Institute for Brain and Neural Systems

October 6, 2009

Departement of the Army
Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

Re: Final Report W911NF-04-1-0357

To Whom It May Concern:

This is written to provide a final report for the contract W911NF-04-1-0357 entitled "Visual analysis of complex scenes: breaking camouflage and detecting occluded objects using Bayesian inference". Within the scope of our project we developed a model for detection of targets in complex visual scenes that: a) is computationally efficient when analyzing vast amounts of information contained in the scenes, and b) is appropriate for detection of occluded and camouflaged targets. Our model has been implemented on a portable computer and tested on a variety of real-world images. The benefit of our system to the soldier is twofold: it can reduce the cognitive workload of the soldier operating in complex visual environments (such as those encountered in urban combat), and it can alert the soldier to possible threats that might otherwise be overlooked due to the camouflage or occlusion.

Leon N Cooper

Thomas J. Watson, Sr.
Professor of Science
Director, Institute for Brain and Neural Systems

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
06 OCT 2009		**00-00-2009 to 00-00-2009**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Institute for Brain and Neural Systems	**W911NF-04-1-0357**
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
Brown University,Providence ,RI,02912	**; 47010-LS.21**
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
U.S. Army Research Office, P.O. Box 12211, Research Triangle Park, NC, 27709-2211	11. SPONSOR/MONITOR'S REPORT NUMBER(S)
	47010-LS.21

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**	**Same as Report (SAR)**	**54**	

Contents

List of Figures

List of Tables

1 Statement of the problem studied

One of the primary problems faced in constructing a system for detection of targets in complex scenes is how to deal with computational complexity when analyzing large amounts of information in visual scenes. It seems natural that in addition to exploring advanced mathematical algorithms, in designing a model for scene analysis we should use some properties of the best existing machine for analyzing visual scenes the human visual system. The questions that we address are: 1) what are the properties of human vision that are most relevant for the task of object detection from visual scenes and 2) how can those properties be implemented in a working system for scene analysis.

One of the main objectives of our work is to develop a model for integrating information within a fixation and across fixations - during saccadic exploration of the visual scene. In our model, an object is represented with large number of features but, in contrast to other feature-based approaches, the learning of configurations of features does not require large quantities of training data. This is due to the fact that between an object and features we introduce an intermediate representation, object views. Specifically, in our model an object is represented as a collection of different views and each view is associated with different constellations of outputs of feature detectors. Given the location of the specific view, we show that each feature becomes conditionally independent of other features, which means that learning the whole configuration of features is then reduced to a much easier task - learning outputs of each feature detector independently of outputs of other feature detectors.

Another objective of our work is to develop new classification algorithms using methods from statistical pattern recognition and machine learning. Over the last fifteen years, significant advances had been made in constructing new and powerful classification algorithms such as support vector machines (SVM), boosting, and bagging. However, there are still numerous limitations related to selection of data for SVMs, use of unlabeled data, and learning in high dimensional spaces from few examples. Among the algorithms that we developed are a minimum bounding sphere algorithm for classifying object categories, adaptive distance nearest neighbor rule, a classification algorithms that can utilize information from unlabeled data, a model for data selection for SVMs, and a computational model for classifying both segmented and raw images.

2 Biologically inspired learning algorithms

2.1 Introduction

Detection and identification of partially occluded targets in complex scenes becomes an increasingly important task in light of the latest developments in urban warfare. The construction of a system that can automatically identify selected targets or direct soldiers attention to the locations that may contain suspicious activity can be of great use not only as a tool that can reduce the cognitive workload of the soldier but also as a tool that can alert the soldier to possible threats.

Identifying a target in a complex scene is a challenging problem that incorporates several important aspects of vision including: translation and scale invariant recognition, robustness to noise and ability to cope with significant variations in lighting conditions. Identifying an occluded target adds another layer of complexity and this problem can be extremely difficult even for humans. Motion information can be of great help in providing an initial figure-ground segmentation. However, in many situations motion information is not available. In addition, if the input to the system is a video stream then the requirement that the system works in real-time often precludes the use of more sophisticated but computationally involved techniques.

One of the main limitations of classical vision algorithms, such as those utilizing Artificial Neural Networks (ANNs), Radial Basis Functions (RBFs), and Support Vector Machines (SVMs), is that they require a fixed size input. This means that during the recognition phase the input vector to the system has to be of the same size as the input vector used during the training process. Such systems are therefore not well suited for occlusion problems where sections of the input vector are simply missing or carry incorrect information.

In addition, supplying a fixed size input to the recognition system requires the selection of the specific region from the image. This means that such systems have to solve the segmentation problem, find the boundary of the region occupied by the target. However, given an image, it is not known where the target is or what its size is. In order to detect a target, regardless of its location, the detection system is usually (as presented in (Schneiderman and Kanade, 2000)) convolved over the whole image and in order to detect a target at different scales the original image is rescaled and the convolution procedure repeated. Since the methods that rely on exhaustive search are not computationally efficient, they are mostly applied to detection of targets in static images.

Human visual system, on the other hand, does not require any "presegmentation" of the image in order to recognize a specific object. In fact, when we look at an object, our visual system processes not only information coming from the object itself but the whole scene. This is accomplished through an array of neurons that are selective to specific features and whose receptive fields (RFs) are spatially distributed and localized. Although our visual system processes information from

all the regions of the scene, it appears as if it somehow knows to "discard" certain regions (the background) and integrate only information from the object regions. If we are not able to recognize an object from a single fixation, then we make saccades, combine evidence from different fixations and as a result usually improve our perception of the object.

Since our visual system integrates information from neurons that have localized receptive fields, it seems natural to represent an object as a collection of localized features. In contrast to *global* models, such as those that use a Principal Components Analysis (PCA) approach, feature-based approaches are much more robust to partial occlusions. Over the past years, feature-based approaches had become increasingly popular within the computer vision community (Lowe, 1999; Schmid and Mohr, 1997; Serre et al., 2005; Heisele et al., 2001; Torralba et al., 2004). These approaches have been succesfully used in various applications such as face recognition (Schneiderman and Kanade, 2000; Viola and Jones, 2001), handwriting recognition (Wang et al., 2005c; Neskovic et al., 2000), car detection (Agarwal et al., 2004; Schneiderman and Kanade, 2000; Neskovic et al., 2004), and modeling human bodies (Felzenszwalb and Huttenlocher, 2005). One of the problems of probabilistic feature based approaches (such as (Fei-Fei et al., 2003)) is that they can not model an object with a large number of features since calculating the joint probabilities would require an enormous amount of training data. Another problem is how to find the best constellation of features. In one-dimensional case this problem can be solved using a dynamic programming approach but for two dimensional case this is still an open problem and no exact solution that is at the same time computationally efficient exists today. In contrast to approaches presented in (Fei-Fei et al., 2003; Serre et al., 2005), our model uses much simpler features and does not require a feature learning stage. Furthermore, unlike the model of Fei-Fei *et al.*, our system can use an arbitrarily large number of features without an increase in computational complexity.

The main question therefore is how to deal with computational complexity when analyzing large amounts of information contained in visual scenes. It seems natural, that in designing a system for scene analysis we should use some properties of the best existing system for analyzing visual scenes - the human visual system. Unfortunately, biologically inspired models (Keller et al., 1999; Rybak et al., 1998) and models of biological vision (Amit and Mascaro, 2003; Mel, 1997; Riesenhuber and Poggio, 1999) have been much less successful (in terms of real-world applications) compared to computer vision approaches. A model that captures some properties of human saccadic behavior and represents an object as a fixed sequence of fixations has been proposed by Keller *et al.* (Keller et al., 1999). Similarly, Hecht-Nielsen and Zhou (Hecht-Nielsen. and Zhou, 1995) and Rybak *et al.* (Rybak et al., 1998) presented models that are inspired by the scanpath theory (Noton and Stark, 1971). Although these models utilize many behavioral, psychological and anatomical concepts such as separate processing and representation of "what" (object features) and "where" (spatial features: elementary eye movements) information, they still assume that an object is represented as a sequence of eye movements. In contrast to these approaches, our model (Neskovic

et al., 2006a) does not assume any specific sequence of saccades and therefore is more general.

2.2 Bayesian integrate and shift model

When we look at an object, our visual system processes not only information coming from the object itself but the whole scene. This is accomplished through an array of neurons that are selective to specific features and whose receptive fields (RFs) are spatially distributed and localized. Although our visual system processes information from all the regions of the scene, it appears as if it somehow knows to "discard" certain regions (the background) and integrate only information from the object regions.

Our approach for integrating information from different regions of the scene, given a fixation point, utilizes Bayesian inference (Neskovic et al., 2006a). In our model, an object is represented as a collection of features of specific classes arranged at specific locations with respect to the location of the fixation point. Even though the number of feature detectors that we use is large, we show that learning does not require a large amount of training examples. This is due to the fact that between an object and features we introduce an intermediate representation, object views, and thus obtain conditional independence of the outputs of the feature detectors. In order to learn object views, the system utilizes experience from a teacher. Although this paradigm at first appears more user intensive than paradigms that provide only class information to the system, it is actually very fast since the system can learn object categories using only few training examples.

Our model falls into a category of feature-based approaches (Fei-Fei et al., 2003; Lowe, 1999; Schneiderman and Kanade, 2000; Serre et al., 2005; Torralba et al., 2004; Viola and Jones, 2001). The problem that we want to solve is as follows: given a collection of features, their locations $\vec{X}$, and appearances $\vec{A}$ we want to calculate the probability that they represent an object of a specific class n, $P(O^n|\vec{X},\vec{A})$. Since calculating this probability is extremely difficult if the number of features is large, we seek to find suitable approximations. One of the biggest simplifications is to assume that the feature locations are fixed and that all the variations are due to appearances. Unfortunately, this is one of the least reasonable assumptions which holds in only few practical situations.

In order to make the model more realistic, one should include tolerance to variations in feature locations. Instead of assuming that a feature is located at a point, we will assume that it is located within a region. The question is how to design these regions? If we use large regions, we can then easily capture all possible variations in feature locations (excellent generalization) but at the expense of losing location specificity which would decrease discrimination capability of the model. On the other hand, very small regions would provide excellent localization but would lead to poor generalization. We propose that the solution to this trade-off between generalization and retaining location specificity is to use retina-like distribution of regions in combination with saccade-like shifts. If we want to estimate the location of a specific feature, then the size of the region where it can be found (the uncertainty) depends on the location of the point with respect to which we

measure its distance - the center. The further away the feature is from that center, the larger the uncertainty. Therefore, in order to capture variations in feature locations, the sizes of the regions, as well as their overlaps, have to increase with their distance from the center. As a consequence, the accuracy of estimating feature locations is high only for the features that are close to the center. In order to obtain good location estimates for the features that are further away from the center, the recognition system would have to shift the center, to make a "saccade".

2.2.1 The model

Let us assume that we are given an array of feature detectors whose RFs form a grid and completely cover an input image. One RF has a special role during the recognition process. We call it the central RF and the region of the image over which this RF is positioned the fixation region. Similarly, the center of the fixation region is the fixation point. Since the location of each feature is measured with respect to the central RF, the uncertainty associated with feature's position increases with its distance from the fixation point. In order to capture variations in feature locations, the sizes of the RFs of the feature detectors have to increase with their distance from the central RF. Similarly, the overlap among the RFs increase with the distance from the central location.

Object Views. We will call a configuration consisting of the outputs of feature detectors associated with a specific fixation point a *view*. That means that there can be as many views for a given object as there are points within the object and that number is very large. In order to reduce the number of views, we will assume that some views are sufficiently similar to each other so that they can be clustered into the same view. A region that consists of points that constitute the same view we call a view region.

Notation. With symbol H_i^n we denote a random variable with values $H = (n, i) = H_i^n$ where n goes through all possible object classes and i goes through all possible views within the object. Therefore, the symbol H_i^n denotes the i^{th} view of an object of the n^{th} (object) class. With the symbol D_k^r we denote a random variable that takes values from a feature detector that is positioned within the RF centered at $\vec{y}_k$ from the central location, and is selective to the feature of the r^{th} (feature) class, $D_k^r = d(\vec{y}_k)$. The locations of the fixation points, the central locations, are indexed with time variable, $\vec{x}_t$. We denote the collection of the outputs of the feature detectors, given the central location $\vec{x}_t$ at time t with the symbol $A\{d, \vec{x}_t\}$.

2.2.2 Combining information within a fixation

Let us now assume that for a given fixation point $\vec{x}_0$, the feature of the r^{th} class is detected with confidence $d^r(\vec{y}_k)$ within the RF centered at $\vec{y}_k$. The influence of this information on our hypothesis, H_i^n, can be calculated using Bayesian rule as

$$p(H = H_i^n | D_k^r = d^r(\vec{y}_k), \vec{x}_0) = p(H_i^n | d^r(\vec{y}_k), \vec{x}_0) = \frac{p(d^r(\vec{y}_k) | H_i^n, \vec{x}_0) p(H_i^n | \vec{x}_0)}{p(d^r(\vec{y}_k) | \vec{x}_0)}, \quad (1)$$

where the normalization term indicates how likely it is that the same output of the feature detector can be obtained (or "generated") under any other hypothesis,

$$p(d^r(\vec{y}_k)|H_i^n, \vec{x}_0) = \sum_{n,i} p(d^r(\vec{y}_k)|H_i^n, \vec{x}_0) p(H_i^n|\vec{x}_0). \tag{2}$$

We will now assume that a feature detector with RF centered around $\vec{y}_q$ and selective to the feature of the p^{th} class outputs the value $d^p(\vec{y}_q)$. The influence of this new evidence on the hypothesis can be written as

$$p(H_i^n|d^p(\vec{y}_q), d^r(\vec{y}_k), \vec{x}_0) = \frac{p(d^p(\vec{y}_q)|d^r(\vec{y}_k), H_i^n, \vec{x}_0) p(H_i^n|d^r(\vec{y}_k), \vec{x}_0)}{p(d^p(\vec{y}_q)|d^r(\vec{y}_k), \vec{x}_0)}. \tag{3}$$

The main question is how to calculate the likelihood term $p(d^p(\vec{y}_q)|d^r(\vec{y}_k), H_i^n, \vec{x}_0)$? In principle, if the pattern does not represent any object, the outputs of the feature detectors $d^p(\vec{y}_q)$ and $d^r(\vec{y}_k)$ are independent of each other. On the other hand, if the pattern represents a specific object, then the local regions of the pattern within the detectors receptive fields are not independent from each other. However, once we introduce a specific hypothesis, the outputs of feature detectors again become independent of each other, but this time only conditionally independent, given the hypothesis. The likelihood term can therefore be written as $p(d^p(\vec{y}_q)|d^r(\vec{y}_k), H_i^n, \vec{x}_0) = p(d^p(\vec{y}_q)|H_i^n, \vec{x}_0)$. Note that the conditional independence is not an assumption (like a "naive Bayes") but a consequence of the model we use. This property is very important from the computational point of view and allows for very fast training procedure. The dependence of the hypothesis on the collection of outputs of feature detectors $A\{d, \vec{x}_0\}$ can be written as

$$p(H_i^n|, A\{d, \vec{x}_0\}, \vec{x}_0) = \frac{\prod_{rk\in A} p(d^r(\vec{y}_k)|H_i^n, \vec{x}_0) p(H_i^n|\vec{x}_0)}{\sum_{n,i} \prod_{rk\in A} p(d^r(\vec{y}_k)|H_i^n, \vec{x}_0) p(H_i^n|\vec{x}_0)} \tag{4}$$

where r, k goes over all possible feature detector outputs contained in the set A and n, j goes over all possible hypotheses.

