5-1-2003

TOWARD AN OPTIMAL ANALYSIS OF HYPERSPECTRAL DATA

Mehmet M. Dundar

David Landgrebe

TOWARD AN OPTIMAL ANALYSIS OF HYPERSPECTRAL DATA

Mehmet M. Dundar
David Landgrebe

TR-ECE 03-07
May 2003

School of Electrical & Computer Engineering
Electrical Engineering Building
Purdue University
West Lafayette, Indiana 47907-1285

TABLE OF CONTENTS

ABSTRACT

In hyperspectral data materials of practical interest usually exist in a number of states and are observed in a number of conditions of illumination. It is thus necessary to characterize them not with a single spectral response but with a family of responses. One of the most versatile means for representing such a family of responses quantitatively is to model each by a normal distribution, as this makes possible classification by assigning the class to a sample based on Bayes rule. This leads to competitive performance only under special circumstances. The purpose of this dissertation is to improve the quantitative definitions of classes by replacing the classical normal model with more flexible and powerful alternatives and thereby investigate the relationship between class definition precision and classification accuracy. One other factor that directly affects the precision of class definitions is the number of labeled samples available for training. Characterizing class data with a limited set of labeled samples may have severe consequences. In a typical setting a remote sensing analyst should either sacrifice from the classifier performance by confining himself to the already available labeled data set or commit more time and effort to acquire more labeled samples both of which comes at a price. The present study also addresses this problem by proposing a semi-supervised binary classifier which seeks to incorporate unlabeled data into the training in order to improve the quantitative definitions of classes and hence improve the classifier performance at no extra cost.

1. INTRODUCTION

1.1 Statement of Problem

In hyperspectral data analysis, materials of practical interest, such as agricultural crops, forest plantations, natural vegetation, minerals, and fields of interest in urban areas exist in a variety of states and are usually observed in a number of conditions of illumination. That is, most land-cover types do not have a single spectral response. For example a crop type will show different spectral characteristics at different times of the day and year. Similarly, roof tops are usually made of a variety of different materials including concrete, tile, bricks, glass etc. all of which have different spectral responses. The number of such examples can easily be augmented.

Using normal densities to characterize class distributions has long been a common approach in hyperspectral data analysis. This is mostly because pixels belonging to most land cover types are assumed to follow a normality pattern in the feature space in most cases and partly because normal distribution can easily be characterized by two parameters, allowing one to obtain a simple closed-form representation of each class distribution with a limited effort. By assuming a normal distribution for each class one can end up with a simple quadratic classifier which has proved effective in many cases.

However considering the above mentioned characteristics of remotely-sensed data, the validity of the normality assumption is very much arguable. A normal model usually leads to competitive performance when the classes are unimodal and separated by scatter of means or by the covariance difference or both in the feature space. When a multimodal class data is modeled using a single normal density, the resulting distribution may significantly overlap with the distribution of some other class showing similar spectral characteristics to the original class. In this case the separability information which was present initially is lost due to an inappropriate modeling of the data.

Although this problem might be encountered in a variety of land cover types, it becomes quite problematic especially when analyzing data of urban areas. Each urban field has characteristics of its own, but one thing that is common in most urban fields is that they generally encompass quite broad area land-cover distinction. Due to their highly similar spectral characteristics, defining an informative set of classes and then classifying these classes correctly is a very demanding task. In section 1.3 we will scrutinize these issues in more detail with respect to the hyperspectral images considered in this study.

The normal model is certainly not flexible and sophisticated enough to characterize highly complex class data. It is well-known that there is a strong relationship between class definition precision and classification accuracy. Thus by defining classes more precisely, better results can be obtained for challenging classification problems. In this dissertation we will propose two novel techniques which, as the theoretical support and experimental results suggest, can be a powerful alternative for the simple quadratic classifier.

Another factor that affects the precision of class definitions in hyperspectral data analysis is the number of labeled samples available. The techniques proposed in the first part of this Report study assume there are enough labeled data to reveal the underlying structure of the class distributions. However in most real world settings, this may not be the case. The price one must pay for labeled data is usually prohibitively expensive, as acquiring labeled data requires a tedious and time consuming process of human labeling.

Characterizing class data with a limited set of labeled samples may have two severe consequences. First in the presence of few labeled samples it is very impractical to determine an efficient generic model for the class conditional distributions, mostly because the labeled samples do not reveal much about the underlying class structures. Second, even if the generic model is known with few labeled samples statistics estimation becomes quite intractable and obtaining robust estimates free of numerical glitches is almost impossible.

In a typical setting a remote sensing analyst must either sacrifice classifier performance by confining him/herself to the already available labeled data set or commit more time and effort to acquire more labeled samples, both of which come at a price.

Recently there has been an increasing interest in the readily available large amount of unlabeled data. Several studies have been conducted to show that unlabeled data may help improve the performance of the classifier. In the second part of this study we address this problem by proposing a semi-supervised classifier for binary classification problems.

1.2 Organization of Report

In Chapter 2, a model based mixture classifier, which uses mixture models to characterize class densities, is proposed. However, a key outstanding problem of this approach is how to choose the number of components and determine their parameters for such models in practice, and to do so in the face of limited training sets where estimation error becomes a significant factor. The proposed classifier estimates the number of subclasses and class statistics simultaneously by choosing the best model. The structure of class covariances is also addressed through a model-based covariance estimation technique introduced in that chapter. The approach is to divide the data from each class into a specific number of clusters using a nearest means algorithm and fit the clusters into six different covariance models through an expectation-maximization (EM) algorithm. Then, corresponding to each model, a measure of fit is computed and the model with the highest measure of fit is chosen as the best model. Once the structure of the covariance is determined for a fixed number of clusters, the next step is to estimate the mixing proportions, which can be done by maximizing the leave-one-out likelihood for each cluster. This process is repeated by incrementing the number of clusters by one up to a designated number and computing a measure of fit at each stage. Finally the mixture model with the highest measure of fit is chosen as the best characterization of the class of interest. Measures of fit are computed using an approximation to the Bayes factor also known as the Bayesian Information Criterion (BIC).

