the detected features are in the background:

$$p(R_f|\mathbf{h}_0,\chi,\mathcal{O}_1) = \prod_{i=1}^{N_f} p(R_{fi}|\mathbf{h}_0,\chi,\mathcal{O}_0) \tag{3.7}$$

Therefore, in the case where the hypothesis selects a feature:

$$\frac{p(R_f|\mathbf{h},\chi,\mathcal{O}_1)}{p(R_f|\mathbf{h}_0,\chi,\mathcal{O}_0)} = \frac{p(R_{fh_f}|\mathbf{h},\chi,\mathcal{O}_1)\prod_{i=1,i\neq h_f}^{N_f} p(R_{fi}|\mathbf{h},\chi,\mathcal{O}_0)}{\prod_{i=1}^{N_f} p(R_{fi}|\mathbf{h}_0,\chi,\mathcal{O}_0)}$$

$$= \frac{p(R_{fh_f}|\mathbf{h},\chi,\mathcal{O}_1)}{p(R_{fh_f}|\mathbf{h},\chi,\mathcal{O}_0)}$$

When the hypothesis selects a missing feature:

$$\frac{p(R_f|\mathbf{h}_0,\chi,\mathcal{O}_1)}{p(R_f|\mathbf{h}_0,\chi,\mathcal{O}_0)} = \frac{\prod_{i=1}^{N_f} p(R_{fi}|\mathbf{h}_0,\chi,\mathcal{O}_0)}{\prod_{i=1}^{N_f} p(R_{fi}|\mathbf{h}_0,\chi,\mathcal{O}_0)} = 1 \tag{3.8}$$

Therefore, the final decision criteria is:

$$D = \sum_{\mathbf{h}\in\mathcal{H}} \frac{p(\chi,\mathbf{h}|\mathcal{O}_1)p(\mathcal{O}_1)}{p(\chi,\mathbf{h}_0|\mathcal{O}_0)p(\mathcal{O}_0)} \frac{p(R_{fh_f}|\mathbf{h},\chi,\mathcal{O}_1)}{p(R_{fh_f}|\mathbf{h},\chi,\mathcal{O}_0)} \tag{3.9}$$

3.5 Binning

In order to accurately integrate the probability into the previous section's results, the probability of an item existing in the foreground and background given a detector response score must be known. All of the information presented thus far suggests that the probabilities when graphed against detector score are gaussians. See Figure 3.1 for a reminder of why the distribution is probably gaussian.

However, two flaws exist in making this assumption. First, no theoretical reasoning backs the assumption that the form is gaussian. Second, while it seems reasonable to learn the gaussian shape in the case of the foreground responses where most of the data is observed, it is very difficult to estimate the mean and variance when the bulk of the distribution is not viewable (for the background case).

Instead, the observable region (ie the region above threshold) will be broken into a number of discrete regions (called bins). For each region, a single value will describe the detector response distribution (either foreground or background). The advantage to this method is that it can be easily learned (discussed in the next section), and it will approximate any distribution, not just gaussians. The downside is that bins innacurately group responses together and are subject to noise.

To maintain some sense of similarity between different detectors, the region above threshold is broken into a fixed number of equally spaced bins, $\mathcal{B}$. Logically this means that some detectors will have extremely narrow bins (when the threshold is high) while other detectors will have wide bins (when the threshold is low). The other important aspect to remember is that probabilities of detector responses are only learned for values above threshold. Therefore, it is quite possible that the bulk of the curve is below threshold, and therefore invisible in the plot of learned values. (This is especially true for background densities which appear to be zero-mean gaussians)

3.6 Learning

Like the previous solution proposed by Weber [6] , the method of *expectation maximization* (EM) will compute the probability distributions. This method is an iterative approach which should quickly converge on the optimal values while that many other parameters are being estimated. (For a good introduction to expectation maximization see [1])

In the context of EM, update algorithms for the probabilities $p(R_f|\mathbf{h}_0, \chi, \mathcal{O}_1)$ and $p(R_f|\mathbf{h}_0, \chi, \mathcal{O}_1)$ must be computed. Fortunately, the update for these is simple: in each iteration add the probability of object present to the probability of foreground detector response and 1 - probability of a object present to background detector response. At the end of each iteration the learned values are re-normalized to sum to unity. As the model values become more accurate, so do the object presence probabilities, and ideally the learning converges. This method is not guarenteed to converge on the global maximum, but experiments in the robustness section indicate that it tends to in this situation.