2.2.3 Combining information across fixations

We now calculate how the evidence about the locations of different fixations influence the confidence about the specific hypothesis, H_j^n, associated with fixation point $\vec{x}_t$. We assume that at time $t-1$ a hypothesis has been made that the fixation at distance $\vec{z}_{t-1}^{\,i}$ from the current fixation represented the center of the i^{th} view of the object of the n^{th} class. Similarly, we will assume that at time $t-2$ a hypothesis has been made that the fixation at distance $\vec{z}_{t-2}^{\,k}$ from the current fixation represented the center of the k^{th} view. We denote with the symbol A_t the outputs of all the feature detectors that are used to calculate the (new) hypothesis H_j^n. The influence of the evidence about the locations of the previous hypotheses on the current hypothesis can be written as

$$p(H_j^n|\vec{z}_{t-1}^{\,k}, \vec{z}_{t-2}^{\,i}, A_t, \vec{x}_t) = \frac{p(\vec{z}_{t-1}^{\,k}|H_j^n, \vec{z}_{t-2}^{\,i}, A_t, \vec{x}_t) p(H_j^n|\vec{z}_{t-2}^{\,i}, A_t, \vec{x}_t)}{p(\vec{z}_{t-1}^{\,k}|\vec{z}_{t-2}^{\,i}, A_t, \vec{x}_t)}. \tag{5}$$

In order to make the model computationally tractable, we will assume that the view locations are independent from one another given the hypothesis.

Since the location of the k^{th} view of the object does not depend on the configuration of feature detectors that is associated with the current view, and assuming that view locations are independent from one another, the likelihood term from Eq. (5) becomes $p(\vec{z}^{k}_{t-1}|H^n_j, \vec{z}^{i}_{t-2}, A_t, \vec{x}_t) = p(\vec{z}^{k}_{t-1}|H^n_j, \vec{x}_t)$. The probability that the input pattern represents the j^{th} view of the object of the n^{th} class, given the activations of the letter detectors A_t and locations of other views B_t can be written as

$$p(H^n_j|A_t, \vec{x}_t, B_t, f(s)) = \frac{\prod_{s<t} p(\vec{z}^{f(s)}_s|H^n_j, \vec{x}_t)p(H^n_j|A_t, \vec{x}_t)}{\sum_i \prod_{s<t} p(\vec{z}^{f(s)}_s|H^n_j, \vec{x}_t)p(H^n_j|A_t, \vec{x}_t)} \tag{6}$$

where i goes through views of the n^{th} object, s goes through the locations of all the fixations and the function $f(s)$ maps a location $\vec{y}_s$ to a specific hypothesis. With symbol B_t we denoted the set of the locations of all the fixations (object views) with respect to the location of the current fixation, $\vec{x}_t$.

2.3 Implementation

Modeling Likelihoods. We model the likelihoods in Eq. (4) using Gaussian distributions. The probability that the output of the feature detector representing the feature of the r^{th} class and positioned within the receptive field centered at $\vec{y}_k$ has a value $d^r(\vec{y}_k)$, given a specific hypothesis and the location of the fixation point, is calculated as

$$p(d^r(\vec{y}_k)|H^n_i, \vec{x}_t) = \frac{1}{\sigma^r_k\sqrt{2\pi}} exp \frac{-(\mu^r_k - d^r(\vec{y}_k))^2}{2(\sigma^r_k)^2} \tag{7}$$

This notation for the mean and the variance assumes a particular hypothesis so we omitted some indices, $\sigma^r_k = \sigma^r_k(n, i)$. The values for the mean and variance are calculated in the batch mode but, as we will see in the next section, only a small number of instances are used for training so the memory requirement is minimal.

Feature Extraction. We extract features using a collection of Gabor filters where a Gabor function that we use is described with the following equation

$$\psi_{f_0,\theta,\sigma}(x, y) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{8\sigma^2}(4(xcos\theta+ysin\theta)^2+(ycos\theta-xsin\theta)^2)} sin(2\pi f_0(xcos\theta + ysin\theta)). \tag{8}$$

One way to relate the spatial frequency and the bandwidth is using the expression: $2\pi f_o\sigma = 2\sqrt{ln2}(2^\phi+1)/(2^\phi-1)$ (see (Lee, 1996) for more detail). Since the spatial frequency bandwidths of the simple and complex cells have been found to range from 0.5 to 2.5 octaves, clustering around 1.2 octaves, we set ϕ to 1.5 octaves. The orientations and bandwidths of the filters are set to: $\theta = \{0, \pi/4, \pi/2, 3\pi/4\}$ and $\sigma = \{2, 4, 6, 8\}$.

Designing Receptive Fields. We use two different methods for arranging RFs. In the first method, which we call the *directional distribution*, we arrange the RFs along different directions while in the second method, which we call the *circular distribution*, we place the RFs along concentric rings (Schwartz, 1977; Wilson, 1983; Smeraldi and Bigun, 2002; Gomes, 2002).

a) *Directional distribution.* In this implementation, we use four parameters to control the distribution of the RFs: the rate of increase of the sizes of the RFs, the overlap between the adjacent RFs, the size of the central RF, and the number of directions along which the centers of the RFs are placed. Each RF has a square form and the RFs are arranged is such a way as to completely cover the input image. Therefore, in order to satisfy a complete coverage requirement, not all the values of the parameters can be used. The overlap between two RFs is defined as a percentage of the area of the smaller of the two RFs that is being covered. For example, if the $ovr = 50\%$, that means that the larger RF covers 50% of the area of the smaller receptive field.

The main shortcoming of the directional distribution is that it does not provide a sufficiently dense packing of the RFs, especially for regions that are further away from the central field. This is shown in Figure 1 (left) where, for illustrative purpose, we use the circular RFs. In order to prevent the gaps between the RFs, and completely cover the visual field, the rate of increase of the sizes of the RFs has to be sufficiently high.

b) *Circular distribution.* Another solution is to arrange the centers of the RFs using a hexagonal packing, as shown in Figure 1 (right). Within each ring we use a fixed number of RFs, which we set to 10. The radius of a RF, $r(n)$, whose center is on the n^{th} ring, is calculated as $r(n) = B \cdot r(n-1)$, where B is the enlarge parameter. The angle between neighboring RFs from the same ring is called the *characteristic angle*, θ_o, and is calculated as $\theta_o = 2\pi/F$, where F is the number of RFs per ring. In this arrangement, the position of each RF is fully determined by the ring number and the angle, θ, with respect to a chosen direction. For example, the angle of the m^{th} RF is calculated as $\theta_m = m \cdot \theta_o$.

A hexagonal packing is obtained by shifting the angles of all the RFs in all even rings by half the characteristic angle. This disposition of the centers of the RFs is also known as triangular tessellation. The radius of an n^{th} ring, $R(n)$, is calculated using the following equation:

$$R(n) = R(n-1) + r(n) + r(n-1)(1 - 2 * ovr).$$

In our implementation, each RF has a square form and the size of the smallest RF is 31x31 pixel. The RFs are arranged along 8 directions and the sizes of the RFs are increased at the ratio of 1.4 (controlled by the enlarge parameter). For example, the sizes of the RFs that are nearest neighbors to the central RF are (31x1.4)x(31x1.4). The overlap between two neighboring receptive fields is 50% meaning that for two neighboring RFs, the larger RF covers 50% of the area of the smaller receptive field. The recognition results are not very sensitive to the small changes in the overlap, enlarge parameter, and the sizes of the receptive fields.

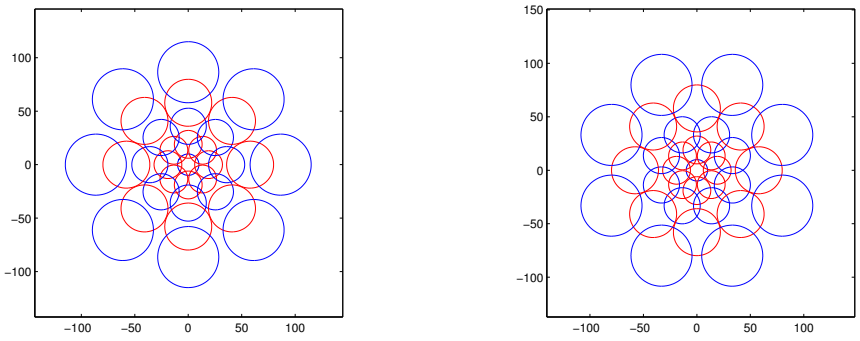

Figure 1: Comparison between the directional and circular arrangements of the RFs. Left: RFs are arranged along 8 directions. Right: RFs are arranged along 4 concentric rings, each ring containing 8 RFs. The circular arrangement provides denser packing of the RFs, especially for the regions that are further away from the center.

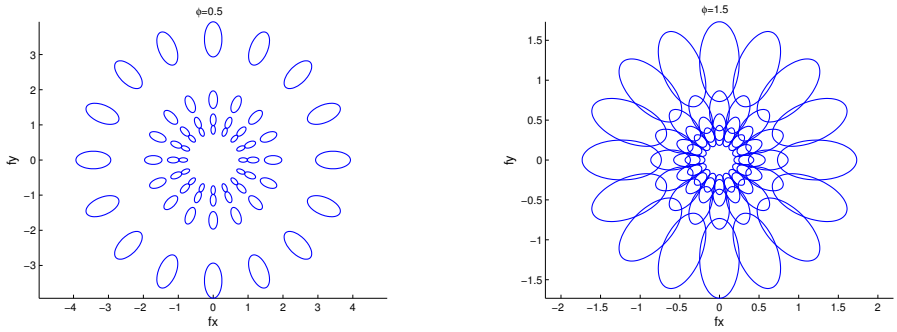

Figure 2: Ensemble of Gabor filters in a frequency plane with $\phi = 0.5$ (left) and $\phi = 1.5$ (right). The arrangement on right provides a better coverage.

Feature Detectors. With each RF we associate 16 feature detectors where each feature detector signals the presence of a feature (i.e. an edge of specific orientation and size) to which it is selective no matter where the feature is within its receptive field. One way to implement this functionality is to use a max operator. The processing is done in the following way. On each region of the image, covered by a specific RF, we first apply a collection of 16 Gabor filters (4 orientations and 4 sizes) and obtain 16 maps. Each map is then supplied to 16 feature detectors where each feature detector finds a maximum over all possible locations. As a result, a feature detector, associated with a specific Gabor filter, finds the strongest feature within its RF but does not provide any information about the location of that feature.

The Training Procedure The training is done in a supervised way. We constructed an interactive environment that allows the user to mark a section of an object and label it as a fixation

region associated with a specific view. Every point within this region can serve as the view center. Once the user marks a specific region, the system samples the points within it and calculates the mean and variance for each feature detector. Since the number of training examples is small the training is very fast.

Note that during the training procedure the input to the system is the whole image and the system learns to discriminate between an object and the background. It is important to stress that the system does not learn parts of the object, but the whole object from the perspective of the specific fixation point.

2.4 Learning a single object category

We tested the performance of our system using the Caltech database (www.vision.caltech.edu). The system was first trained on background images in order to learn the "background" hypothesis. We used 20 random images and within each image the system made fixations at 100 random locations. The system was then trained on specific views of specific objects. For example, in training the system to learn the face from the perspective of the right eye, the user marks with the cursor the region around the right eye and the system then makes fixations within this region in order to learn it. The system was tested on random images and for each fixation point we calculated the probability that the configuration of the outputs of feature detectors represents a face from the perspective of the right eye. Since the system didn't make a single mistake when fixating on locations that belong to the background, in order to make the problem more difficult, we tested the performance using only the face regions. Each new face was divided into the right eye region and the rest of the face. The system made 40 random fixations within the region of the right eye (positive examples) and 200 random fixations outside the region of the right eye (negative examples). The number of training examples that we used varied from 1 to 10. The system was tested on 5 new faces (of people that were not used for training). Therefore, we used 200 positive examples and 1000 negative examples for testing. As a measure of performance we use the error

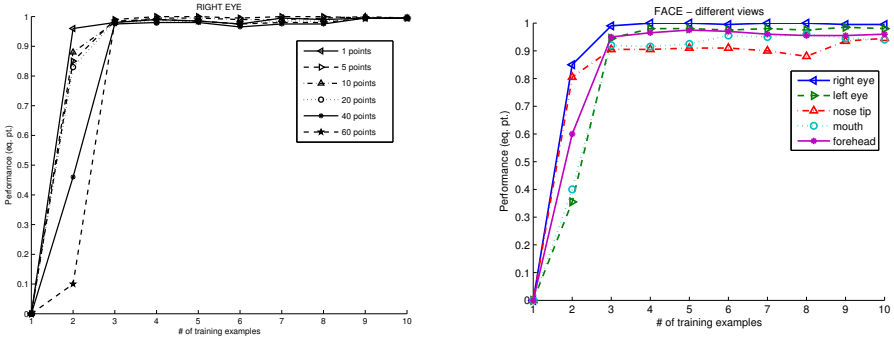

Figure 3: Left: Performance as a function of the number of training examples and sampling points. Right: Performance comparison for different views.

rate at equilibrium point (as in (Serre et al., 2004)) which means that the threshold is set so that the miss rate is equal to the false positive rate. The performance depends both on the number of faces used for training and on the number of sampling points that the system used in order to learn the view. The results are illustrated in Figure 3 (left). As we can see, using more sampling points is not necessarily better especially if the number of training faces is small. This is to be expected since the system becomes biased to the training face(s). On the other hand, using just a single fixation per face is not sufficient if the number of faces is small since the system cannot estimate the variances. For learning a view using one fixation and only one training face we set the variance by hand and in Figure 3 (left) this number just happened to be a good guess. In order for the system to learn the face (and "discard" information from the background) it has to be presented with more than one face. As it turns out, two examples are not quite enough but with three examples the system can learn the face (the specific view of the face) with high confidence. In all of the experiments that follow, we set the number of saccades per view (the number of sampling points to 10). The performance of the system using different views of a face is illustrated in Figure 3 (right). It is clear that the easiest views are the right and the left eye while the tip of the nose required more training examples.

2.5 Learning multiple object categories

We also tested the performance of our system on four object categories (faces, cars, airplanes and motorcycles) using the Caltech database, Figures 4 and 5.

Figure 4: View regions as selected by the teacher.

Although the performance of the system is very good using only a single view, we tested whether

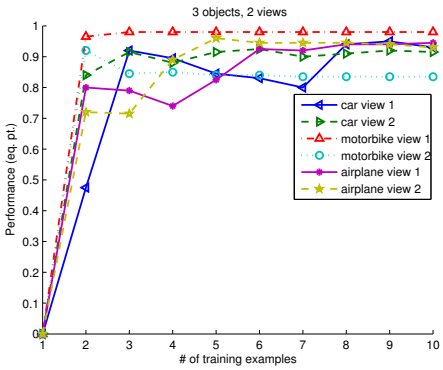

Figure 5: Performance graphs for different object categories.

Table 1: Multiple Views

	Faces					Cars			
	1	**2**	**3**	**4**		**1**	**2**	**3**	**4**
r. eye	98.0 %	99.0 %	99.5 %	100 %	v1	88.2 %	91.1 %	94.2%	95.8 %
l. eye	94.5 %	99.5 %	100 %	100 %	v2	86.6 %	91.5%	93.3 %	94.0 %
nose	90.5 %	94.0 %	96.5 %	97.5%	v3	88.9 %	90.4 %	93.0 %	94.9 %
mouth	92.0 %	97.5 %	98.5 %	99.0 %	v4	84.4 %	90.7 %	92.5%	93.2 %

and how much information from other fixations improves the performance, Table 1. The tests were done on faces and cars and we used 4 very good views for faces and 4 below the average views for cars. In both cases the information about the spatial location of other views improved the performance. During the training phase, the user marks the fixation (view) regions and the system then calculates the location likelihoods for each pair of regions separately by randomly selecting n points from each region. During the testing phase, in order to estimate the location of the view center, the system selects 10 points with the highest probabilities (as representing the view) and takes the average over their locations.