In Chapter 3, we introduce a supervised classifier based on implementation of the Bayes rule with kernels. This technique first proposes an implicit nonlinear transformation of the data into a new feature space and seeks to fit normal distributions having a common covariance matrix into the mapped data. One requirement of this

approach is the evaluation of posterior probabilities. We express the discriminant function in dot-product form, and then apply the kernel concept to efficiently evaluate the posterior probabilities. The proposed technique gives the flexibility required to model complex data structures that originate from a widerange of class conditional distributions. Although we end up with piece-wise linear decision boundaries in the feature space, these corresponds to powerful nonlinear boundaries in the original input space.

In Chapter 4, we propose a cost-effective iterative semi-supervised classifier based on a kernel concept. The proposed technique incorporates unlabeled data into the design of a binary classifier by introducing and optimizing a cost function in a feature space which maximizes the Rayleigh coefficient while minimizing the total cost associated with misclassified labeled samples. The cost assigned to misclassified labeled samples accounts for the number of misclassified labeled samples as well as the amount by which they are on the wrong side of the boundary, and this counter-balances any potential adverse effect of unlabeled data on the classifier performance. Several experiments performed with remotely-sensed data demonstrate that using the proposed semi-supervised classifier shows considerable improvements over the supervised-only counterpart.

Finally in Chapter 5, we present our concluding comments and give some insight into future research ideas.

1.3 Data Sets Used in Experiments

Three different data sets, each with significantly different characteristics, were used in this work. All three are available from the CD accompanying [47]. They are briefly described in the following.

1.3.1 Flightline C1

Flightline C1 (FLC1) is a historically significant data set which is taken from the southern part of Tippecanoe County, Indiana by the M7 scanner [1]. Although it was

collected with an airborne scanner in June 1966, the data set remains contemporary. It has several key attributes which make it valuable, especially for illustrative purposes. These are; it has a moderate number of dimensions, it contains a significant number of ground cover classes, includes many regions containing a large numbers of contiguous pixels from a given class (this makes it relatively easier to classify the data with automated algorithms) and has ground truth available. The spectral bands available cover the 0.40-1.00 μm region of the spectrum. The data set consists of 949 scan lines with 220 pixels per scan line, or 208,780 pixels. The scanner used had an instantaneous field of view (IFOV) of 3 milliradians and was flown at an altitude of 2600 ft above terrain. Each pixel was digitized to 8-bit precision. For more information about this data set as well as an experiment performed please see [47]. The list of classes and the number of labeled samples available for each class is given in Table 1.1. The image of the scene is shown in Figure 1.1. The labeled field map obtained using the ground truth is shown in Figure 1.2.

Table 1.1 List of classes and the number of labeled samples available in Flightline C1 data set.

Class Name	**Number of Labeled Samples**
Alfalfa	3624
Bare Soil	1230
Corn	10623
Oats	5732
Red Clover	12147
Rye	2383
Soybeans	22918
Wheat	10676
Water	87
Total number of samples	69466

1.3.2 Washington DC Mall

This is an airborne hyperspectral data set gathered over the Washington DC Mall collected by the HYDICE system [2]. This sensor collects data in 220 contiguous,

uniformly-spaced spectral channels in the 0.40-2.40 μm region of the visible and infrared spectrum. In the analysis the water absorption bands are removed and the remaining 191 bands are used. The original data set contains 1208 scan lines with 307 pixels in each scan line. Throughout the experiments in this study we used the portion of the image between scan lines 543 and 822. The list of classes defined and the number of labeled samples available for each class is given in Table 1.2. The image of the scene is shown in Figure 1.3. The labeled field map is shown in Figure 1.4.

Table 1.2 List of classes and the number of labeled samples available in Washington DC Mall Data Set.

Class Name	**Number of Labeled Samples**
Roof Tops	9701
Grass	3246
Roads	4248
Paths	1414
Trees	1748
Total number of samples	20357

1.3.3 Purdue Campus Data

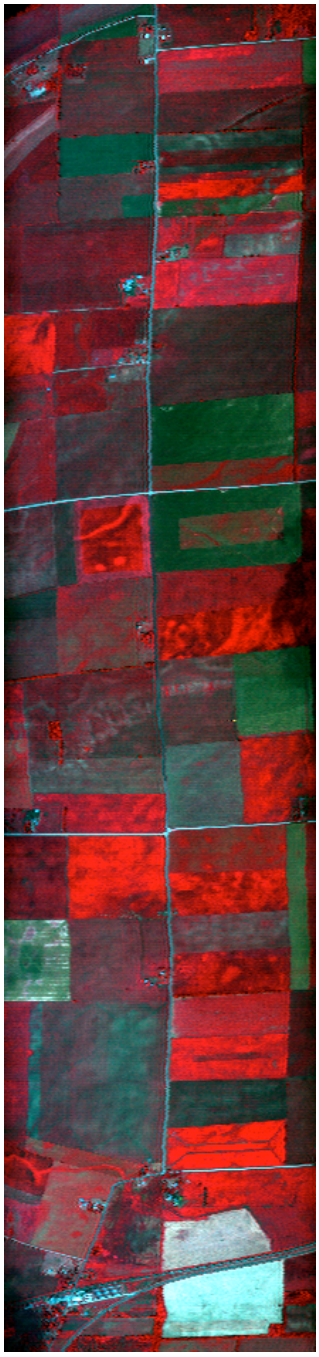
This data set is a flightline over the Purdue University West Lafayette Campus. The hyperspectral data used was collected on September 30, 1999 with the airborne HYMAP system [3], providing image data in 126 spectral bands in the visible and IR regions (0.4 □m – 2.4 □m). The system was flown at an altitude such that the pixel size is about 5 meters. The data set contains 358 scan lines with 390 pixels in each scan line. The list of classes and number of labeled samples for each class is given in Table 1.3. The image of the scene is shown in Figure 1.5. The labeled field map is shown in Figure 1.6. A Purdue campus map is provided in Fig. 1.7 to designate some landmark structures.