One problem with this method is empty bins. If for some reason in one iteration of EM, the bin remains empty, then that bin will never gain a value in any subsequent iterations. This occurs because the probabilities are evaluated using the previous iteration and then multiplied. The solution to this problem is to enforce a lower limit on each bin. This limit will keep any bin from being effectively removed during learning, but has the downside that sparse regions in the detector responses will have unusually high probabilities according to the minimum value set.

Chapter 4

Experiments

4.1 Overview

After deriving the mathematical expressions for the soft detection terms, many parameters remain. For example: At what point should the threshold be set? How many bins should be used?

This chapter will first investigate typical results to show that soft detection is indeed helpful. Later in the chapter the parameter values are studied in a series of experiments for two reasons:

1. To find optimal values for learning / recognition.
2. To gain a better insight into the underlying data and to develop ideas for future research on ways to remove these hard coded values.

Table 4.1 includes a list of the parameters used in the object recognition software. While this list is long, in this section the emphasis will be on a select few that seem most crucial to soft detection: thresholds and the number of bins.

constant	**description**	**typical value**
$\mathcal{F}$	number of detectors used in the model	3
$\mathcal{B}$	number of bins used in soft detection	8
AvrCand	average number of feature detections per image	8
part size	pixel resolution of the detectors	11x11
codebook size	number of possible detectors in codebook	150
bin minimum	minimum value of bin used in soft detection	0.01

Table 4.1: A list of relavent parameters.

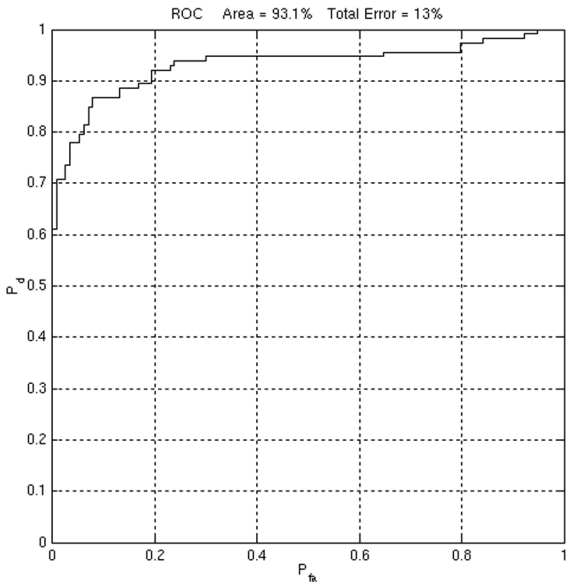

Figure 4.1: An example ROC curve for a decent classifier.

4.2 Testing Conditions and Metrics

In order to gauge the effects of changing these parameters, there are several different performance metrics. The first of these is the *receiver operating characteristics* (ROC), which characterizes the classification system by adjusting the classification threshold. The ROC is a curve, where the x-axis is percentage of foreground images classified as foreground, and the y-axis is the percentage of background images classified as foreground. Two points are therefore necessarily occur on every ROC: all images classified as foreground and all images classified as background. A good ROC curve will look like a Γ function and a bad ROC will be a diagonal line. (Theoretically, a really bad receiver would classify all background as foreground and all foreground as background, but then the classification could be inverted to produce a really good receiver.)

A good measure of performance is the area under this curve because is quatifies the performance under different circumstances. In many situations, the equal error rate classification is used (misclassify foreground and background equally), but in other cases a false negative has a higher cost than a false positive. Examples include medical diagnostics and screening for terrorists - better to double check potential problems.