The system was first trained on individual views and the results are illustrated in column **1**. When the system used information about the location of one more veiw, the performance improved, as shown in column **2**. Utilizing information about the location of two different views further improved the performance as captured by the numbers in column **3**. The best performance, as expected, was obtained when the system used the information about centers of the three views as shown in column **4**.

In order to verify whether high recognition rates can be achieved using a uniform distribution of the RFs, we tested the system using the grid of the RFs of the same size, and repeated the

Table 2: Performance Using Uniform Distribution of the RFs

RF Size	10	20	30	40	50	60	70	80	90	100	110
Performance (%)	60.0	60.0	57.5	59.5	60.8	70	69	75.7	78.5	73.5	73.5

experiment choosing different RF sizes. In Table 2 we illustrate the performance of the system when trained on a face using the tip of the nose as a fixation region - the "nose view". The performance is much worse compared to retina-like distribution and, as expected, the system has difficulties learning the view using small RFs. Using large RFs recognition improves but only to the point and then decreases again.

2.6 Detection of partially occluded targets

In this section, we present the results of our model (Neskovic et al., 2006a) when tested on three types of occlusions: a) the bar covering both eyes (denoted as cover 1 in Figure 6, b) two large disconnected regions covering the face (cover 2), and c) the rectangle covering the face below the nose (cover 3). Tests were done on face images of people that were not used for training. As one can see, system can recognize the face even when the fixating region is covered (Figure 7, top), which means that it utilizes information from the whole face and not only local information around the fixation point. We use yellow stars to display correctly detected positive fixations, green stars for correctly detected negative fixations, red stars for missed fixations, and blue stars for false alarm fixations, Figure 7. Incorrect fixations are bigger in size.

The training was done on different instances of one category and tested on partially occluded examples that the system had never seen before. We demonstrate that the system is very robust to occlusions and clutter and can recognize a target even if it fixates on the occluded part.

The system was first trained on background images in order to learn the "background" hypothesis. We used 20 random images and within each image the system made fixations at 100 random locations. The system was then trained on specific views of specific objects. For example, in training the system to learn the face from the perspective of the right eye, the user marks with the cursor the region around the right eye and the system then makes fixations within this region in order to learn it.

Since our system uses information from over 1,000 feature detectors distributed over the whole image, it is very robust to occlusions. This is demonstrated in Figure 6, bottom, that illustrates the performance of the system when tested on occluded images as shown in Figure 6, top . The system was trained to recognize a face category from the perspective of the right eye (righ-eye-view) using

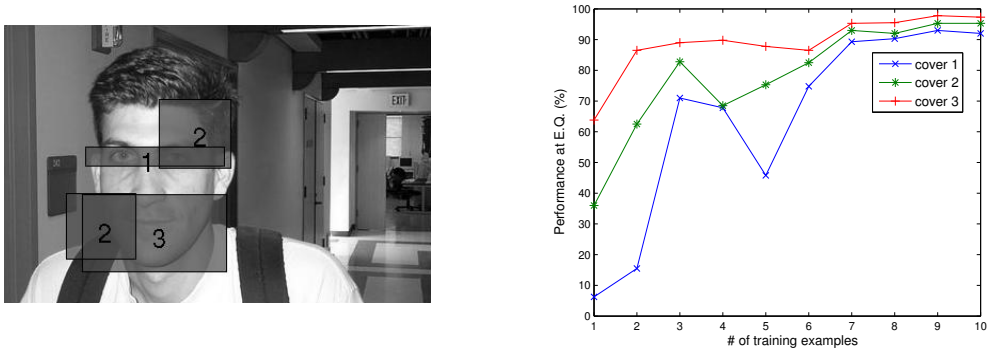

Figure 6: Top: Three different occlusions used for testing the system on the face it has never seen before. The task is to detect a face and estimate the location of the right eye. Bottom: Corresponding performances under different occlusions.

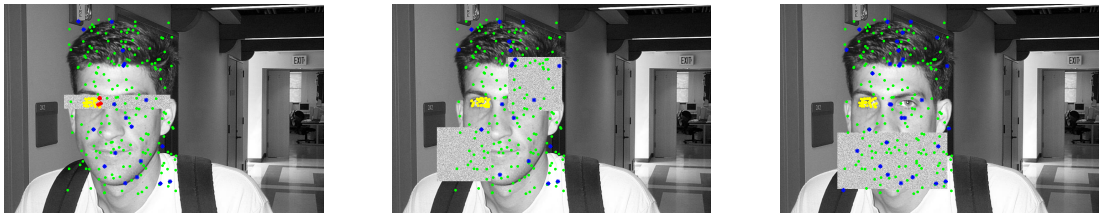

Figure 7: Performance of the system on different face occlusions. Yellow stars denote correctly detected positive fixations, green stars denote correctly detected negative fixations, red stars denote missed fixations, and blue stars denote false alarm fixations.

(non-occluded) examples from different people.

2.7 Dependence of the BIAS model on the sizes and location of the fixation regions

In this section we describe the experimental setup and provide classification results of the recognition system when trained on different configurations of fixation regions (Neskovic et al., 2009).

In the first experiment, we used 9 different fixation regions selected by the teacher. The instruction given to the teacher was: "segment a face into 9 regions that you think are perceptually important". For training, we used N images where N goes from 1 to 10. Each of the training images represented a different person. For testing, we randomly selected 2 images from each subject (if possible) and in total we used 40 images. Most of the faces in the testing set were from different persons compared to the training set. We repeated the experiment 5 times.

In Figure 8, we show nine fixation regions as selected by the user. These regions serve as a

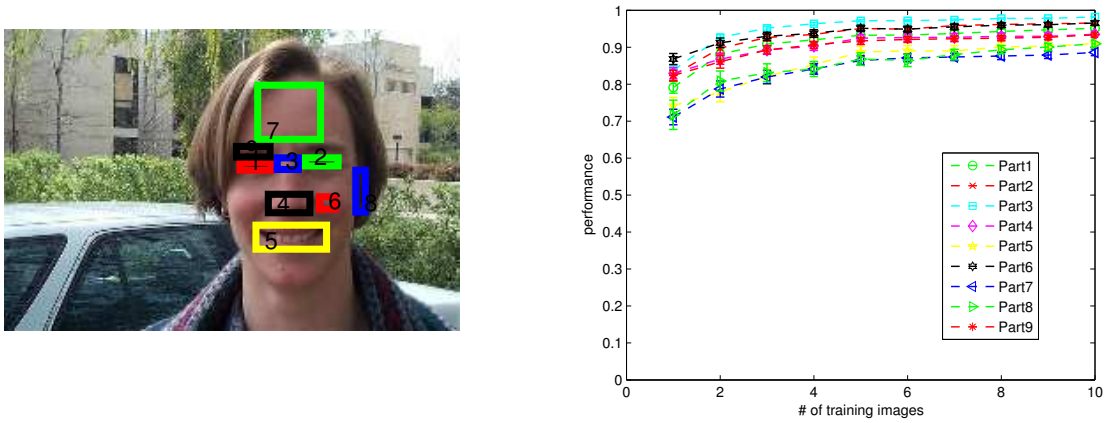

Figure 8: Top: Nine fixation regions used for training the recognition system. Each region is numbered and those numbers appear in the performance graphs as "part numbers". Bottom: Performance of the system.

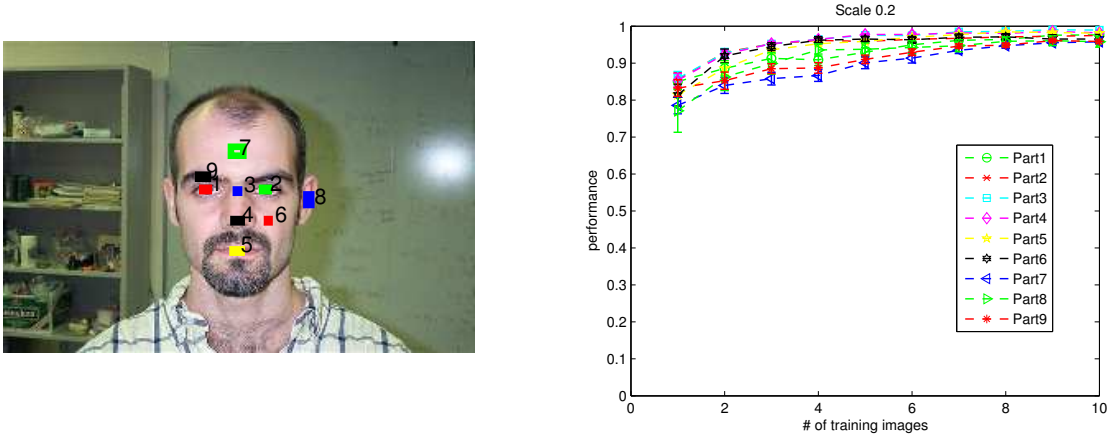

Figure 9: Performance of the system (bottom image) when using the configuration of the fixation regions illustrated in the top image. The sizes of the fixation regions are reduced by the factor 0.2 compared to the sizes illustrated in Figure 8.

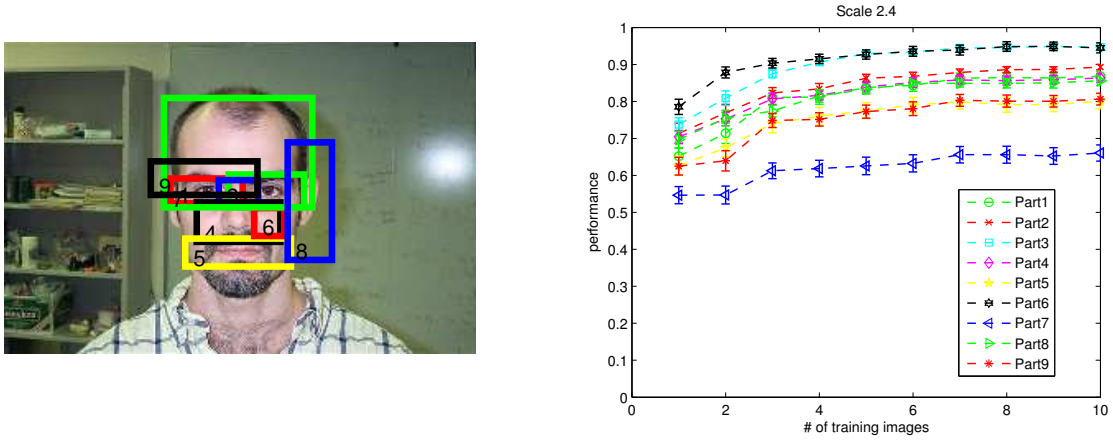

Figure 10: Performance of the system (bottom image) when using the configuration of the fixation regions illustrated in the top image. The sizes of the fixation regions are increased by the factor 2.4 compared to the sizes illustrated in Figure 8.

"baseline" against which we then contrast regions of different sizes and locations.

One can see that the highest performance is not achieved by focusing on the eye regions but rather on the region between the eyes, part 3. One reason for this is that the region 3 is the smallest in size and another reason is that it is close to the center of the face. The fact that the regions 4 and 5 also produce higher classification rates compared to other regions also confirms that the system prefers locations that are close to the center of the face.

In the second set of experiments we investigated the importance of the sizes of the fixation regions. We tested the performance using the scale factors ranging from 0.1 to 2.5. A scale factor of 0.1 means that the size of each RF was 10 times smaller than the size selected by the teacher. In Figure 9 we illustrate the results when using the scale factor 0.2 and in Figure 10 we illustrate the results using the scale factor 2.4. As expected, the performance is inversely proportional to the sizes of the fixation regions. Furthermore, the performance of the system when using small fixation regions, Figure 9, is even better than the performance when using regions selected by the human teacher. It is interesting to note that the system is able to learn the face category even when the fixation region includes a large number of points that are not on the face, as is clearly the case with the region 8 in Figure 10. Another trend is that the error bars (the variances) decrease with the number of the training examples, which is to be expected.

In the third set of experiments we investigated the importance of the locations of the fixation regions using 10 different random configurations. In Figures 11-12 we show examples of two such configurations. For each random configuration, it is important to be consistent in choosing the same relative positions of the fixation regions from one image to another. This can be accomplished in a number of different ways. For example, once a random configuration is generated, the location

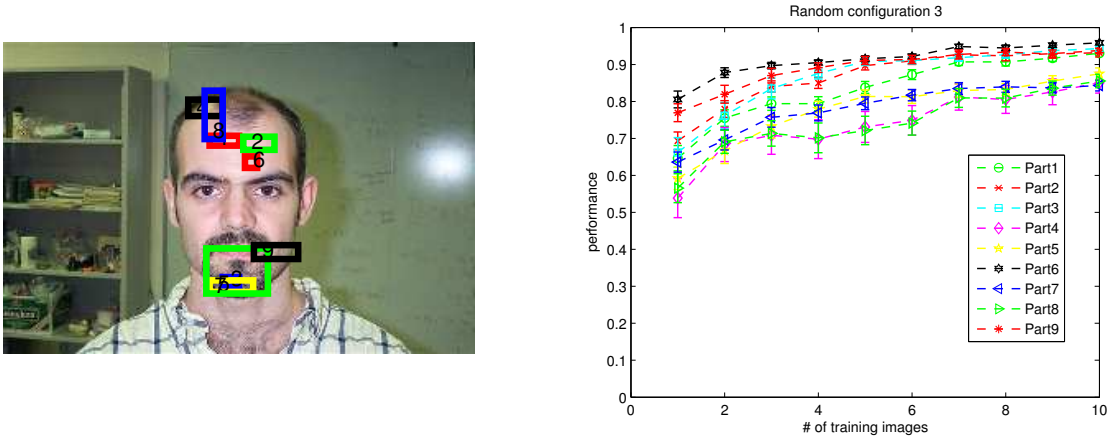

Figure 11: An example of a random configuration and corresponding performances.

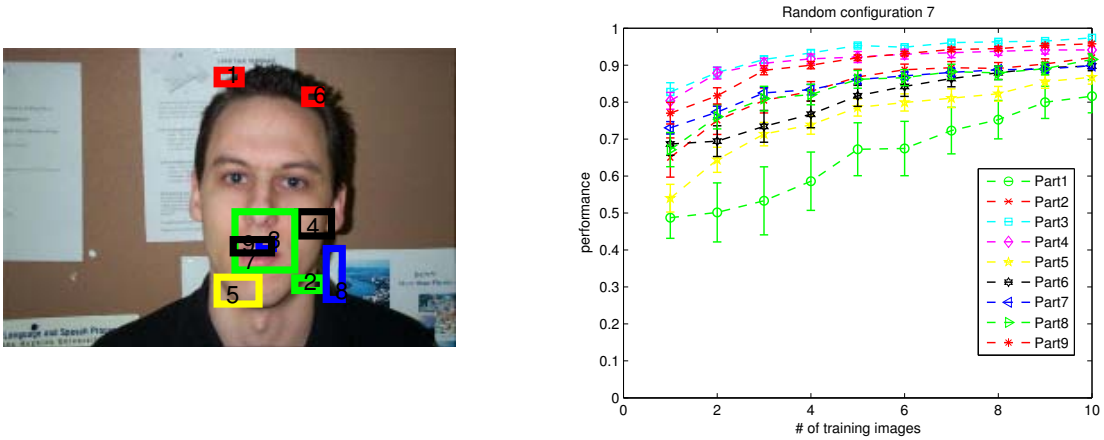

Figure 12: An example of a random configuration and corresponding performances.

of each region can be measured with respect to the location of a single fixed point within a face. However, this approach is not very robust since it depends on the choice of the fixed point and it also does not take into consideration variations of people's faces.