Note that this map is older than the aerial scene, it is slightly miss-scaled, and only the university facilities are shown.

This data set is very challenging to analyze in many respects. First it is very hard to define a set of efficient informative classes. There are three parking garages and many parking lots throughout the campus, and they all show similar spectral behavior in the image data due to the presence of large numbers of cars. It is very hard to distinguish between these two unless another class is assigned for cars. Second, some of the parking lots are graveled, and they differ from others, which are made of asphalt. Third, rooftops are made of a great variety of materials including glass (garden frame). Fourth, paths are spectrally very similar to some rooftops and streets. Fifth, basketball fields, tennis courts, tracking fields, all of which may show different spectral behavior, are all grouped into the same class.

Table 1.3 List of classes and the number of labeled samples available in Purdue Campus Data Set.

Class Name	**Number of Labeled Samples**
Roof Tops	9701
Grass	3246
Roads	4248
Paths	1414
Trees	1748
Total number of samples	20357



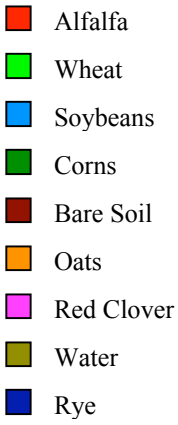


Fig. 1.1 Image of Flightline C1 Data Set

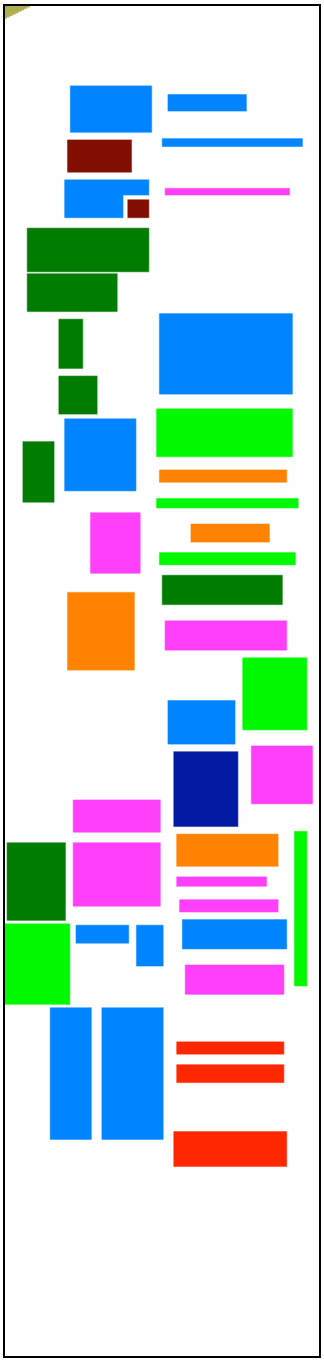


Fig. 1.2 Labeled field map for Flightline C1

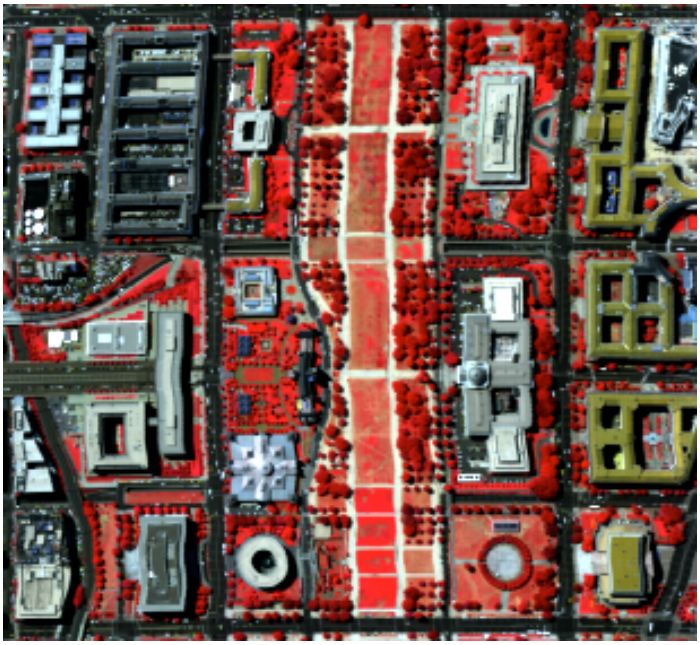


Fig. 1.3 Image of Washington DC Mall Data Set

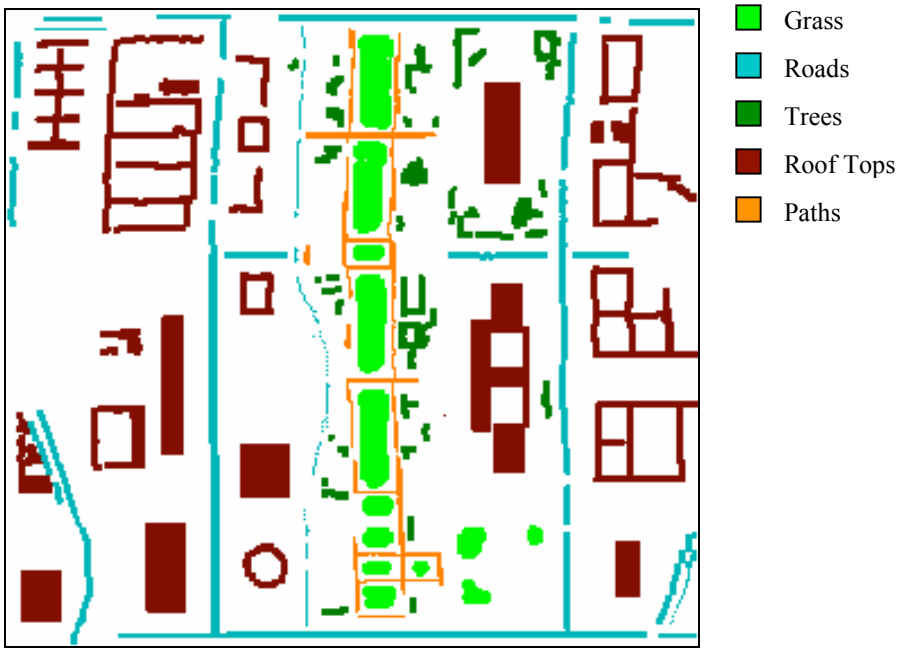


Fig. 1.4 Labeled Field Map for Washington DC Mall Data Set

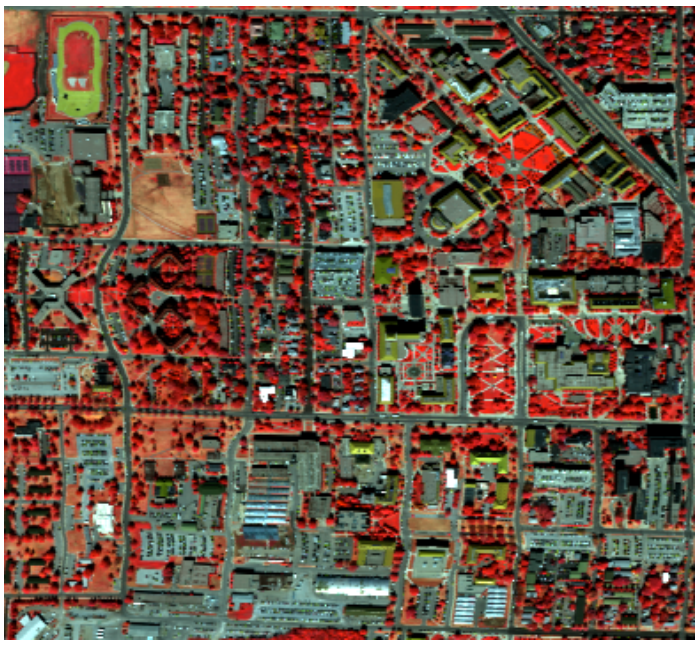


Fig. 1.5 Image of Purdue Campus Data Set

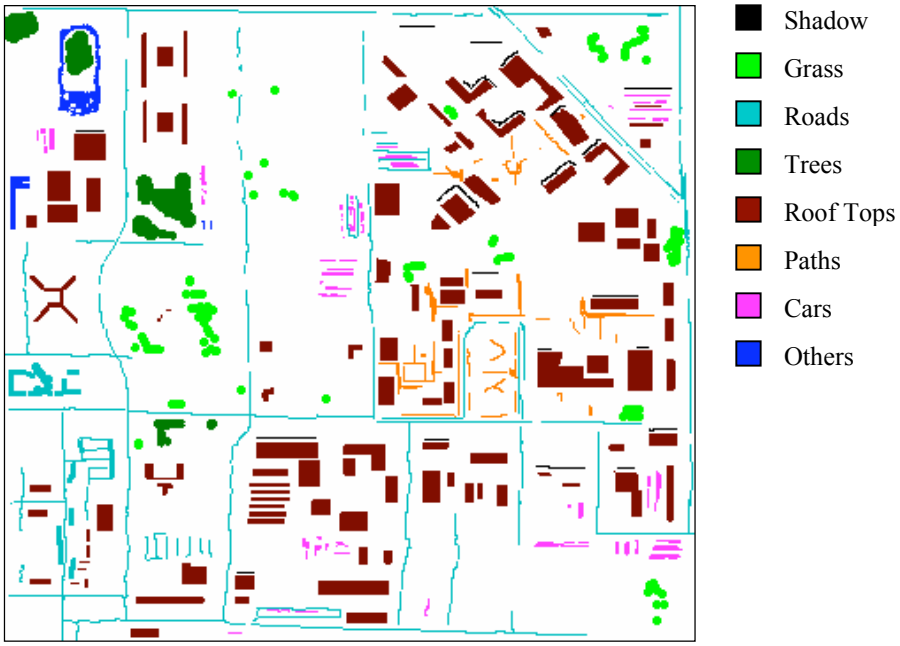


Fig. 1.6 Labeled Field Map for Purdue Campus Data Set

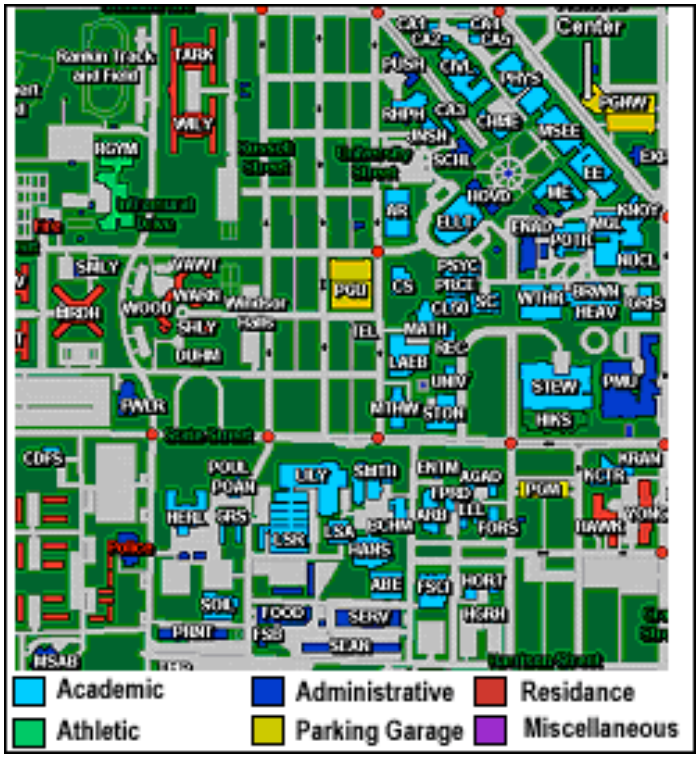


Fig. 1.7 Purdue Campus Map [4].