To this end, a decision criteria for choosing new models during the learning stage is the area under the ROC. However, this area accentuates the positive aspects of the classifier, and the misclassification at equal error rate (as many foreground misclassifications as background misclassifications) is more indicative of actual performance in a given circumstance. In Figure 4.1 an example ROC curve is shown. Notice that the misclassification at equal error rate is 13% (correct classification is 87%) while the area under the ROC is 93% . Here using value of the area under the ROC makes the classifier sound better than it could perform in any actual condition.

Unfortunately, because of the convergence of the model learning, it is quite possible that it converges on a non-optimal model. Therefore, to accurately compare different test conditions it is necessary to run each

			Hard Detection		Soft Detection	
Data Set	# parts	part size	1 - A_{ROC}	error	1 - A_{ROC}	error
Faces	2	11 x 11	6.32	11.11	3.94	8.33
Faces	2	21 x 21	6.44	12.22	4.15	9.44
Cars	3	11 x 11	14.51	22.22	11.88	20.95
Cars	3	21 x 21	18.10	26.99	12.28	19.84

Table 4.2: Some preliminary results with soft detection. All unstated parameters are the "typical values" listed in the previous Table.

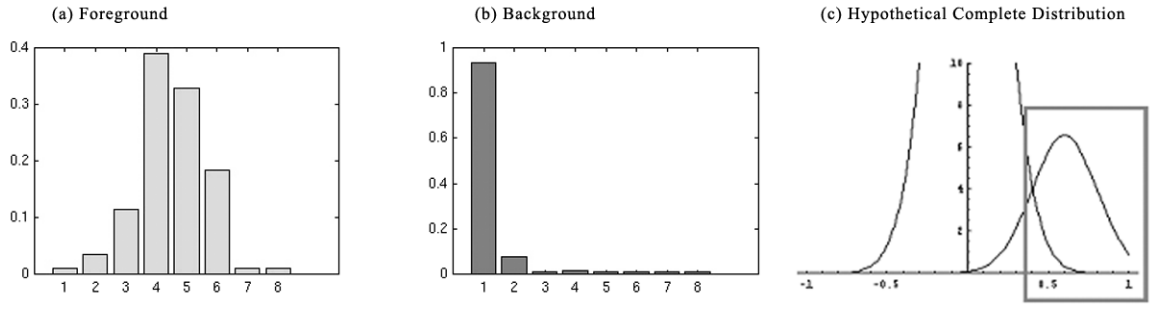

Figure 4.2: An example of a learned detector response distribution for a single feature in a given data set.

learning on the same data set many times. For the duration of the chapter, each test represents a minimum of twenty different test run and averaged. While twenty seems like a significant number, when looking at the standard error bars, it is appearant that more tests could reveal significantly more accurate results.

4.3 Example results

This section presents a "snapshot" of what happens with soft detection. This is a far from complete picture, but it should provide the reader some insight into the solution. Further characterization in a far more accurate form is provided in the later sections of this chapter.

Results here were obtained by modifying the MATLAB and C code created by Markus Weber [6] to include the soft detection terms. Tests were run on a two data sets: Faces and Cars.

Performance improvements for different data sets are summarized in Table 4.2 . The results of soft detection here are noticeable, though not extreme. The differences are significantly larger than the associated standard errors. As shown in later sections, these results will improve as the system learns the appropriate constants and adjust parameters accordingly.

Example learned detector responses are shown in Figure 4.2 . These curves reveal what is intuitively expected: the foreground curve is roughly gaussian with a mean well above threshold while the background curve is also roughly gaussian with a mean well below the threshold. To aid in understanding, a model of perfect gaussians (foreground and background) is drawn with a box to represent the region above threshold.

Hard Detection				
	1- Area under ROC		*Error Rate*	
AvrCand	*Test*	*Std. Err.*	*Test*	*Std. Err.*
3	6.16	1.37	8.61	1.46
5	9.79	1.77	11.4	1.85
6	8.79	1.66	11.7	1.75
7	11.3	1.70	13.6	1.80
8	12.7	2.29	14.4	2.46
12	28.9	5.52	33.3	4.47
Soft Detection				
	1 - Area under ROC		*Error Rate*	
AvrCand	*Test*	*Std. Err.*	*Test*	*Std. Err.*
3	4.14	0.88	7.92	1.31
5	3.47	0.88	6.81	1.46
6	2.47	0.86	5.97	1.43
7	3.01	0.67	7.50	1.46
8	2.48	0.54	5.97	0.96
12	5.93	1.46	13.3	2.28

Table 4.3: Data obtained by running multiple tests at different thresholds with hard detection (top) and with soft detection (bottom). Each number represents an average over six tests. Tests performed on Face data set with a 150 entry codebook, 2 feature models using 11x11 pixel features using roughly 130 foreground image, 170 background images and eight detection bins.