To generate random configurations of fixation regions we used the following approach: we start with the configuration selected by the teacher and then generate a random configuration by offsetting each fixation region by a random vector. The magnitude of each vector is bounded (so that the new position is likely to be within a face) and is also normalized by the face size. For each new face that is to be labeled, we start with the configuration that was already labeled by the teacher and then offset the location of each region by the vector previously assigned to that region. In this way the consistency of the configurations is mostly dependent on the human teacher.

By investigating the graphs in Figures 11-12 one can make the following observations: First, the

fixation regions that are small in size, in general, perform better compared to the large fixation regions. However, the size alone is not sufficient as illustrated in Figure-12, part 1. Ideally, the region should also be close to the center of the face since in that case the largest number of small RFs will cover the image and thus provide good estimates of the feature locations. Second, the system is very robust to changes over a large range of scale/location values. Regardless of the size and/or location of the fixation region, the system was able to learn the face category given a sufficient number of training examples. It is clear that the location/size of the fixation region plays a much more important role when the number of training images is small. Third, the structure of the fixation region itself is not very important e.g. similar results are obtained for uniform regions (part number 6) and for regions that contain high intensity gradients.

2.8 Biologically inspired hierarchical model for feature extraction and localization

Among the most important problems of computer vision are feature extraction and subsequent localization of those features in a new image. Since it is computationally prohibitive to search for the features over all possible locations and scales, it is necessary to design an algorithm that can selectively focus on and process information from only some regions within the image (Walther et al., 2005). Although there are numerous feature detection algorithms, such as the entropy-based feature detector (Kadir and Brady, 2001) and interest operators (C. Harris, 1988), matching is still computationally expensive given that the number of key points is usually over 10,000. (Lowe, 2000). In our model (Wu et al., 2006b), we use an approach that places the emphasis on the efficient matching procedure and utilizes top-down information. The model is inspired by human perception and the fact that when we search for an object in an image, our attention shifts from one region to another. In the process, we integrate information from different regions and the past explorations guide our future attempts until we finally localize an object.

The main problem that we addressed can be formulated as follows: given a collection of (object) features in one image (the reference image), find those feature in a new image where the new image can be a result of non-linear transformations applied to the reference image. In (Wu et al., 2006b) we presented a computationally efficient algorithm for solving this problem. We introduced a hierarchical representation that captures an object over different scales and is invariant to slight variations in feature locations. We demonstrated the robustness of the algorithm to non-linear image transformations (such as changes in scale, rotation, skew, addition of noise, and changes in brightness and contrast) as well as its computational efficiency on several real world images.

In our model (Wu et al., 2006b) we extract features using a collection of Gabor filters. The inspiration for using these features comes from the fact that simple cells in the visual cortex can be modeled by Gabor filters and the fact that they provide a sparse representation of images (Lee and Mangasarian, 2001). We construct Gabor filters using only phase zero and different phases are

crudely approximated by convolving the image with the kernel at all locations. In order to make a system tolerant to spatial distortions, we use a max operator where the pooling range is limited to the area covered by a receptive field. This range is much smaller than the range used in other approaches (Riesenhuber and Poggio, 1999) so that little information is lost about the locations of features. Similarly, we do not apply a max operator over different scales in order to minimize the loss of information.

The receptive fields (RFs) are arranged within M different layers. In order to capture features at different scales, the sizes of the RFs differ across the layers (and are the same with a given layer). We set the ratio of the size of a RF in one layer to the size of a RF in the layer of the nearest scale to $\sqrt{2}$. We use 19 feature detectors within each layer and call them elements. Each element is composed of 19 overlapping RFs.

The algorithm for localizing features in a new image consists of the following steps:
a) pre-select some interesting sub regions in a new image with saliency based algorithm;
b) choose the central point from each sub region and obtain the feature vectors of the largest layer;
c) find the nearest neighbor of the largest layer from the template;
d) calculate the relative scale of two images and new initial location;
e) localize the target from the new location with the smallest layer;

The results of the experiment designed to test the performance of the system for localizing object features are illustrated in Fig 13. We selected five features within the object in the reference image and then placed an object within a completely different environment and changed its location and scale with respect to the reference image. The algorithm was able to accurately localize the correct locations of all the features.

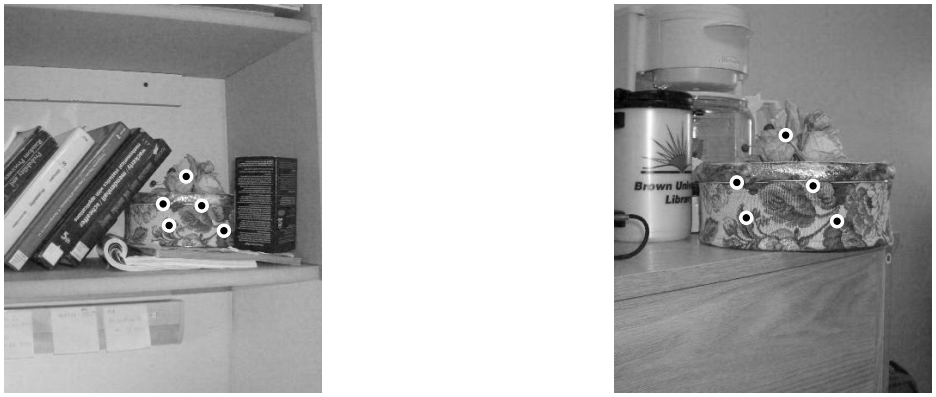

Figure 13: The 5 selected points in the left image are correctly identified in the right image.

3 Statistical pattern recognition-based classification algorithms

In this section we describe several new classification algorithms that use and further develop techniques from statistical pattern recognition and machine learning. Specifically, we will present a single sphere classification algorithm, a minimum bounding sphere algorithm for classifying object categories, adaptive distance nearest neighbor rule, a classification algorithms that can utilize information from unlabeled data, a model for data selection for SVMs, and a computational model for classifying both segmented and raw images

3.1 Pattern classification via single spheres

When objects are represented as d-dimensional vectors in some input space, classification amounts to partitioning the input space into different regions and assigning unseen objects in those regions into their corresponding classes. In the past, researchers have used a wide variety of shapes, including rectangles, spheres, and convex hulls, to partition the input space.

Spherical classifiers were first introduced into pattern classification by Cooper in 1962 and subsequently studied by many other researchers (Cooper, 1962; Batchelor, 1974). One well known classification algorithm consisting of spheres is the Restricted Coulomb Energy (RCE) network. The RCE network, first proposed by Reilly, Cooper, and Elbaum, is a supervised learning algorithm that learns pattern categories by representing each class as a set of prototype regions - usually spheres (Reilly et al., 1982; Scofield et al., 1987). The RCE network incrementally creates spheres around training examples that are not covered, and it adaptively adjusts the sizes of spheres so that they do not contain training examples from different classes. After the training process, only the set of class-specific spheres is retained and a new pattern is classified based on which sphere it falls into and the class affiliation of that sphere.

Another learning algorithm that is also based on spherical classifiers is the set covering machine (SCM) proposed by Marchand and Shawe-Taylor (Marchand and Shawe-Taylor, 2002). In their approach, the final classifier is a conjunction or disjunction of a set of spherical classifiers, where every spherical classifier dichotomizes the whole input space into two different classes with a sphere. The set covering machine aims to find a conjunction or disjunction of a minimum number of spherical classifiers such that it classifies the training examples perfectly.

Regardless of whether the influence of a sphere is local (as in the RCE network) or global (as in the SCM), classification algorithms that use spheres normally need a number of spheres in order to achieve good classification performance, and therefore have to deal with difficult theoretical and practical issues such as how many spheres are needed and how to determine the centers and radii of the spheres.

In our work presented in (Wang et al., 2005b) we explored the possibility of using single spheres for pattern classification. Inspired by the support vector machines and the support vector data

description method, we presented an algorithm that constructs single spheres in the kernel feature space that separate data with the maximum separation ratio. By incorporating the class information of the training data, our approach provides a natural extension to the SVDD method of Tax and Duin, which computes minimal bounding spheres for data description (also called One-class classification).

By adopting the kernel trick, the new algorithm effectively constructs spherical boundaries in the feature space induced by the kernel. As a consequence, the resulting classifier can separate patterns that would otherwise be inseparable when using a single sphere in the input space. In addition, by adjusting the ratio of the radius of the separating sphere to the separation margin, a series of solutions ranging from spherical to linear decision boundaries can be obtained. Specifically, when the ratio is set to be small, a sphere is constructed that gives a compact description of the positive examples, coinciding with the result of the SVDD method; when the ratio is set to be large, the solution effectively coincides with the maximum margin hyperplane solution. Therefore, our method effectively encompasses both the support vector machines for classification and the SVDD method for data description. This feature of the proposed algorithm may also be useful for dealing with the class-imbalance problem. We tested the new algorithm and compared it to the support vector machines using both artificial and real-world datasets. The experimental results show that the new algorithm offers comparable performance on all the datasets tested. Therefore, our algorithm provides an alternative to the maximum margin hyperplane classifier.

3.2 Training data selection for support vector machines

Support vector machines (SVMs), introduced by Vapnik and coworkers in the structural risk minimization (SRM) framework (Boser et al., 1992; Cortes and Vapnik, 1995; Vapnik, 1998), have gained wide acceptance due to their solid statistical foundation and good generalization performance that has been demonstrated in a wide range of applications.

Training a SVM involves solving a constrained quadratic programming (QP) problem, which requires large memory and takes enormous amounts of training time for large-scale applications (Joachims, 1999). On the other hand, the SVM decision function depends only on a small subset of the training data, called support vectors. Therefore, if one knows in advance which patterns correspond to the support vectors, the same solution can be obtained by solving a much smaller QP problem that involves only the support vectors. The problem is then how to select training examples that are likely to be support vectors. Recently, there has been considerable research on data selection for SVM training. For example, Shin and Cho proposed a method that selects patterns near the decision boundary based on the neighborhood properties (Shin and Cho, 2003). In (Zheng et al., 2003), k-means clustering is employed to select patterns from the training set. In (Zhang and King, 2002), Zhang and King proposed a β-skeleton algorithm to identify support vectors. In (Abe and Inoue, 2001), Abe and Inoue used Mahalanobis distance to estimate boundary points. In the reduced

SVM (RSVM) setting, Lee and Mangasarian chose a subset of training examples using random sampling (Lee and Mangasarian, 2001). In (Huang and Lee, 2004), it was shown that uniform random sampling is the optimal robust selection scheme in terms of several statistical criteria.

In our work presented in (Wang et al., 2007c) we introduced two new data selection methods for SVM training. The first method selects training data based on a statistical confidence measure introduced in (Wang et al., 2003). The second method uses the minimal distance from a training example to the training examples of a different class as a criterion to select patterns near the decision boundary. This method is motivated by the geometrical interpretation of SVMs based on the (reduced) convex hulls. To analyze their effectiveness in terms of their ability to reduce the training data while maintaining the generalization performance of the resulting SVM classifiers, we conducted a comparative study using several real-world datasets. More specifically, we compared the results obtained by these two new methods with the results of the simple random sampling scheme and the results obtained by the selection method based on the desired SVM outputs. Through our experiments, several important observations have been made: (1) In many applications, significant data reduction can be achieved without degrading the performance of the SVM classifiers. For that purpose, the performance of the confidence measure-based selection method is often comparable to or better than the performance of the method based on the desired SVM outputs. (2) When the reduction rate is high, some of training examples that are 'extremely' close to the decision boundary have to be removed in order to maintain the generalization performance of the resulting SVM classifiers. (3) In spite of its simplicity, random sampling performs consistently well, especially when the reduction rate is high. However, at low reduction rates, random sampling performs noticeably worse compared to the confidence measure-based method. (4) When conducting training data selection, sampling training data from each class separately according to the class distribution often improves the performance of the resulting SVM classifiers.

By directly comparing various data selection schemes with the scheme based on the desired SVM outputs, we are able to conclude that the confidence measure provides a criterion for training data selection that is almost as good as the optimal criterion based on the desired SVM outputs. At high reduction rates, by removing training data that are likely to be outliers, we boost the performance of the resulting SVM classifiers. Random sampling performs consistently well in our experiments, which is consistent with the results obtained by Syed et al. in (Syed et al., 1999) and the theoretical analysis of Huang and Lee in (Huang and Lee, 2004). The robustness of random sampling at high reduction rates suggests that, although an SVM classifier is fully determined by the support vectors, the generalization performance of an SVM is less reliant on the choice of training data than it appears to be.

3.3 Pattern classification based on minimum bounding spheres

Given a set of training data, the minimum bounding sphere is defined as the smallest sphere that encloses all the data. Similarly, the minimum bounding sphere of each class is the smallest sphere enclosing all the training data from the corresponding class. The minimum bounding sphere can be computed by solving a quadratic programming (QP) problem. In (Schölkopf et al., 1995), Schölkopf, Burges, and Vapnik computed the radius of the minimum bounding sphere of all training data to estimate the VC-dimension of a SVM classifier. In (Tax and Duin, 1999), Tax and Duin applied the minimum bounding sphere to data domain description (also called one-class classification).

Given the fact that a minimum bounding sphere of each class is constructed without considering the distribution of training examples of other classes, it is not immediately clear whether or not an effective classifier can be built based on these class-specific minimum bounding spheres. However, from a computational point of view, there is a great advantage to use the minimum bounding spheres for pattern classification purposes. Most noticeably, a classifier based on the class-specific minimum bounding spheres can deal with multi-class problems easily and efficiently. This is because one needs to compute the minimum bounding sphere of each class only once, which is in direct contrast with the support vector machines (SVMs) trained with the one-against-all or one-against-one methods, where the optimal separating hyperplanes have to be computed many times. In addition, because the minimum bounding sphere for each class can be computed separately, independent of training examples of other classes, the size of the resulting quadratic programming problem is therefore smaller than that of the support vector machine algorithm (Boser et al., 1992; Cortes and Vapnik, 1995). To explore this possibility, Zhu et al. proposed a multi-class classification algorithm that uses the minimum bounding spheres to classify a new example and showed that the resulting classifier performs comparable to the standard SVMs.

In our work presented in (Wang et al., 2005a), we conducted a comparative study using both artificial and real-world datasets and showed that the decision rule proposed by Zhu et al. is generally insufficient for achieving the state-of-the-art classification performance. Motivated by the Bayes decision theory, we proposed a new decision rule for making classification decisions based on the constructed minimum bounding spheres. Experimental results demonstrate that the new decision rule significantly improves the performance of the resulting minimum sphere-based classifier. Furthermore, on most of the datasets being tested, the sphere-based classifier achieves comparable results as the standard SVMs. Since the minimum bounding sphere-based classifier is computationally more efficient to train and it deals with the multi-class problem more easily than the SVMs, the proposed method provides an alternative approach to large-scale multi-class classification problems.

3.4 Improving nearest neighbor rule with a simple adaptive distance measure

The nearest neighbor (NN) rule, first proposed by (Fix and Hodges, 1951), is one of the oldest and simplest pattern classification algorithms. Given a set of n labeled examples $D_n = \{(\vec{X}_1, Y_1), \ldots, (\vec{X}_n, Y_n)\}$ with input vectors $\vec{X}_i \in R^d$ and class labels $Y_i \in \{\omega_1, \ldots, \omega_M\}$, the NN rule classifies an unseen pattern $\vec{X}$ to the class of its nearest neighbor in the training data D_n. To identify the nearest neighbor of a query pattern, a distance function has to be defined to measure the similarity between two patterns. In the absence of prior knowledge, the Euclidean and Manhattan distance functions have conventionally been used as similarity measures for computational convenience.