2. A MODEL BASED MIXTURE SUPERVISED CLASSIFICATION APPROACH

2.1 Introduction

In hyperspectral data analysis, materials of practical interest, such as agricultural crops, forest plantations, natural vegetation, minerals, and fields of interest in urban areas exist in a number of states and are observed in a number of conditions of illumination. It is thus necessary to characterize them not with a single average or typical spectral response, but with a family of responses.

One of the most powerful and versatile means for representing such a family of responses quantitatively is to model each as a multivariate probability density function, as this makes possible classification by assigning the class to a sample based on likelihood. Maximum likelihood methods have long been used in the field of digital communication systems and have made possible communication in very noisy and complex environments. To use these methods for multispectral data classification, one must determine the probabilistic density function that correctly models each class of a data set. Most commonly this is done by estimating at least the first two orders of statistics, the mean vector and the covariance matrix, for each class.

Quantifying higher order statistics of each class rather than just mean and covariance to better fit the data into a parametric class density function might seem desirable at first sight. Indeed, it is well established that, in theory a complete description of an arbitrary distribution can be made by the use of statistics of all orders, as in an infinite series. The reason it is customary to use no more than two orders of statistics, the mean vector and the covariance matrix, arises from the practical problem of estimating these two statistics from the available set of labeled data. If one were to estimate higher

order statistics one would definitely need more labeled samples to arrive at an adequately precise estimate. Usually the number of these samples is very limited since the labeling of such samples is one of the most onerous and time-consuming aspects of designing a classifier. Indeed, there often is not much information available about the scene to use in labeling the samples. Thus it is not practical to use statistics beyond the second order [5].

Another solution might be to use nonparametric class density estimations. A nonparametric classifier does not rely on any assumption about the structure of the underlying density function. The classifier may become the *Bayes* classifier and the resulting error *Bayes* error, the smallest achievable error, if the density estimates converge to the true densities, which is only achieved when an infinite number of labeled samples are used. When the densities are estimated non-parametrically with limited number of samples, the estimate is far less reliable with larger bias and variance than the parametric counterpart [6].

The normal mixture model, which is the sum of one or more weighted Gaussian components, combines much of the flexibility of nonparametric methods with certain of the analytic advantages of parametric methods. Under fairly weak conditions and given enough components a mixture model can approximate a given density arbitrarily closely allowing great flexibility.

Although, mixture models are widely used in applied statistics and recently in knowledge discovery and data mining, they have not attracted much attention in the remote sensing community. In [7], Hoffbeck and Landgrebe have introduced a method for using mixture models to characterize the class densities. Their approach is to partition the data into a designated number of clusters, and compare the mixture models corresponding to each of these partitions by means of a selection criterion. They used a nearest means clustering algorithm to partition the data and leave-one-out likelihood criterion (LOOL) to select the best fit. This method has two shortcomings. First, statistics estimation, which is done through maximum likelihood, does not address the ill-conditioned cluster covariance case. This is almost unavoidable when clustering the data of limited size to several subclusters in hyperdimensional space. Second, nearest mean clustering alone does not provide an efficient model fit.

In this work, we will address these issues by proposing a method based on model-based expectation-maximization (EM). When EM is used, data is better fit into the model and hence more reliable measures of fit are obtained. EM requires an initialization, which is done by initial partitioning of the data through a nearest mean clustering algorithm. The model based approach, which utilizes the common and sample covariance matrices as well as their trace and diagonal forms not only helps us to identify the underlying class structure providing better characterization but also produces robust estimates of component covariance matrices when the number of labeled samples in each cluster is less than the dimensionality.

The measures of fit are computed through an approximate Bayes factor, which is known as the Bayesian Information Criterion (BIC) [8].

The design of the proposed mixture classifier involves three core stages. These are:

1. Initialization and clustering
2. Expectation-maximization
3. Model selection

The rest of this chapter is organized as follows. In the first section, we give the necessary background in expectation-maximization (EM) for mixture models. In the second section, a model based approach for identifying mixtures is introduced. In the third section experiments with real aerial data are performed to test the proposed classifier.

2.2 Expectation Maximization for Finite Mixture Models

The mixture model approach assumes that the data $\Box = [\mathbf{x}_1,...,\mathbf{x}_n]$ in R^d (a d-dimensional feature space) arises from a linear combination of component density functions resulting in a mixture probability density function of the form:

$$f(X) = \sum_{k=1}^{K} \Box_k f_k(X \mid \mathbf{m}_k, \Box_k) \qquad (2.1)$$

where K is the number of mixture components, the $\Box_k$'s are the mixing proportions

with $\sum_{k=1}^{K} \pi_k = 1,\ 0 \le \pi_k \le 1,\ 1 \le k \le K$ and $f(X \mid \mathbf{m}_k, \Sigma_k)$ denotes the conditional density of class k given mean vector $\mathbf{m}_k$ and covariance matrix Σ_k.