4.4 Thresholding

The significant effects of changing threshold values are examined in this section. The first problem is adjusting thresholds to be comparable across different detectors. The idea used by Weber [6] is to set an average number of feature detections ("candidates") per image. If there are a hundred images and the average candidates per image is three, then the code adjusts the threshold until there are three hundred detections. It is quite possible that any given image will have no detections or several times the average.

There are several different factors that change as the threshold is adjusted. Allowing more candidates means that the model learning has far more data to deal with, making the execution time significantly longer. However, a lower threshold means that fewer actual features will be missed. In the hard detection case, it means that more items will be lumped together as foreground feature detections.

After running tests where the average candidates per image ranged from 3 to 12, it appears that hard detection degrades rapidly, while soft detection peaks somewhere around 6 detections per image. Table 4.3 summarizes the results from each test condition and Figure 4.3 plots them in a more convenient form. Here it is evident that the number of tests is sub-optimal simply because the standard error bars are so large.

Logically, these results seem reasonable. In the hard threshold regime, the accuracy decreases as threshold increases because the distinguishing power inherent in the threshold decreases. (More false detections are lumped in with true detections) For soft detection, more candidates means more accurate distributions. Presumably as the number of candidates increase the number lumped into each bin increases, so eventually the same problem arises: the bins become over crowded and too generalized. The other factor that should

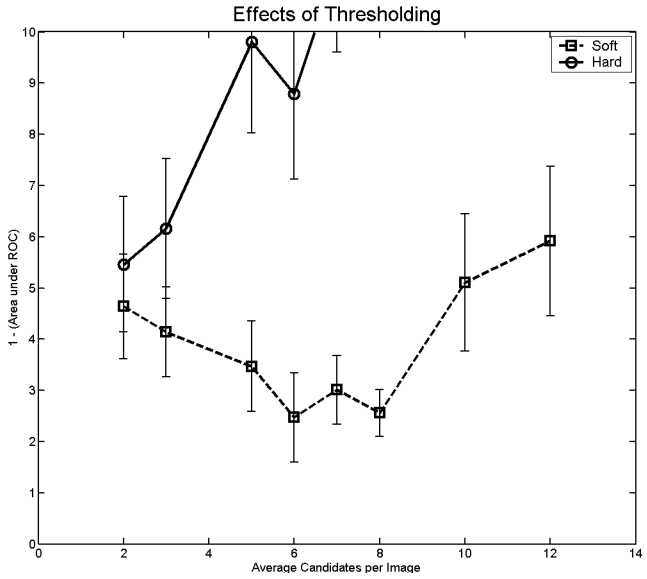

Figure 4.3: Above are $1 - A_{ROC}$ errors for soft and hard detection as the average number of candidates per image (and therefore the threshold) changes. The data set here is the Face data set and the error bars are standard errors.

be considered here is the increase in execution time. Although not recorded, more candidates significantly increase the execution time.

4.5 Number of Bins

The number of bins used to learn the probabilities of detector responses is fairly critical to success. In the limit of small numbers, a single bin is used, which is effectively the case of hard detection. However, as the number of bins increases, the number of detections that contributes to the bin during learning decreases, increasing the effects of noise and sparse data.

In Table 4.4 and Figure 4.4 results from tests varying the number of bins are shown. In these it is interesting to note that the validation errors continues to decline as the number of bins increases even after the test error troughs. Most likely this means that the algorithm is over-fitting the particular data that it is using to make decisions. This probably results from bins that specifically model very few detector responses, and fit them quite cleanly even though they deviate from the norm.