The basic rationale for the NN rule is both simple and intuitive: patterns close in the input space R^d are likely to belong to the same class. This intuition can be justified more rigorously in a probabilistic framework in the large sample limit. Indeed, as one can easily show, as the number of training examples $n \to \infty$, the nearest neighbor of a query pattern converges to the query pattern with probability one, independently of the metric used. Therefore, the nearest neighbor and the query pattern have the same *a posteriori* probability distribution asymptotically, which leads to the asymptotic optimality of the NN rule:

$$L^* \leq L_{\mathrm{NN}} \leq L^*(2 - \frac{M}{M-1}L^*) \ , \tag{9}$$

where L^* is the optimal Bayes probability of error, see (Cover and Hart, 1967). According to (9), the NN rule is asymptotically optimal when $L^* = 0$, i.e., when different pattern classes do not overlap in the input space. When the classes do overlap, the sub-optimality of the NN rule can be overcome by the k-nearest neighbor (k-NN) rule that classifies $\vec{X}$ to the class that appears most frequently among its k nearest neighbors (Stone, 1977).

It should be noted that the above results are established in the asymptotic limit and essentially rely on averaging over an infinite amount of training examples within an infinitesimal neighborhood to achieve optimality. In reality, one most often only has access to a finite number of training examples, and the performance of the k-NN rule depends crucially on how to choose a suitable metric so that according to the chosen metric the majority of the k nearest neighbors to a query pattern is from the desired class. In the past, many methods have been developed to locally adapt the metric so that a neighborhood of approximately constant *a posteriori* probability can be produced. Examples of these methods include the flexible metric method by (Friedman, 1994), the discriminant adaptive method developed by (Hastie and Tibshirani, 1996), and the adaptive metric method by (Domeniconi et al., 2002). The common idea underlying these methods is that they estimate feature relevance locally at each query pattern. The locally estimated feature relevance leads to a weighted metric for computing the distance between a query pattern and the training data. As a result, neighborhoods get constricted along the most relevant dimensions and elongated along the less important ones. Although these methods improve the original k-NN rule due to their

capability to produce local neighborhoods in which the *a posteriori* probabilities are approximately constant, the computational complexity of such improvements is high. More recently, there has been considerable research interest in directly learning distance metrics from training examples to improve the k-NN rule. For example, (Goldberger et al., 2004) proposed a method for learning a Mahalanobis distance measure by directly maximizing a stochastic variant of the leave-one-out k-NN score on the training data. (Weinberger et al., 2005) developed a method for learning a Mahalanobis distance metric by semidefinite programming. Many other methods along this line can be found in the references therein.

3.4.1 Adaptive nearest neighbor rule

We briefly describe the k-NN rule to introduce notation. Let us assume that patterns to be classified are represented as vectors in a d-dimensional Euclidean space R^d. Given a set of training examples $\{(\vec{X}_1, Y_1), \ldots, (\vec{X}_n, Y_n)\}$ and a query pattern $\vec{X}$, the k-NN rule first finds the k nearest neighbors of $\vec{X}$, denoted by $\vec{X}_{(1)}, \ldots, \vec{X}_{(k)}$, and assigns $\vec{X}$ to the majority class among $Y_{(1)}, \ldots, Y_{(k)}$, where $Y_{(i)}$ are the corresponding class labels of $\vec{X}_{(i)}$. Without prior knowledge, the Euclidean distance (L2)

$$d(\vec{X}, \vec{X}_i) = (\sum_{j=1}^{d} |X^j - X_i^j|^2)^{1/2} \tag{10}$$

and the Manhattan distance (L1)

$$d(\vec{X}, \vec{X}_i) = \sum_{j=1}^{d} |X^j - X_i^j| \tag{11}$$

have conventionally been used for measuring the similarity between $\vec{X}$ and $\vec{X}_i$. For a binary classification problem in which $Y \in \{-1, 1\}$, the k-NN rule amounts to the following decision rule:

$$f(\vec{X}) = \mathrm{sgn}(\sum_{\mathrm{i}=1}^{\mathrm{k}} \mathrm{Y}_{(\mathrm{i})}) \ . \tag{12}$$

To define the locally adaptive distance between a query pattern $\vec{X}$ and a training example $\vec{X}_i$, we first construct the largest sphere centered on $\vec{X}_i$ that excludes all training examples from other classes. This can be easily achieved by setting the radius of the sphere to

$$r_i = \min_{l: Y_l \neq Y_i} d(\vec{X}_i, \vec{X}_l) - \epsilon \ , \tag{13}$$

where $\epsilon > 0$ is an arbitrarily small number. Notice that depending on the metric $d(\vec{X}_i, \vec{X}_l)$ that is actually used, the regions defined by points with distance to $\vec{X}_i$ less than r_i may not be a sphere. However, for simplicity, we refer to such defined regions as spheres for convenience when no

confusion arises. The locally adaptive distance between $\vec{X}$ and the training example $\vec{X}_i$ is defined as

$$d_{\text{new}}(\vec{X}, \vec{X}_i) = \frac{d(\vec{X}, \vec{X}_i)}{r_i} \ . \tag{14}$$

Several important points are immediately clear from the above definition. First, although the above distance measure (14) is only defined between a query pattern $\vec{X}$ and existing training examples $\vec{X}_i$, the definition can be easily extended to measure the similarity between $\vec{X}$ and an arbitrary point $\vec{X}'$ by first defining a radius r' associated with $\vec{X}'$ similarly to (13). Secondly, by definition, the distance function (14) is not symmetric. For example,

$$d_{\text{new}}(\vec{X}_i, \vec{X}_j) \neq d_{\text{new}}(\vec{X}_j, \vec{X}_i) \tag{15}$$

if the radii r_i and r_j associated with $\vec{X}_i$ and $\vec{X}_j$ respectively are not the same. Therefore, the new distance measure is generally not a metric. Finally, according to the new distance measure, the smallest distance between a training example and training examples of other classes is one, and training examples with their distances less than one to a training example all have the same class label.

After adopting the new distance measure (14), the adaptive nearest neighbor rule works exactly the same as the original nearest neighbor rule except that we use the adaptive distance measure to replace the original L2 or L1 distance measure for identifying the nearest neighbors. Formally, given a query pattern $\vec{X}$ for a binary classification problem, the adaptive nearest neighbor rule first identifies its k nearest neighbors, denoted again by $\vec{X}_{(1)}, \ldots, \vec{X}_{(k)}$, according to the new distance measure $d(\vec{X}, \vec{X}_i)/r_i$ for $i = 1, \ldots, n$, and classifies $\vec{X}$ to the class

$$f(\vec{X}) = \text{sgn}(\sum_{\text{i=1}}^{\text{k}} \text{Y}_{\text{(i)}}) \ . \tag{16}$$

3.4.2 Experimental results

We tested our Adaptive k-NN model on several real-world benchmark datasets from the UCI Machine Learning Repository [1]. Throughout our experiments, we used 10-fold cross validation to estimate the generalization error. Table 3 shows the error rates and the standard deviations of the 1-NN rule using the Euclidean metric (1-NN), the 1-NN rule using the adaptive distance measure (A-1-NN), the k-NN rule using the Euclidean metric (k-NN), Support Vector Machines (SVMs) with Gaussian kernels, and the k-NN rule using the adaptive distance measure (A-k-NN).

It is easy to see from the first two columns that the A-1-NN rule outperforms the 1-NN rule on all the 5 datasets. On most datasets, the improvements of the A-1-NN rule over the 1-NN rule is statistically significant. Because the only difference between these two rules is the distance functions used to identify the nearest neighbors, these results show that the nearest neighbor

[1] http://www.ics.uci.edu/∼mlearn/MLRepository.html

Table 3: Comparison of classification results

Dataset	1-NN	A-1-NN	k-NN	SVM	A-k-NN
Breast Cancer	04.85 (0.91)	03.09 (0.71)	02.79 (0.67)	03.68 (0.66)	02.79 (0.74)
Ionosphere	12.86 (1.96)	06.86 (1.36)	12.86 (1.96)	04.86 (1.05)	04.86 (1.28)
Pima	31.84 (1.05)	28.16 (1.57)	24.61 (1.36)	27.50 (1.68)	25.13 (1.46)
Liver	37.65 (2.80)	32.94 (2.23)	30.88 (3.32)	31.47 (2.63)	30.88 (1.77)
Sonar	17.00 (2.26)	13.00 (1.70)	17.00 (2.26)	11.00 (2.33)	13.00 (1.70)

identified according to the adaptive distance measure is more likely to have the same class label as the query pattern than the nearest neighbor identified according to the Euclidean distance. Similar results also hold for k nearest neighbors with k greater than 1. In the last three columns of Table 3, we report the lowest error rates of the k-NN rule and the A-k-NN rule and compare them to the lowest error rates of the SVMs with Gaussian kernels. On each dataset, we run the k-NN rule using both distance measures at various values of k from 1 to 50 and picked the lowest error rate. As we can see from Table 3, the k-NN rule with the adaptive distance measure performs significantly better than the k-NN rule with the Euclidean distance measure on the Breast Cancer and Sonar datasets, making the A-k-NN rule overall comparable to the state-of-the-art SVMs.

3.5 Bayesian learning from unlabeled data

Designing a classification system for the problem where the number of training examples is very small and the dimensionality of the data very large is extremely challenging. Over the last few years, this problem has been addressed by both the computer vision and machine learning researches and several promising models have been proposed. A frequently used approach is to reduce the dimensionality of the features space, using some of the unsupervised algorithms such as the PCA, or to extract few very informative features for accurately representing the object class (Li et al., 2006; Levi and Weiss, 2004). Unfortunately, automatically selecting such features is a very difficult task for which an optimal solution still does not exist. Another possibility is to artificially enlarge the small training set by adding copies of the training data to the original set (Wolf and Martin, 2005).

Within the machine learning community, a number of semi-supervised learning algorithms have been introduced aiming to improve the performance of classifiers by using large amounts of unlabeled samples together with the labeled ones (Zhu, 2005). Some popular semi-supervised methods within the generative classification framework include co-training (Blum and Mitchell, 1998; Goldman and Zhou, 2000) and expectation maximization (EM) mixture models (Nigam and Ghani, 2000; Baluja, 1998). In co-training, one selects samples that have a high confidence of belonging to a given class from a large reservoir of samples to improve the accuracy of a classifier. In some

situations, although the number of training examples is small, the number of unlabeled data can be quite large and the semi-supervised algorithms can be used. However, in many other cases, such as the medical diagnosing that involves a biopsy, the number of training and testing examples can be very small. Another example is the fMRI analysis where in addition to the small sample size (often smaller than 20) the dimensionality of the feature space is extremely large (e.g. the number of voxels can be over 100,000). In these situations, the algorithm that can learn from few examples in high dimensional space and successfully classify an unlabeled example can be of great importance.

The topic of discriminant analysis using Bayesian estimation has been addressed by many researchers and dates back to early work of Geisser (Geisser, 1964) and Keehn (Keehn, 1965) in which the authors use non-informative and Wishart priors respectively. Recently, a distribution-based Bayesian discriminant (DBBD) analysis has been proposed (Srivastava and Gupta, 2006) in which the uncertainty is considered to be over the set of Gaussian pdfs and the Bayesian estimation is done over the domain of the Gaussian pdfs. The authors demonstrated that the DBBD algorithm improves classification in the high dimensional space where the estimation of the covariance matrix is very difficult with limited number of training samples. However, in some situations the features are assumed to be independent (like in the Naive Bayes approach used in this paper) and in those cases the DBBD approach does not improve classification.

We have recently introduced a self-improving model (Wu and Neskovic, 2007) that utilizes information from an unlabeled sample to improve the classification rate of the sample itself. The idea is to include the unlabeled sample in each labeled training set and use it to modify the parameters of the class-conditional density function (CCDF) for a Bayes classifier. Although there is no obvious reason why the inclusion of a sample should improve classification (since it is added to training data of all the classes), we show that when the number of training samples is small this procedure significantly improves classification rates. From our analysis of two normal distributions, we conclude that the gains are the consequence of moving one decision boundary toward its optimal location.

We tested the self-improved procedure on both the artificial and real-world data and demonstrated that it achieves classification rates that are significantly higher compared to the performance of the classifiers that did not include the unlabeled example. Furthermore, we showed that the procedure works extremely well when the dimensionality of the data is high, which is in contrast to most current classifiers (Wu and Neskovic, 2007).

3.5.1 Experimental results

In the following, we present the results of our classification procedure when tested on artificial and real data sets using normal distributions. We start from a one-dimensional normal distribution and show that the modified rule improves the classification accuracy and that the improvement is especially large when the number of training samples is small. Furthermore, when one class has

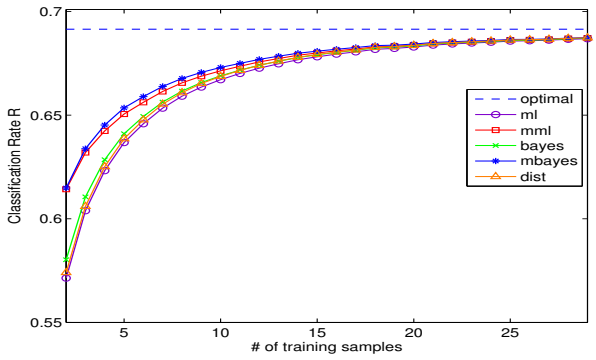

Figure 14: Classification rates with different estimation approaches.

a dominating prior over other classes, we show that improvement can be very large. Generalizing this model to the high-dimensional case, we show that small boosts from each component produce combined improvements that far exceed the rates obtained with original classifiers C_{bayes} and C_{ml}. We also compare our algorithm to distribution-based Bayesian discriminant analysis (Srivastava and Gupta, 2006) assuming the same noninformative priors.

In all the experiments that involve one-dimensional distributions, we generated examples through the numerical integration with 5,000,000 random samples. We sample from distributions $p(\vec{x}|w)$ and $p(\mathcal{D})$ for which we assume a Gaussian form.

1D Gaussians, same priors. In the first experiment, we consider two classes and assume that the data are generated from Gaussian distributions with parameters: $\{\mu_1, \sigma_1^2\} = \{0, 1\}$ and $\{\mu_2, \sigma_2^2\} = \{1, 1\}$.

In Fig. 14 we show the classification rates with different methods of estimation as a function of the number of training samples. Modified approaches improve the classification rate in both Bayes and ML cases. In this example, the modified ML approach outperforms the ML approach and the Bayes estimation with uninformative prior, and the best results are obtained with the modified Bayes estimation.

1D Gaussians, different priors. In the second experiment, we test the importance of the priors on the classification performance. We choose the prior for class 1 to be $p(y = 1) = 0.9$. The parameters for Fig. 15(a) are $\{\mu_1, \sigma_1^2\} = \{0, 1\}$, $\{\mu_2, \sigma_2^2\} = \{1, 1\}$, and the parameters for Fig. 15(b) are $\{\mu_1, \sigma_1^2\} = \{0, 1\}$, $\{\mu_2, \sigma_2^2\} = \{2, 4\}$. Compared to the results in Fig. 14 where $p(y = 1)$ is $1/2$, these results are significantly better. The modified ML and modified Bayes approaches almost reach the Bayes limit with only 2 or 3 training samples, while more than 20 samples are required to do that for the regular ML and Bayes approaches. The reason for the striking difference is likely the consequence of the fact that the decision boundary goes through the tails of the normal distributions when one class has a dominating prior. It was also suggested that one should avoid the densest region with many other semi-supervised classifiers (Zhu, 2005).