There are two commonly used approaches in mixture analysis, the mixture approach and the classification approach. In short, the mixture approach aims to maximize the likelihood over the mixture parameters, whereas the classification approach aims to maximize the likelihood over the mixture parameters and the identifying labels of the mixture components for each sample.

In the mixture approach there is no direct interest in the discrete labeling of the samples. More specifically, the parameter set is chosen to maximize the log-likelihood.

$$L(\Theta \mid X) = \sum_{i=1}^{n} \ln\left[\sum_{k=1}^{K} \pi_k f(\mathbf{x}_i \mid \mathbf{m}_k, \Sigma_k)\right] \tag{2.2}$$

In the classification approach, the indicator vectors, $\mathbf{z}_i = (z_{ik}, k = 1,...,K)$ with $z_{ik} = 1$ or 0 identifies the mixture component, according as $\mathbf{x}_i$ $(1 \le i \le n)$ has been drawn from the k^{th} component or from another one and they are treated as unknown parameters.

Our main concern is to identify the origin of each sample within a class; therefore we will adopt the classification log-likelihood approach in this paper. In what follows EM equations for the classification log-likelihood will be derived.

In EM for mixture models, the "complete" data are considered to be $X = (Y,Z)$ where $Y = [\mathbf{y}_1,...,\mathbf{y}_n]$ is observable and $Z = [\mathbf{z}_1,...,\mathbf{z}_n]$ constitutes the "missing data". In other words there exists a finite set of k states, and that each $\mathbf{y}_i$ is associated with an indicator vector $\mathbf{z}_i$ of length k whose components are all zero except for one indicating the unobserved state associated with $\mathbf{y}_i$.

Let $\mathbf{z}_i = [z_{i1},...,z_{iK}]$ with $z_{ik} = 1$ if $\mathbf{y}_i$ belongs to component k and $z_{ik} = 0$ otherwise and each $\mathbf{z}_i$ be independent and identically distributed according to a multinomial distribution of one draw on K categories with probabilities $\pi_1,...,\pi_K$ and $\mathbf{y}_i$ given $\mathbf{z}_i$ be independent and identically distributed. Then assuming $\mathbf{z}_i$ is associated with $\mathbf{y}_i$ and $\mathbf{y}_i$ belongs to component j, the probability of $\mathbf{z}_i$ becomes

$$f(\mathbf{z}_i) = \frac{1!}{0!\ldots0!1!0!\ldots0!}\,\pi_1^{0}\ldots\pi_{j-1}^{0}\,\pi_j^{1}\,\pi_{j+1}^{0}\ldots\pi_K^{0} = \pi_j \tag{2.3}$$

and the probability of $\mathbf{y}_i$ given $\mathbf{z}_i$ becomes

$$f(\mathbf{y}_i \mid \mathbf{z}_i) = f_j(\mathbf{y}_i \mid \mathbf{m}_j, \Gamma_j) \tag{2.4}$$

Combining (2.3) and (2.4) to obtain $f(\mathbf{x}_i)$ yields

$$f(\mathbf{x}_i) = f_j(\mathbf{y}_i \mid \mathbf{m}_j, \Gamma_j)\pi_j \tag{2.5}$$

Suppose j is unknown, since $z_{ik} = 1$ if $\mathbf{y}_i$ belongs to component k and $z_{ik} = 0$ otherwise (2.5) can be generalized as,

$$f(\mathbf{x}_i) = \prod_{k=1}^{K} [f_k(\mathbf{y}_i \mid \mathbf{m}_k, \Gamma_k)\pi_k]^{z_{ik}} \tag{2.6}$$

For all i=1,…,n (2.6) can be written as,

$$f(X) = \prod_{k=1}^{K}\prod_{i=1}^{n} [f_k(y_i \mid \mathbf{m}_k, \Gamma_k)\pi_k]^{z_{ik}} \tag{2.7}$$

Finally, the resulting complete-data log likelihood is

$$L(\Theta, Z \mid X) = \sum_{k=1}^{K}\sum_{i=1}^{n} z_{ik}[\log \pi_k f_k(\mathbf{x}_i \mid \mathbf{m}_k, \Gamma_k)] \tag{2.8}$$

The quantity z_{ik} (i=1,…,n and k=1,…,K) for equation (2.8) can be estimated as the conditional expectation of z_{ik} given the observation $\mathbf{x}_i$ and the parameter set Θ.

The EM algorithm iterates between an E-step in which values of z_{ik} are estimated from the data with the current parameter estimates as $\hat{z}_{ik}$, and an M-step in which the complete-data log likelihood (2.8), with z_{ik} replaced by its current conditional expectation $\hat{z}_{ik}$, is maximized with respect to the parameters. The outline of the algorithm is given in Figure 2.1. Under fairly weak regularity conditions [9], the method can be shown to converge to a local maximum of the classification likelihood.

The initial partitioning of the data has a crucial importance on the output of the EM stage, as EM will converge to a local maximum, which will be in the neighborhood of the starting point.

The discrete partitioning algorithms by which the initialization is done can be grouped into two types: The hierarchical and nonhierarchical approaches. The fact that

initialize $\hat{z}_{ik}$ (This is done through the output of nearest-mean clustering)

repeat

M-step: maximize (2.8) given $\hat{z}_{ik}$ ($f(\mathbf{x} \mid \theta, \Sigma)$ is a Gaussian density)

$$\hat{n}_k = \sum_{i=1}^{n} \hat{z}_{ik} \tag{2.9}$$

$$\hat{\pi}_k = \frac{\hat{n}_k}{n} \tag{2.10}$$

$$\hat{\mathbf{m}}_k = \frac{\sum_{i=1}^{n} \hat{z}_{ik} \mathbf{x}_i}{\hat{n}_k} \tag{2.11}$$

$$\hat{\Sigma}_k = \frac{\sum_{i=1}^{n} \hat{z}_{ik} (\mathbf{x}_i - \mathbf{m}_k)(\mathbf{x}_i - \mathbf{m}_k)^T}{\hat{n}_k} \tag{2.12}$$