For comparison, the learned values for the bins are shown in Figure 4.5 (these graphs aren't actually for identical features, but the common distribution shape is dominant). While the corresponding models aren't identical, there is still a strong correlation between the shapes of the responses. As the number of bins increases from eight to sixteen bins, the error due to clumping large groups decreases, but the effects of randomness and over-fitting become more obvious.

	1 - Area under ROC		*Error Rate*	
Bins	*Test*	*Std. Err.*	*Test*	*Std. Err.*
4	3.20	0.89	6.67	1.61
8	2.48	0.54	5.97	0.96
12	2.82	0.61	8.06	1.45
16	3.10	0.93	6.90	1.51

Table 4.4: Data obtained by running learning with different numbers of soft detection bins. Each number represents an average over six tests. Tests performed on Face data set with a 150 entry codebook, 2 feature models using 11x11 pixel features using roughly 130 foreground image, 170 background images and eight candidates per image.

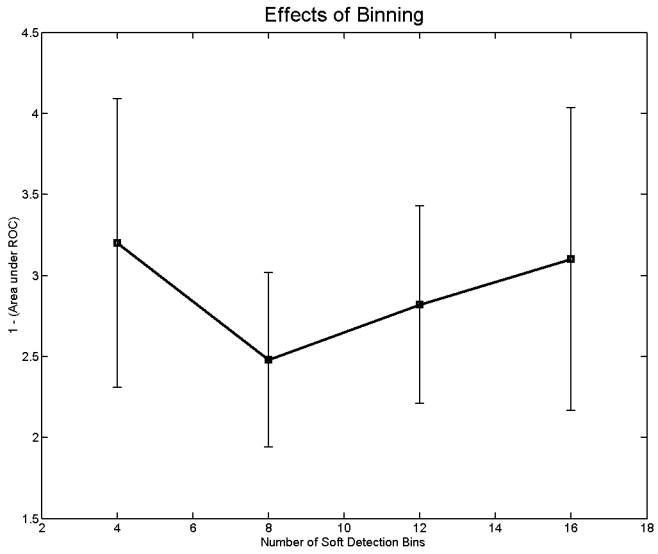

Figure 4.4: A graph of the error rate as the number of bins changes. As the number of bins increases, the error rate reaches a minimum before rebounding.

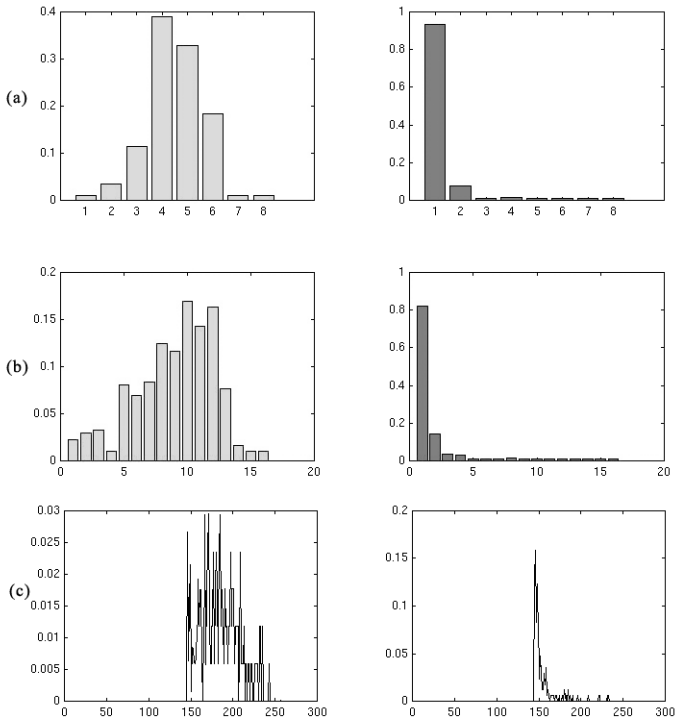

Figure 4.5: A detector response curve is learned for detecting faces with eight bins (a,) sixteen bins (b) and approx 110 bins above threshold (c). ((c) actually uses 256 bins but starts at R = 0 so only about 110 are above threshold)

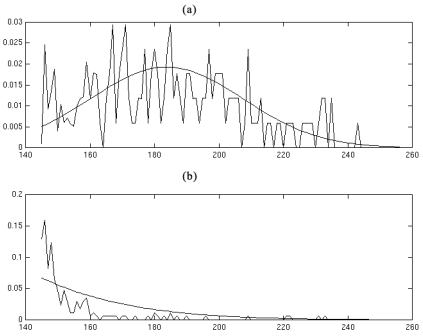

Figure 4.6: Learned detector responses with approx 110 bins and the corresponding guassians.