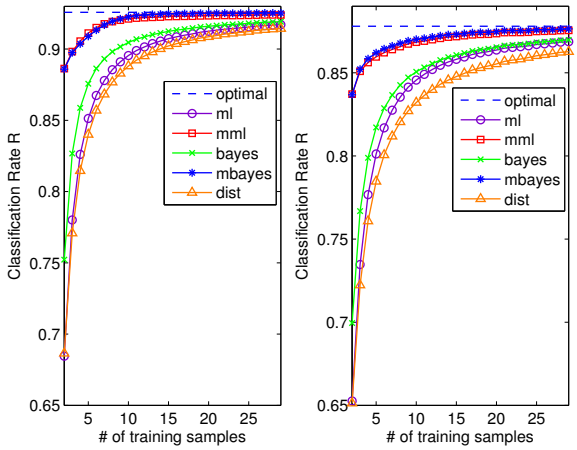

Figure 15: Two examples of classification rates when the prior of one class dominates the other class.

High-dimensional space. One possible application of the proposed estimators is to improve the accuracy of a Gaussian Naive Bayes (GNB) classifier, especially in situations when the dimensionality of the feature space is high. When one does not have many training samples but has many good features, the boost from each feature can significanlty improve the accuracy.

Suppose, for instance, that we have 2 classes, and each component of these classes' feature vector are normally distributed according to $\mathcal{N}(\prime, \infty)$ and $\mathcal{N}(\cdot, \infty)$ for class 1 and class 2, respectively. When Δ is small, one can get a classification rate only slightly above the chance using one component. However, one can expect to get reasonably high classification rate when one has many such independent features and ideally 100% with infinite features. To achieve this, one can train one Bayes classifier using one component and use a voting from all classifiers. Assume that classification R_i from one component has the probability $p > 1/2$ of being correct, and denote with d the number of features. Then there exists a positive real number ϵ, such that, $p - \epsilon > 1/2$. From the weak law of large numbers,

$$R = \Pr(\frac{1}{d}\sum_{i=1}^{d} R_i > 1/2) > \Pr(\frac{1}{d}\sum_{i=1}^{d} R_i > p - \epsilon) >$$

$$\Pr(|\frac{1}{d}\sum_{i=1}^{d} R_i - p| < \epsilon) \overset{d\to\infty}{\to} 1.$$

Although we do not use the voting procedure but stick to the GNB classifier similar behavior should be expected when d tends to infinity.

Fig. 3 shows two examples when $\Delta = 0.5$. The prior for Fig. 3(a) is $p(y = 1) = 0.5$ while the prior for Fig. 3(b) is $p(y = 1) = 0.1$. There is little difference in performance between the two approaches

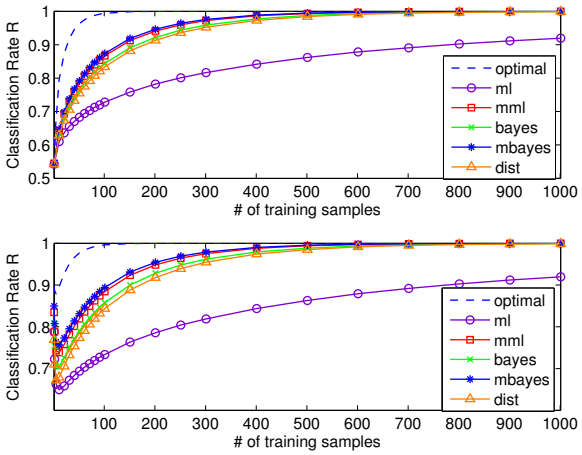

Figure 16: Two examples of classification rates for high-dimensional normal distributions.

when the number of features is at two extremes, very small or very large. In a very large region in the middle, the difference can be big when one has few training samples. In our case when the number of training examples is $n = 5$, the modified estimators outperform traditional estimators significantly (sometimes more than 20%). If one uses many training samples, e.g. $n = 20$, for each class, the difference is small. This is expected since the inclusion of one unlabeled sample will not influence much the mean and variance for each component when one has many training samples.

Real-world examples. The following results are obtained on several real-world benchmark datasets from the UCI machine learning repository. We choose only datasets whose attributes are continuous real numbers so that we can apply a GNB classifier. Through our test, we use equivalent priors for all classes. We choose a pool of random samples from each dataset, and use this pool for training and the rest for testing. We repeat this for 1000 pools. For the Spambase dataset that has too many samples, we only test on the first 100 samples for each class. The average classification rates are shown in Fig. 17 for the modified ML and ML approaches. For almost each dataset, the modified ML approach outperforms the ML approach in a similar way as we can observe with the normally-distributed artificial examples. The difference in performance becomes smaller as the number of training samples becomes larger. However, in the case of the Spambase, there is still a 15% difference in the performance when we use 20 samples from each class for training. This is because the attributes of Spambase are more likely to be exponentially distributed than normally distributed. This result illustrates that our model can work well even when the suggested model does not agree with the underlying distribution, which might be a problem with many other semi-supervised algorithms (Cozman and Cohen, 2002).

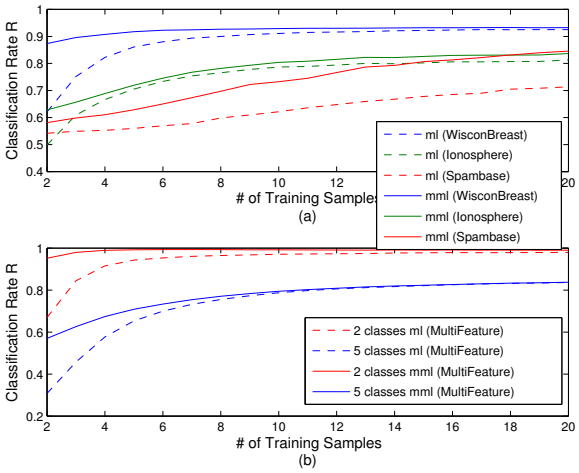

Figure 17: Classification rates obtained from real datasets.

3.6 Classifying raw and segmented images

Classification and segmentation are among the most commonly used methods for analyzing images. They are especially used in medical applications, for analyzing and decoding fMRI images (Mitchell et al., 2004), and for understanding the functional properties of the human brain (Pessoa and Padmala, 2006). However, these two methods are usually used independently despite the fact that they could potentially benefit each other. From the machine learning point of view, among the most difficult challenges are: dealing with extremely noisy data, and learning in high dimensional feature space. To address the first problem, researchers often use "blocked" data in an attempt to average out the noise. For example, Cox and Savoy (Cox and Savoy, 2003) employed 20s blocks containing 10 stimulus repetitions while Haynes and Rees (Haynes and Rees, 2005) used 30s blocks. To deal with the second problem, a common approach is to employ various feature selection techniques (Guyon et al., 2002), to reduce the number of voxels from over 10^4 to several hundred or less (Mitchell et al., 2004; Pessoa and Padmala, 2006).

One way to alleviate the previous problems is to use segmentation (clustering) in combination with a classification algorithm. The number of clusters is in general much smaller than the number of voxels, and including spatial constraints in the clustering algorithm can help with noise. However, classifying segmented images presents its own challenges: how should one evaluate similarity between two different segmentations? Which cluster from one segmentation corresponds to which cluster from another segmentation, and how should one compare the shapes of different regions?

Probably the simplest solution is to consider only two cluster labels, e.g. a background label and a region of interest (ROI) label, and then focus only on the ROIs and compare their shapes across different images (Pokrajac et al., 2005). Although this approach can be useful in some clinical

applications (Pokrajac et al., 2005), the analysis is limited to only ROIs and its associated binary assignment (active versus non-active) is not sufficient. For example, in many situations the same region of the brain might be involved in several functional activities at the same time and therefore the same voxel should have multiple labels (belong to several clusters) and this assignment should be probabilistic.

In our recent work (Wu et al., 2007), we proposed a non-parametric model that can classify not only raw images but also segmented images. We also introduced a clustering algorithm which can incorporate spatial information (to alleviate noise) and represent the assignment of voxels to clusters using probabilistic representation. Instead of focusing only on specific ROIs, our model can classify fMRI images of the whole brain using a single trial. We demonstrated our results on a challenging fMRI dataset using single trials in which a stimulus is only 70ms long. The proposed classifier is voxel-based (uses cluster assignment information from each voxel) as opposed to region-based (e.g. representing clusters with density functions) and it can therefore easily deal with any kind of region boundaries (e.g. sharp, fuzzy or irregular). The algorithm is very general in that it can utilize both deterministic and probabilistic voxel to clusters assignments, and it can also deal with clusters with multiple labels.

To segment images, we implemented a deterministic k-means algorithm and a probabilistic Hidden Markov Random Field (HMRF) finite mixture model (Zhang et al., 2001). The advantage of the HMRF model is that it imposes spatial constraints on the neighboring voxels which is biologically realistic assumption since neighboring voxels tend to have similar activations. In this work, we build a HMRF Dirichlet process mixture model and derive a collapsed Variational Bayesian (VB) approach (Welling et al., 2007) to integrate out the mixture weights.

We tested our model on real fMRI images and demonstrate that our classifier significantly outperforms the parametric GNB and the Maximum Likelihood (ML) k-means classifiers. Furthermore, we showed that higher classification rates are obtained when the images are segmented using a probabilistic HMRF approach compared to deterministic k-means method.

3.6.1 The model

In this section, we provide a description of our model, a non-parametric classifier. We assume that we are given an N-dimensional observation vector $\vec{z}$ that belongs to one of Y classes and the task is to find a class, y, for which $p(y|\vec{z})$ is maximal. We take a Bayesian approach and further assume that the likelihood is a non-parametric probability mass function. We discretize the observation vector and divide the values of each vector element, z_i, into K bins. The probability that the value of the i^{th} element falls into the k^{th} bin we denote as η_{ik}. This probability can be written in a compact form by introducing the indicator matrix U as $p(u_{ik} = 1) = \eta_{ik}$. Each element of the indicator matrix, $u_{ik} = 1, i = 1, ..., N, k = 1, ..., K$, takes only two values, 0 and 1, and provides an assignment for the value of the i^{th} element of the observation vector to the k^{th} bin. Note that

$\sum_{k=1}^{K} u_{ik} = 1, i = 1, ..., N$ since each element has one value and can only choose from 1 to K. Using this notation, calculating the likelihood $p(\vec{z}|\eta^y)$ is equivalent to calculating $p(U|\eta^y)$, where η^y denotes a *class specific* parameter vector. To simplify notation, in the following we will derive the expression for one class only and therefore will omit the subscript y. Assuming that all the elements of the observation vector are independent, the likelihood function is

$$p(U|\vec{\eta}) = \prod_{i}^{N} \prod_{k}^{K} \eta_{ik}^{u_{ik}}. \tag{17}$$

In principle, the parameter $\vec{\eta}$ can be estimated from the training data and then used to calculate the class conditional likelihood function (CCLF), Eq (17). However, the likelihood can also be calculated without explicitly estimating the parameter $\vec{\eta}$. In this work we use Bayesian approach and integrate over the parameter. If we denote the training examples of a given class y as $V^j = \{(\vec{v}_1, ..., \vec{v}_N)^j\}$, then the class conditional likelihood can be written as $p(U|V^1, ..., V^m)$, where m is the number of training examples. More specifically, the likelihood function $p(U|V^1, ..., V^m)$ is given by,

$$p(U|V^1, ..., V^m) = \int d\vec{\eta} p(U|\vec{\eta}) p(\vec{\eta}|V_1, ..., V_m) \tag{18}$$

The unknown parameters $\vec{\eta}_i$ of the ith element can be estimated from the m training samples,

$$p(\vec{\eta}_i|\vec{v}_i^1, ..., \vec{v}_i^m) = \frac{p(\vec{v}_i^1, ..., \vec{v}_i^m|\vec{\eta}_i) p(\vec{\eta}_i)}{\int d\vec{\eta}_i p(\vec{v}_i^1, ..., \vec{v}_i^m|\vec{\eta}_i) p(\vec{\eta}_i)} \tag{19}$$

where $p(\vec{v}_i^1, ..., \vec{v}_i^m|\vec{\eta}_i) = \prod_{j=1}^{m} p(\vec{v}_i^j|\vec{\eta}_i)$. We choose for the prior $p(\vec{\eta}_i)$ a Dirichlet distribution

$$p(\vec{\eta}_i) = \frac{\Gamma(K\lambda)}{\Gamma(\lambda)^K} \prod_{k}^{K} \eta_{ik}^{\lambda-1} \tag{20}$$

and therefore our posterior will also have the form of Dirichlet distribution which will allow the exact calculation of the integral. Note that in calculating the integral over $\vec{\eta}_i$, one has to include the constraint that $\eta_{ik} \geq 0, \sum_k \eta_{ik} = 1$. Knowing that,

$$\int d\vec{\eta}_i \prod_{k=1}^{K} \eta_{ik}^{\lambda_k - 1} = \frac{\prod_{k=1}^{K} \Gamma(\lambda_k)}{\Gamma(\sum_{k=1}^{K} \lambda_k)} \tag{21}$$

Eq (18) can be integrated as,

$$p(U|V^1, ..., V^m) = \prod_{i=1}^{N} \frac{\prod_{k=1}^{K} \Gamma(s_{ik} + u_{ik} + \lambda) \Gamma(m + K\lambda)}{\prod_{k=1}^{K} \Gamma(s_{ik} + \lambda) \Gamma(m + K\lambda + 1)}$$

where $s_{ik} = \sum_{j=1}^{m} v_{ik}^j$. s_{ik} represents the number of times the i-th component equals k across the training samples.

3.6.2 Classifying segmented fMRI images

Our model can be easily extended to the case when the input image is not a raw image (e.g. the one represented with gray-scale or voxel activation values) but a segmented image. We now describe the situation when the input to the classifier is a segmented fMRI image. This image can be obtained using a number of clustering algorithms such as the k-means or the HMRF algorithm. Each voxel of the segmented image is associated with a sequence of numbers that provide an assignment of the voxel to each of the K clusters. We call this sequence of numbers a clustering distribution vector (CDV). The assignment can be either fixed, in which case a voxel is assigned to only one cluster, or probabilistic, in which case the sequence represents probabilities of assigning a given voxel to each of the K clusters. For example, if the CDV is obtained after maximizing a posterior or from other deterministic algorithms such as the expectation-maximization algorithm, then the assignment matrix v_{ik} is the indicator matrix and for some $k^*, v_{ik^*} = 1$. However, if the CDV is obtained using a probabilistic clustering approach, such as the HMRF finite mixture model, the assignment for each voxel is $\vec{v}_i = \{p(c_i = 1), ..., p(c_i = K)\}, \sum_k v_{ik} = 1$.

We will assume that there exists a true underlying distribution that assigns a voxel to each of the K clusters and that this distribution is class specific. If as a result of the deterministic clustering procedure the i-th voxel is assigned to k^*th cluster, $v_{ik^*} = 1$, then the distribution of $\vec{v}$ is given as $p(\vec{v}_i|\vec{\eta}_i) = p(v_{ik^*} = 1|\vec{\eta}_i) = \eta_{ik^*}$, which can be viewed as a generalization of the Bernoulli distribution to more than two outcomes (or the categorical distribution). When the cluster assignment consist of probabilities instead integers, we generalize the distribution as $p(\vec{v}_i|\vec{\eta}_i) \sim \prod_{k=1}^{K} \eta_{ik}^{v_{ik}}$, which reduces to η_{ik^*} in a deterministic case $v_{ik^*} = 1$.

3.6.3 The correspondence problem

The result of a clustering algorithm is a segmented image where all the voxels from one region (or cluster) have the same label. However, the labeling of each region is essentially random and therefore even the clusters representing two exact segmentations can have different labels. We assume that all the images from one class will have similar segmentations and we want to find an assignment between clusters of any two images that reflects this property. The problem of which cluster from one image should be assign to which cluster of another image is known as the correspondence problem. It is a combinatorial optimization problem and it can be solved using the Hungarian algorithm in polynomial time (Kuhn, 1955). The algorithm models an assignment problem as a $n \times m$ cost matrix, where each element represents the cost of assigning the kth cluster in one image to the jth cluster in a different image. The algorithm performs minimization on the elements of the cost matrix.