E-step: compute $\hat{z}_{ik}$ given the parameter estimates from the M-step.

$$\hat{z}_{ik} = \frac{\pi_k f_k(\mathbf{x}_i \mid \hat{\mathbf{m}}_k, \hat{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j f_j(\mathbf{x}_i \mid \hat{\mathbf{m}}_j, \hat{\Sigma}_j)} \tag{2.13}$$

until convergence criterion is satisfied

Fig. 2.1 The EM algorithm for normal mixture models.

initialization is not required makes the hierarchical methods very appealing at first sight. Some hierarchical methods even guarantee global optimality. The *Branch and Bound* algorithm [10] is of this type. Both the global optimality and a good initialization is what we are looking for. However, there is a major drawback of hierarchical methods, which makes them very impractical to use in some cases. That is, the time required for the

algorithm to converge increases rapidly with the number of samples and the dimensionality of the data.

The scattering of the data is another negative factor on the performance of some of the hierarchical algorithms. Although some efficient implementation techniques have recently been developed, hierarchical algorithms are still far from accommodating a few thousand samples in hyperdimensional space. Hyperdimensionality is an inherent characteristic of hyperspectral data analysis, and an efficient identification of subclasses usually requires a considerable number of training samples. For this reason, a hierarchical approach is not adopted in this work.

We use nearest mean clustering to initialize the EM algorithm. Compared to hierarchical methods, nearest-mean clustering is very efficient in computational time but does not guarantee a convergence to a global optimum point or a convergence at all and requires an initialization for itself. The initialization method we use is data dependent, which consists of initializing the centers of each cluster using principal components. For well-separated data, a shift in the initial cluster centers will not affect the result of the algorithm. However as the number of samples increases and the data becomes more scattered the algorithm is likely to produce a different output when started with a different set of cluster centers.

2.3 Model Based Mixture Identification

As stated earlier, in hyperspectral analysis the performance of the classifier is highly restricted by the number of training samples available and due to the high dimensional nature of the data. Not surprisingly, this restriction is more severe in the mixture classifier than in any simple quadratic classifier. By partitioning the already small set of training data into multiple clusters and then estimating the cluster statistics, one ends up with a mixture model whose component statistics and thus overall statistics are ill-conditioned unless each cluster has at least d+1 samples, where d is the number of dimensions. Assuming each class of data is to be partitioned into a designated number of clusters K, we should have at least K(d+1) training samples for each class and these

samples should be partitioned evenly among clusters when a clustering algorithm is run. This is practically hard to achieve, let alone having K(d+1) training samples for each class. Therefore the EM equations derived in the previous section based on the *unconstrained* covariance model is by itself not efficient.

One way to get around this problem is to use regularized covariance estimators based on leave-one-out likelihood (LOOL). LOOC introduced in [11], BLOOC introduced in [12], and MIXED-LOOC [13] are of this type. For a quadratic classifier, these estimators examine the sample covariance and the common covariance estimates, as well as their diagonal (LOOC) or trace forms (BLOOC), to determine which would be most appropriate. The same idea can be used in mixture classifiers by applying these estimators to each of the individual class training data. However in mixture identification the main emphasis is on model fit. That is to say, statistics estimation is not the only concern as in simple quadratic classifiers. Therefore the success of LOOL based estimators would be very limited in mixture classifiers unless statistics estimation is accompanied with EM.

On the other hand, when working with EM the structure of the covariance matrix is needed as *a priori* information to avoid possible and unexpected breakdowns of the algorithm. Unfortunately LOOL based estimators do not provide this information, as the covariance matrix obtained this way need not necessarily be unconstrained. When a diagonal, a trace, or a common covariance form is favored alone by these estimators, and equation (2.12) is used to update the covariance matrices, structure of these matrices cannot be preserved. As a result, the statistics estimation may change substantially during the EM stage.

In this chapter we will propose a mixture classifier whose core component is the model based mixture identification. The new approach will eliminate the above problems to a greater extent. In this framework the covariance matrix of each cluster is assumed to be a weighted mixture of constrained and unconstrained covariance matrices. The proposed covariance estimator has the following form:

$$C_{jk}(\Sigma_{jk}) = (1 - \alpha_{jk})\Sigma_{jk} + \alpha_{jk}\Sigma_j \tag{2.14}$$

where $0 \le \alpha_{jk} < 1$, $1 \le j \le N$, $1 \le k \le m_j$, Σ_{jk} is the unconstrained covariance of cluster k in class j, α_{jk} is the corresponding mixing parameter, m_j is the number of clusters in class j, and N is the total number of classes. Note that, m_j can vary from one up to a designated number K. In the above formulation, Σ_j is the unknown covariance structure of class j, which is to be chosen among six possible covariance models through a process involving clustering, EM and model selection.

In what follows, EM equations for the six covariance models considered in this paper will be derived. Note that a change in the covariance model only affects the update equation for the covariance matrix. The rest of the equations in Figure 2.1 remain the same. Before passing into the derivations, two additional matrices will be defined. These are the within class scatter matrix W_j estimated by,

$$\hat{W}_j = \sum_{k=1}^{m_j} \sum_{i=1}^{n_j} \hat{z}_{ik}(\mathbf{x}_i - \mathbf{m}_{jk})(\mathbf{x}_i - \mathbf{m}_{jk})^T \tag{2.15}$$

and the within cluster scatter matrix W_{jk} estimated by,

$$\hat{W}_{jk} = \sum_{i=1}^{n_j} \hat{z}_{ik}(\mathbf{x}_i - \mathbf{m}_{jk})(\mathbf{x}_i - \mathbf{m}_{jk})^T \tag{2.16}$$