4.6 Robustness

While somewhat unconnected with the previous tests, it is important to know if the learned detector response distributions are robust with respect to noise. In order to ascertain this, results from learning the same model from several different starting points for EM are compared.

Fortunately, after several repetitive tests, the learned values (with a normalized sum of unity) were found to vary by a maximum of 10^{-16}. Considering the numerical accuracy of numerical storage on a PC, these variations are probably due to quantization errors. Because the average value of the bins is 1 / the number of bins, these small quantization errors are inconsequential.

4.7 Learning Gaussian Distributions

Because all of the detector responses so far follow the form of a gaussian, it only seems logical to try to fit the data to a gaussian and use the gaussian itself for detection. By setting the number of bins very high, and then using statistical measures, the foreground gaussian can be estimeted.

The background is more tricky: only a small portion of the data is observed. (See Figure 4.5 and remember that the background also takes the form of a guassian.) However, the mean and variance are actually known from a much earlier stage in the learning process when the images are first filtered by the detector. At that point all the responses are available, and a mean and variance can be computed.

This process was not successful. The results of learning using these new gaussians is significantly worse than using the bins. By looking at the learned values and the corresponding guassians in Figure 4.6 several problems can be noted. First, the learned values are incredibly sparse - but the guassians solve this problem. Second, the shape of the background rolloff does not correspond to the guassian. The two distributions come from relatively separate parts of the code, and thus provide significantly different results. Finally, part of the foreground response is still unobserved, so the guassian estimation is still incorrect.

However, given these flaws, the decline in performance should be minimal, and results should be better than hard detection. Despite expectations, the classifiers are almost useless, implying an implementation error.

Chapter 5

Conclusion

5.1 Summary

The results presented here show that soft detection is a superior method to hard detection for overall object recognition when adjusting parameters properly. However, the largest caveat is that there is an increase in execution time when reaping the benefits of soft detection. Furthermore, there are several parameters used in soft detection that can be optimized to improve results further.

These results are robust to several different data sets, and the learned values converge on the same values from different starting points.

5.2 Future Work

The first expansion to the work presented here is to complete the experiments with learned guassians. That model seems significantly superior, and should be studied further.

Ideally, one could come up with some algorithm to learn the parameters tested in the Experiments chapter of this paper. The number of bins and the threshold appear to be linked problems, and should only be solved in a combined context paying attention to the specific data being studied.

Another large addition would be to re-evaluate some decision criteria for the model learning in order to incorporate this new information more effectively. For example, it doesn't make much sense to learn a model where the features are known to have bad response curves. Instead, selecting features could possibly weed out a significant number of useless features in order to concentrate on the more important ones. Presumably the execution time would decrease accordingly.

Bibliography

[1] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995. Pages 1-73.

[2] M. Burl, T. Leung, and P. Perona. Face localization via shape statistics. *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, 1996.

[3] M.C. Burl. *Recognition of Visual Object Classes*. PhD thesis, Department of Electrical Engineering, California Institute of Technology, Pasadena, CA, 1997.

[4] T. Leung, M. Burl, and P. Perona. Finding faces in cluttered scenes using random labeled graph matching. *Proc. 5th Int. Conf. Computer Vision*, pages 637–644, May 1996.

[5] A. Papoulis and S. U. Pillai. *Probabilitiy, Random Variables and Stochastic Processes*. McGraw Hill, 2002.

[6] M. Weber. *Unsupervised Learning of Models for Object Recognition*. PhD thesis, Department of Electrical Engineering, California Institute of Technology, Pasadena, CA, May 2000.