We define the distance between the kth cluster from image 1 and the lth cluster from image 2 as

$$d_{k,l} = \sum_i |v_{ik}^1 - v_{il}^2|. \tag{22}$$

The objective is to find a permutation of the clusters from the second image that produces the highest overlapping among the clusters of the two images. This is equivalent to minimizing the following objective function over the one to one cluster permutation mapping $p : \{1, ..., K\} \rightarrow \{1, ...K\}$,

$$f(p) = \sum_{k} d_{k,p(k)} = \sum_{i,k}^{K} |v_{ik}^{1} - v_{ip(k)}^{2}| \tag{23}$$

where k goes over all clusters, and i over all voxels.

To learn the parameter s_{ik} for a given class, we solve the correspondence among all the training images of that class. To calculate the CCLF given by Eq. (22), we then solve the correspondence problem between the given test image and each class of training images.

3.6.4 Experimental results

In this section, we evaluate the performance of our non-parametric (NP) classifier on raw and segmented fMRI images. To cluster fMRI images we used both the deterministic (k-means) and probabilistic (HMRF) algorithms. We used $K = 5$ for k-means algorithm, and the parameters for the HMRF were $K = 3$ and $\lambda = 1$.

The fMRI data used in this work were recorded while the subjects were looking at face images and trying to detect their emotional expressions. Each face was shown for very brief amount of time, $33ms$, making the task very difficult. The images consisted of disgusted and fearful faces so the number of classes was two. The number of fMRI images for each class was 31. For each subject, we used 30 images from each class for training and 1 image from each class for testing. The data were collected with a Siemens 1.5-T scanner, and the activation of each voxel was represented with 7 points. In order to reduce the dimensionality of the data, we modeled the voxel activation curve by fitting it to a polynomial function of the second order.

$$z_i(t) = a_i t^2 + b_i t + c_i. \tag{24}$$

As a result, activation of each voxel was represented not with a scalar but with a 3D vector.

Subject	1	2	3	4	5
NP (%)	57.6	62.1	56.1	54.5	63.3
Gaussian (%)	50.0	50.0	56.1	50.0	39.4

Table 4: Classification rates for the non-parametric and Gaussian likelihood classifiers on the raw fMRI images.

In the first experiment, we contrasted the classification rates of the NP classifier and the Gaussian Naive Bayes (GNB) classifier using raw images. We chose the GNB classifier since it is a is a

parametric model and was successfully used for classifying fMRI images (Mitchell et al., 2004). The basic assumptions behind the GNB model are that the voxel activations are independent from one another and that activation of each voxel, z_i, can be modeled using a Gaussian distribution

$$p(z_i|\mu_i, \sigma_i) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(z_i - \mu_i)^2}{2\sigma^2}\right). \tag{25}$$

Parameters μ_i and σ_i are estimated from training samples as, $\mu_i = \frac{1}{m}\sum_j^m z'_{ij}, \sigma_i^2 = \frac{1}{m-1}\sum_j^m (z'_{ij} - \mu_i)^2$ where z'_{ij} is the i-th component of j-th training sample. The results are illustrated in Table 4. As one can see, the NP classifier outperforms the GNB classifier which suggests that the Gaussian assumption for the voxel activations is not the most appropriate for this dataset.

In the following two experiments, we contrasted the classification rates of the NP and ML k-means classifiers. We should note that while the NP classifier can be applied to images in which each voxel has a multilabel assignment, the ML k-means classifier is applied to segmented images in which each voxel has a binary assignment. This is because the ML k-means classifier basically compares only the shapes of distributions among different class images. To obtain a binary representation of segmented images, we assigned the zero label to voxels of the largest cluster (the ones with green color in Figure 18), and the label one to all other voxels. As it turns out, the identification of the largest cluster was always trivial since it contains the white matter and mostly inactive voxels within the grey matter area. The ML k-means algorithm captures the shape of the cluster with label 1 using a spatial distribution function, e.g., mixture of Gaussians

$$p(\vec{x}_i) \sim \sum_j^J \pi_j f_j(\vec{x}_i, \vec{\mu}_j, \vec{\Sigma}_j), \tag{26}$$

where $\vec{x}_i$ is the spatial coordinate of the i-th voxel, J is the number of Gaussians, $\vec{\mu}_j$ and $\vec{\Sigma}_j$ are the mean and variance matrix of the j-th Gaussian. It is a $K-$means clustering algorithm based on the coordinates of voxels with label 1. To estimate the parameters of the ML k-means classifier, we used an Expectation Maximization (EM) algorithm. In our implementation, we treat the mixture density as the likelihood function and estimate the class label by maximizing the likelihood function conditioned on all possible classes.

Subject	1	2	3	4	5
NP (%)	68.3	58.3	65.0	61.7	63.3
ML k-means	51.7	50.0	48.3	50.0	51.7

Table 5: Classification rates for the NP and ML k-means classifiers on images segmented with HMRF method.

In the second experiment, we compared the classification rates of the NP and ML k-means classifiers on the fMRI images that were clustered using HMRF model and the collapsed VB inference

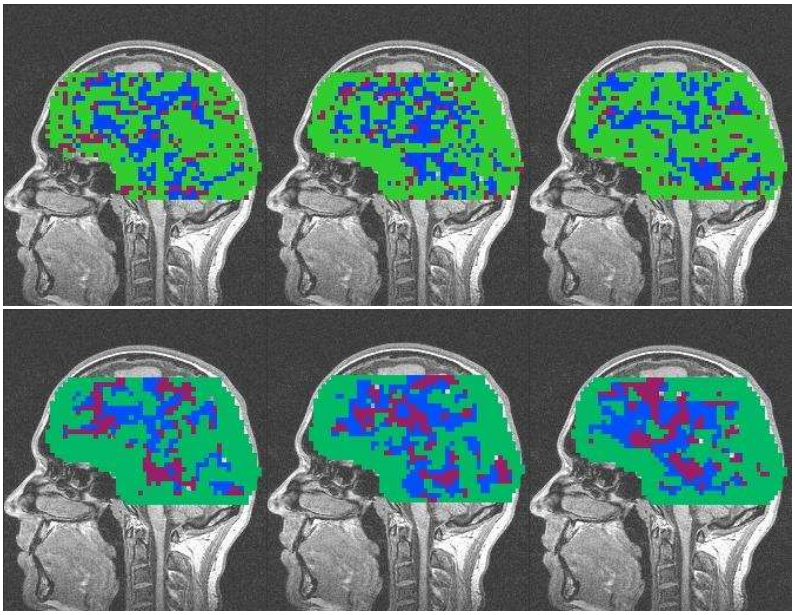

Figure 18: Sagittal views of segmented fMRI image using k-means (top row), and Hidden Markov Random Field (HMRF) methods. The number of clusters is the same, K=3, for both methods.

algorithm. This means that the inputs to the NP classifier was applied directly to HMRF segmented images in which each voxel has multiple probabilistic assignments. As one can see from Table 5, the classification rates when using the NP classifier are consistently higher than the rates when the images were classified with ML k-means algorithm.

Subject	1	2	3	4	5
NP(%)	58.3	55.0	48.3	51.7	65.0
ML k-means(%)	53.3	50.0	51.7	50	48.3

Table 6: Classification rates for the NP and ML k-means classifiers on images segmented with k-means method.

In the third experiment we used as a clustering method the k-means algorithm. As shown in Table 6, the NP classifier outperforms the ML k-means classifier for most subjects. In addition, by contrasting Tables 5 and 6, it is clear that the NP classifier was able to better utilize probabilistic assignment of voxels and produced higher classification rates (compared to ML k-means classifier) when using the images segmented with the HMRF method.

In Figure 18, we present several images that were segmented using k-means (top row) and HMRF (bottom row) methods. In this example, we used the same number of clusters in both algorithms, K=3. The largest cluster (green color) denotes mostly white matter and inactive grey matter voxels (the "background"). As one can see, the k-means algorithm produces much sparser clusters

with one dominating cluster (blue color), compared to the HMRF algorithm which produced much tighter and more balanced cluster distributions.

4 Summary of the most important results

In summary, within the period of September 1st 2005 and September 1st 2009, we successfully completed the following projects:

• Developed a biologically inspired algorithm for learning new object categories and tested the performance on a real-world database consisting of different object categories. Specifically we: 1) developed a Bayesian model for integrating information from different regions of the visual field and for integrating information across fixations (Neskovic et al., 2006a), 2) constructed an interactive interface for training the system, 3) tested the dependence of the learning algorithms on the distribution of the receptive fields and demonstrated advantages of the retina-like distribution over the fixed size distribution of the receptive fields (Wu et al., 2006a), 4) demonstrated that the model is robust to partial occlusions and clutter and can recognize a target even if it fixates on the occluded part (Neskovic et al., 2006b), and 5) demonstrated that the recognition system (Neskovic et al., 2006a) is robust to significant variations of the sizes and locations of the fixation regions (Neskovic et al., 2007; Neskovic et al., 2009).

• Developed a new algorithm for localizing objects and object features in new images in which objects appear at different locations, sizes and different lighting conditions (Wu et al., 2006a).

• Developed new classification algorithms: 1) a minimum bounding sphere algorithm for classifying object categories (Wang et al., 2007a), and 2) developed a new decision rule, based on the Bayesian decision theory, that partitions the feature space using a small number of bounding spheres (Wang et al., 2007a), 3) designed a new classification method that uses a single sphere (Wang et al., 2005b), in the feature space, to separate object classes. In addition, we designed a new procedure for selecting data for support vector machine (SVM) training (Wang et al., 2007c).

• We introduced an adaptive distance measure (Wang et al., 2006; Wang et al., 2007b) that significantly improved the performance of one of the most powerful classification algorithms in use today, the k-NN algorithm, and tested the new classification algorithms on real-world examples and achieved state-of-the-art recognition rates.

• Constructed a model that can improve learning by utilizing information from unlabeled training samples (Wu and Neskovic, 2007). More specifically, we developed a new algorithm and tested it on both artificial and real datasets. We concluded that the contribution from the unlabeled example is very high in situations when the number of training examples is small resulting in classification rates that are significantly larger compared to the performance of the classifiers that did not include

the unlabeled example. We also showed that the procedure works extremely well in cases when one class has a dominating prior. Most importantly, we demonstrated that the performance of the self-improving classifier improves with the dimensionality of the feature space which is in contrast to most existing classifiers.

• Developed a model for classifying segmented images (Wu et al., 2008). Specifically, we developed a non-parametric (NP) model that can classify both raw and segmented images and tested it on real fMRI images. We showed that our model can successfully classify whole brain images (without a feature selection stage) using challenging single trial examples. We demonstrated that the NP classifier is very general in the sense that it can deal not only with images that were segmented with deterministic segmentation algorithms but also with probabilistic clustering approaches. Furthermore, we showed that our classifier can be used not only on binary images, but also on images that contain multiple clustering labels which can be of great importance when analyzing medical data.

5 List of publications

Papers published in peer-reviewed journals, books or conference proceedings:

1) P. Neskovic, I. Sherman, L. Wu, and L. N. Cooper. Learning faces with the BIAS model: on the importance of the sizes and locations of fixation regions, Neurocomputing, vol. 72, pp. 2915-2922, 2009.

2) L. Wu, P. Neskovic, and L. N. Cooper. A probabilistic model for classifying segmented images, ICPR 2008.

3) H. Ren, L. Wu, P. Neskovic, and L. N. Cooper. Approximating a non-homogenous HMM with dynamic spatial Dirichlet process, ICPR 2008.

4) P. Neskovic, Ian Sherman, L. Wu, and L. N. Cooper. How Important are the Sizes and Locations of the Fixation Regions for the BIAS Model? Natural Computation, Vol. 2, pp. 17-21, 2007.

5) J. Wang, P. Neskovic, and L. N. Cooper. Bayes Classification Based on Minimum Bounding Spheres. Neurocomputing, Vol. 70, pp. 801-808, 2007.

6) J. Wang, P. Neskovic, and L. N. Cooper. Improving Nearest Neighbor Rule with a Simple Adaptive Distance Measure. Pattern Recognition Letters, 28(2), pp. 207-213, 2007.

7) J. Wang, P. Neskovic, and L. N. Cooper. Selecting Data for Fast Support Vector Machine Training. Studies in Computational Intelligence, Vol. 35, pp. 61-84, 2007.

8) P. Neskovic, L. Wu, and L. N. Cooper. Learning by Integrating Information Within and Across Fixations. Lecture Notes In Computer Science: Artificial Neural Networks - ICANN, Vol. 4132, pp. 488-497, 2006.

9) J. Wang, P. Neskovic, and L. N. Cooper. A minimum Sphere Covering Approach to Pattern Classification. ICPR, pp. 433-436, 2006.

10) J. Wang, P. Neskovic, and L. N. Cooper. Neighborhood Size Selection in the k-Nearest Neighbor Rule Using Statistical Confidence. Pattern Recognition, 39(3), pp. 417-423, 2006.

11) L. Wu, P. Neskovic, and L. N. Cooper. Biologically Inspired Hierarchical Model for Feature Extraction and Localization. ICPR, pp. 259-262, 2006.

12) J. Wang, P. Neskovic, and L. N. Cooper. Improving Nearest Neighbor Rule with a Simple Adaptive Distance Measure. Lecture Notes In Computer Science: Advances in Natural Computation - ICNC, Vol. 4221, pp. 43-46, 2006.

13) L. Wu, P. Neskovic, and L. N. Cooper. Biologically Inspired Bayes Learning and its Dependence on the Distribution of the Receptive Fields. Lecture Notes In Computer Science: Advances in Natural Computation - ICNC, Vol. 4221, pp. 279-288, 2006.

14) J. Wang, P. Neskovic, and L. N. Cooper. Pattern classification via single spheres. Lecture Notes in Computer Science: Discovery Science (DS), A. Hoffmann, H. Motoda, and T. Scheffer (Eds.), Springer-Verlag, Vol. 3735. pp. 241-252 , 2005.

15) J. Wang, P. Neskovic, and L. N. Cooper. A Statistical Confidence-Based Adaptive Nearest Neighbor Algorithm for Pattern Classification. Lecture Notes in Computer Science: Advances in Machine Learning and Cybernetics (ICMLC), Vol. 3930, pp. 548-557, 2005.

16) P. Neskovic and L. N. Cooper. Visual Search for Object Features. Lecture Notes In Computer Science: Advances in Natural Computation (ICNC), L. Wang, K. Chen, Y. S. Ong (Eds.), Vol. 3610, pp. 877-887, 2005.

17) J. Wang, P. Neskovic, and L. N. Cooper. Training Data Selection for Support Vector Machines. Lecture Notes In Computer Science: Advances in Natural Computation (ICNC), L. Wang, K. Chen, Y. S. Ong (Eds.), Vol. 3610, pp. 554-564, 2005.

18) J. Wang, P. Neskovic, and L. N. Cooper. Locally Determining the Number of Neighbors in the k-Nearest Neighbor Rule Based on Statistical Confidence, Lecture Notes In Computer Science: Advances in Natural Computation (ICNC), L. Wang, K. Chen, Y. S. Ong (Eds.), Vol. 3610, pp. 71-80, 2005.

19) J. Wang, P. Neskovic, and L. N. Cooper. Pattern classification based on minimum bounding spheres. International Conference on Intelligent Computing (ICIC), pp. 1969 - 1978, 2005.

20) J. Wang, P. Neskovic, and L. N. Cooper. A Probabilistic Model For Cursive Handwriting Recognition Using Spatial Context. ICASSP, Vol. 5, pp. 201-204, 2005.

Technical Reports:

21) L. Wu, P. Neskovic and Luiz Pessoa. Dirichlet Process Mixture Model with Spatial Constraints. Technical Report IBNS-TR-2007-02, Brown University, 2007.

22) J. Wang, P. Neskovic, and L. N. Cooper. A Minimum Sphere Covering Approach to Learning. IBNS Technical Report 2006-03, Brown University, 2006.

23) J. Wang, P. Neskovic, and L. N. Cooper. Learning Class Regions by Sphere Covering. IBNS Technical Report 2006-02, Brown University, 2006.

24) P. Neskovic, L. Wu, and L. N. Cooper. Biologically Inspired Bayesian Approach for Learning Object Categories From Few Training Examples. IBNS Technical Report 2006-01, Brown University, 2006.

References

Abe, S. and Inoue, T. (2001). Fast training of support vector machines by extracting boundary data. In *Proceedings of the International Conference on Artificial Neural Networks (ICANN)*, pages 308–313.

Agarwal, S., Awan, A., and Roth, D. (2004). Learning to detect objects in images via a sparse, part-based representation. *PAMI*, 26(11):1475–1490.

Amit, Y. and Mascaro, M. (2003). An integrated network for invariant visual detection and recognition. *Vision Research*, 43(19):2073–2088.

Baluja, S. (1998). Probabilistic modeling for face orientation discrimination: Learning from labeled and unlabeled data.

Batchelor, B. G. (1974). *Practical Approach to Pattern Classification*. Plenum, New York.

Blum, A. and Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. In *Proceedings of the Workshop on Computational Learning Theory*.

Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In Haussler, D., editor, *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, pages 144–152.

C. Harris, M. S. (1988). A combined corner and edge detector. In *Proc. 4th Alvey Vision Conference*, pages 147–152.

Cooper, P. W. (1962). The hypersphere in pattern recognition. *Information and Control*, 5:324–346.

Cortes, C. and Vapnik, V. N. (1995). Support vector networks. *Machine Learning*, 20:273–297.

Cover, T. M. and Hart, P. E. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13:21–27.

Cox, D. D. and Savoy, R. L. (2003). Functional magnetic resonance imaging (fMRI) "brain reading": detecting and classifying distibuted patterns of fMRI activity in human visual cortex. *NeuroImage*, 19:261–270.

Cozman, F. and Cohen, I. (2002). Unlabeled data can degrade classification performance of generative classifiers. In *Fifteenth International Florida Artificial Intelligence Society Conference*, pages 327–331.

Domeniconi, C., Peng, J., and Gunopulos, D. (2002). Locally adaptive metric nearest neighbor classification. *IEEE PAMI*, 24:1281–1285.

Fei-Fei, L., Fergus, R., and Perona, P. (2003). A bayesian approach to unsupervised one-shot learning of object categories. In *Proc. ICCV*.

Felzenszwalb, P. and Huttenlocher, D. (2005). Pictorial structures for object recognition. *Intl. Journal of Computer Vision*, 61(1):55–79.

Fix, E. and Hodges, J. (1951). Discriminatory analysis, nonparametric discrimination: consistency properties. Tech. report 4, USAF School of Aviation Medicine, Randolph Field, Texas.

Friedman, J. (1994). Flexible metric nearest neighbor classification. Tech. report, Stanford University, Statistics Department.

Geisser, S. (1964). Posterior odds for multivariate normal distributions. *Journal of Royal Society Series B Methodological*, 26:69–76.

Goldberger, J., Roweis, S., Hinton, G., and Salakhutdinov, R. (2004). Neighbourhood component analysis. In *Advances in Neural Information Processing Systems*, volume 17, pages 513–520. Morgan Kaufmann.

Goldman, S. and Zhou, Y. (2000). Enhancing supervised learning with unlabeled data. In *Proc. 17th International Conf. on Machine Learning*, pages 327–334. Morgan Kaufmann, San Francisco, CA.

Gomes, H. M. (2002). Model learning in iconic vision. Ph. D. Thesis, University of Edinburgh, Scotland.

Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3):389–422.

Hastie, T. and Tibshirani, R. (1996). Discriminant adaptive nearest neighbor classification. *IEEE PAMI*, 18(6):607–615.

Haynes, J. D. and Rees, G. (2005). Predicting the orientation of invisible stimuli from activity in human primary visual cortex. *Nat Neuroscience*, 8(5):686–691.

Hecht-Nielsen., R. and Zhou, Y. (1995). VARTAC: A foveal active vision ATR system. *Neural Networks*, 8(7/8):1309–1321.

Heisele, B., Serre, T., Pontil, M., Vetter, T., and Poggio, T. (2001). Categorization by learning and combining object parts. In *Proc. NIPS*.

Huang, S. and Lee, Y. (2004). educed support vector machines: a statistical theory.

Joachims, T. (1999). Making large-scale svm learning practical. In Schölkopf, B., Burges, C., and Smola, A., editors, *Advances in Kernel Methods - Support Vector Learning*, pages 169–184. MIT Press, Cambridge, MA.

Kadir, T. and Brady, M. (2001). Scale, saliency and image description. *Intl J. Computer Vision*, 45(2).

Keehn, D. G. (1965). A note on learning for gaussian properties. *IEEE Transactions on Information Theory*, 11:126–132.

Keller, J., Rogers, S., Kabrisky, M., and Oxley, M. (1999). Object recognition based on human saccadic behaviour. *Pattern Analysis and Applications*, 2:251–263.

Kuhn, H. (1955). The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97.

Lee, T. S. (1996). Image representation using 2d gabor wavelets. *PAMI*, 18(10):1–13.

Lee, Y. and Mangasarian, O. (2001). Rsvm: Reduced support vector machines. In *Proceedings of the First SIAM International Conference on Data Mining*.

Levi, K. and Weiss, Y. (2004). Learning object detection from a small number of examples: the importance of good features. In *CVPR*.

Li, F.-F., Fergus, R., and Perona, P. (2006). One-shot learning of object categories. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(4):594–611.

Lowe, D. (1999). Object recognition from local scale-invariant features. In *Proc. ICCV*.

Lowe, D. (2000). Towards a computational model for object recognition in it cortex. In *Proc. Intl Workshop on Biologically Motivated Computer Vision*.

Marchand, M. and Shawe-Taylor, J. (2002). The set covering machine. *Journal of Machine Learning Research*, 3:723–746.

Mel, B. (1997). Seemore: Combining color, shape and texture histogramming in a neurally-inspired approach to visual object recognition. *Neural Computation*, 9(4):777–804.

Mitchell, T. M., Hutchinson, R., Niculescu, R. S., Pereira, F., Wang, X., Just, M., and Newman, S. (2004). Learning to decode cognitive states from brain images. *Machine Learning*, 57(1-2):145–175.

Neskovic, P., Davis, P., and Cooper, L. (2000). Interactive parts model: an application to recognition of on-line cursive script. In *Advances in Neural Information Processing Systems*.

Neskovic, P., Schuster, D., and Cooper, L. (2004). Biologically inspired recognition system for car detection from real-time video streams. In *Neural Information Processing: Research and Development*, pages 320–334. J. C. Rajapakse and L. Wang (Eds.), Springer - Verlag.

Neskovic, P., Sherman, I., Wu, L., and Cooper, L. (2007). How important are the sizes and locations of the fixation regions for the bias model? In *Proc. International Conference on Natural Computation*, volume 2, pages 17–21.

Neskovic, P., Sherman, I., Wu, L., and Cooper, L. (2009). Learning faces with the bias model: on the importance of the sizes and locations of fixation regions. *Neurocomputing*, 72:2915–2922.

Neskovic, P., Wu, L., and Cooper, L. (2006a). Learning by integrating information within and across fixations. In *Proc. ICANN*, pages 488–497.

Neskovic, P., Wu, L., and Cooper, L. (2006b). A system for automatic detection of occluded objects from real world images. In *25th Army Science Conference*.

Nigam, K. and Ghani, R. (2000). Analyzing the effectiveness and applicability of co-training. In *CIKM: Ninth International Conference on Information and Knowledge Management*, pages 86–93.

Noton, D. and Stark, L. (1971). Scanpaths in eye movements during pattern perception. *Science*, 171:308–311.

Pessoa, L. and Padmala, S. (2006). Decoding near-threshold perception of fear from distributed single-trial brain activation. *Cerebral Cortex*, 17:691–701.

Pokrajac, D., Megalloikonomou, V., Lazarevic, A., Kontos, D., and Obradovic, Z. (2005). Applying spatial distribution analysis techniques to classification of 3D medical images. *Artificial Intelligence in Medicine*, 33:261–280.

Reilly, D. L., Cooper, L. N., and Elbaum, C. (1982). A neural model for category learning. *Biological Cybernetics*, 45:35–41.

Riesenhuber, M. and Poggio, T. (1999). Hierarchical models of object recognition in cortex. *Nat. Neurosci.*, 2(11):1019–1025.

Rybak, I. A., Gusakova, V. I., Golovan, A., Podladchikova, L. N., and Shevtsova, N. A. (1998). A model of attention-guided visual perception and recognition. *Vision Research*, 38:2387–2400.

Schmid, C. and Mohr, R. (1997). Local greyvalue invariants for image retrieval. *PAMI*, 19(5):530–534.

Schneiderman, H. and Kanade, T. (2000). A statistical method for 3d object detection applied to faces and cars. In *Proc. CVPR*.

Schölkopf, B., Burges, C., and Vapnik, V. (1995). Extracting support data for a given task. In *Proceedings of First International Conference on Knowledge Discovery and Data Mining*, pages 252–257.

Schwartz, E. (1977). Spatial mapping in primate sensory projection: analytic structure and relevance to perception. *Biological Cybernetics*, 25:181–194.

Scofield, C. L., Reilly, D. L., Elbaum, C., and Cooper, L. N. (1987). Pattern class degeneracy in an unrestricted storage density memory. In Anderson, D. Z., editor, *Neural Information Processing Systems*, pages 674–682.

Serre, T., Wolf, L., and Poggio, T. (2004). A new biologically motivated framework for robust object recognition. Ai memo 2004-026, cbcl memo 243, MIT.

Serre, T., Wolf, L., and Poggio, T. (2005). Object recognition with features inspired by visual cortex. In *Proc. CVPR*.

Shin, H. J. and Cho, S. Z. (2003). Fast pattern selection for support vector classifiers. In *Proceedings of the 7th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 376–387. Lecture Notes in Artificial Intelligence (LNAI 2637).

Smeraldi, F. and Bigun, J. (2002). Retinal vision applied to facial features detection and face authentication. *Pattern Recognition Letters*, 23:463–475.

Srivastava, S. and Gupta, M. R. (2006). Distribution-based bayesian minimum expected risk for discriminant analysis. In *IEEE International Symposium on Information Theory*, pages 2294–2298.

Stone, C. J. (1977). Consistent nonparametric regression. *Annals of Statistics*, 5:595–645.

Syed, N., Liu, H., and Sung, K. (1999). A study of support vectors on model independent example selection. In *Proceedings of the Workshop on Support Vector Machines at the International Joint Conference on Artificial Intelligence*.

Tax, D. M. J. and Duin, R. P. W. (1999). Data domain description by support vectors. In Verleysen, M., editor, *Proceedings ESANN*, pages 251–256, Brussels. D. Facto Press.

Torralba, A., Murphy, K. P., and Freeman, W. T. (2004). Sharing features: efficient boosting procedures for multiclass object detection. In *Proc. CVPR*.

Vapnik, V. N. (1998). *Statistical Learning Theory*. Wiley, New York, NY.

Viola, P. and Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *Proc. CVPR*.

Walther, D., Rutishauser, U., Koch, C., and Perona, P. (2005). Selective visual attention enables learning and recognition of multiple objects in cluttered scenes. *Computer Vision and Image Understanding*, 100(2):41–63.

Wang, J., Neskovic, P., and Cooper, L. (2005a). Pattern classification based on minimum bounding spheres. In *Proc. ICIC*, pages 1969–1978.

Wang, J., Neskovic, P., and Cooper, L. (2005b). Pattern classification via single spheres. In *Lecture Notes in Computer Science: Discovery Science*, volume 3735, pages 241–252.

Wang, J., Neskovic, P., and Cooper, L. (2005c). A probabilistic model for cursive handwriting recognition using spatial context. In *Proc. ICASSP*.

Wang, J., Neskovic, P., and Cooper, L. (2006). Improving nearest neighbor rule with a simple adaptive distance measure. In *Lecture Notes In Computer Science: Advances in Natural Computation*, pages 43–46.

Wang, J., Neskovic, P., and Cooper, L. (2007a). Bayes classification based on minimum bounding spheres. *Neurocomputing*, 70:801–808.

Wang, J., Neskovic, P., and Cooper, L. (2007b). Improving nearest neighbor rule with a simple adaptive distance measure. *Pattern Recognition Letters*, 28(2):207–213.

Wang, J., Neskovic, P., and Cooper, L. (2007c). Selecting data for fast support vector machine training. In Chen, K. and Wang, L., editors, *Trends in Neural Computation*, pages 320–334. Springer - Verlag.

Wang, J., Neskovic, P., and Cooper, L. N. (2003). Partitioning a feature space using a locally defined confidence measure. In *Joint 13th International Conference on Artificial Neural Networks and 10th International Conference on Neural Information Processing*.

Weinberger, K., Blitzer, J., and Saul, L. (2005). Distance metric learning for large margin nearest neighbor classification. In *Advances in Neural Information Processing Systems*, volume 18. Morgan Kaufmann.

Welling, M., Kurihara, K., and Teh, Y. W. (2007). Collapsed variational dirichlet process mixture models. In *IJCAI*, pages 2796–2801.

Wilson, S. (1983). On the retino-cortical mapping. *International Journal on Man-Machine Studies*, 18:361–389.

Wolf, L. and Martin, I. (2005). Robust boosting for learning from few examples. In *CVPR*, pages 359–364.

Wu, L. and Neskovic, P. (2007). A self-improving procedure for bayes classification with few training examples. Tech. Report, Institute for Brain and Neural Systems, Brown University. IBNS-TR-2007-01.

Wu, L., Neskovic, P., and Cooper, L. (2006a). Biologically inspired bayes learning and its dependence on the distribution of the receptive fields. In *Lecture Notes In Computer Science: Advances in Natural Computation*, pages 279–288.

Wu, L., Neskovic, P., and Cooper, L. (2006b). Biologically inspired hierarchical model for feature extraction and localization. In *Proc. ICPR*.

Wu, L., Neskovic, P., and Cooper, L. N. (2008). A probabilistic model for classifying segmented images. In *International Conference on Pattern Recognition*. accepted.

Wu, L., Neskovic, P., and Pessoa, L. (2007). Dirichlet process mixture model with spatial constraints. Tech. Report, Institute for Brain and Neural Systems, Brown University. IBNS-TR-2007-02.

Zhang, W. and King, I. (2002). Locating support vectors via β-skeleton technique. In *Proceedings of the International Conference on Neural Information Processing (ICONIP)*, pages 1423–1427.

Zhang, Y., Brady, M., and Smith, S. (2001). Segmentation of brain MR images through a hidden markov random field model and the expectation-maximization algorithm. *IEEE Trans. Med. Imag.*, 20(1):45–57.

Zheng, S., Lu, X., Zheng, N., and Xu, W. (2003). Unsupervised clustering based reduced support vector machines. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* ***2***, pages 821–824.

Zhu, X. (2005). Semi-supervised learning literature survey. *Technical Report 1530, Computer Science, University of Wisconsin-Madison.*