where n_j is the number of samples and m_j is the number of clusters in class j and $\hat{z}_{ik}$, $\mathbf{m}_{jk}$ are estimated through the update formulas in Figure 2.1. For a multivariate quadratic case, equation (2.8) can be expanded as follows:

$$L_j(\Sigma_{j1}, \ldots, \Sigma_{jm_j} \mid X) = \sum_{k=1}^{m_j} \sum_{i=1}^{n_j} -\hat{z}_{ik}(\log|\Sigma_{jk}| + \mathrm{tr}[(\mathbf{x}_i - \hat{\mathbf{m}}_{jk})(\mathbf{x}_i - \hat{\mathbf{m}}_{jk})^T \Sigma_{jk}^{-1}] + c) \tag{2.17}$$

where c is a constant with respect to Σ_k and tr(.) is the trace operator. After substituting $\hat{W}_{jk}$ and $\sum_{i=1}^{n_j} \hat{z}_{ik} = \hat{n}_{jk}$ into (2.17), we end up with the following equation.

$$L_j(\Sigma_{j1}, \ldots, \Sigma_{jm_j} \mid \Sigma_j) = \sum_{k=1}^{m_j} -(\hat{n}_{jk}\log|\Sigma_{jk}| + \mathrm{tr}[\hat{W}_{jk}\Sigma_{jk}^{-1}] + c) \tag{2.18}$$

Covariance models:

1. Unconstrained case:

No restriction is placed on the covariance matrices Σ_{jk}. The update equation for this case is already given in (2.12). Maximizing (2.18) leads to the minimization of

$$L_j(\Sigma_{j1},K,\Sigma_{jm_j} \mid \pi_j) = \sum_{k=1}^{m_j} (\hat{n}_{jk} \log|\Sigma_{jk}| + tr[\hat{W}_{jk}\Sigma_{jk}^{-1}] + c) \tag{2.19}$$

and the covariance matrices Σ_{jk} are estimated by

$$\hat{\Sigma}_{jk} = \frac{1}{\hat{n}_{jk}}\hat{W}_{jk} \tag{2.20}$$

2. $\Sigma_{jk} = \sigma_j^2 I$

All the clusters have the same spherical covariance matrix. In this situation, maximizing (2.18) leads to the minimization of

$$L_j(\sigma_j^2 \mid X_j) = n_j d \log(\sigma_j^2) + \frac{1}{\sigma_j^2} tr[\hat{W}_j] + c \tag{2.21}$$

and we get

$$\sigma_j^2 = \frac{tr[\hat{W}_j]}{n_j d} \tag{2.22}$$

3. $\Sigma_{jk} = \sigma_{jk}^2 I$

Clusters are spherical with different volumes. In this situation maximizing (2.18) leads to the minimization of

$$L_j(\sigma_{j1}^2,K,\sigma_{jm_j}^2 \mid \pi_j) = d\sum_{k=1}^{m_j} \hat{n}_{jk} \log(\sigma_{jk}^2) + \sum_{k=1}^{m_j} \frac{1}{\sigma_{jk}^2} tr[\hat{W}_{jk}] + c \tag{2.23}$$

and we get

$$\sigma_{jk}^2 = \frac{tr[\hat{W}_{jk}]}{n_{jk} d} \tag{2.24}$$

4. $\Sigma_{jk} = D_j$

All the clusters have the same diagonal covariance matrix. In this situation maximizing (2.18) leads to the minimization of

$$L_j(D_j \mid X_j) = n_j \log(\mid D_j \mid) + tr[\hat{W}_j D_j^{-1}] + c \tag{2.25}$$

and we get

$$D_j = \frac{diag(\hat{W}_j)}{n_j} \tag{2.26}$$

5. $\Sigma_{jk} = D_{jk}$

All the clusters have different diagonal covariance matrices. In this situation maximizing (2.18) leads to the minimization of

$$L_j(D_{j1}, \ldots, D_{jm_j} \mid X_j) = \sum_{k=1}^{m_j} (\hat{n}_{jk} \log(\mid D_{jk} \mid) + tr[\hat{W}_{jk} D_{jk}^{-1}] + c) \tag{2.27}$$

and we get

$$D_{jk} = \frac{diag(\hat{W}_{jk})}{n_{jk}} \tag{2.28}$$

6. $\Sigma_{jk} = \Sigma_j$

All the clusters have the same covariance matrix. In this situation maximizing (2.18) leads to the minimization of

$$L_j(\Sigma_j \mid X_j) = \hat{n}_j \log(\mid \Sigma_j \mid) + tr[\hat{W}_j \Sigma_j^{-1}] + c \tag{2.29}$$

and we get

$$\Sigma_j = \frac{\hat{W}_j}{n_j} \tag{2.30}$$

For a fixed number of clusters k, the training data for each class are fit into each of the six models through clustering and EM. Then, corresponding to each model, a measure of fit is computed and the model with the highest measure of fit is chosen as Σ_j. Once thestructure of the mixture is determined, the next step is to estimate the mixing proportions, π_{jk} which can be done by maximizing the LOOL for each cluster. This process is repeated by incrementing the number of clusters, k by one up to a designated number K, and a measure of fit is computed at each stage. Finally the mixture model with the highest measure of fit is chosen as the best fit.

Computing a measure of fit for a few thousand samples in a hyperdimensional space is computationally very challenging if not impossible. For the practicality it

provides in this study we adopt the Bayesian Information Criterion (BIC) [8] to compute the measure of fits. Although regularity conditions for the BIC do not hold for mixture models, there is considerable theoretical and practical support for its use in this context [14], [15]. The closed form expression for BIC is given below.

$$2L_{M_k}(\Box,\hat{\Box}_k)\Box d_{M_k}\log(n) \tag{2.31}$$

where $L_{M_k}(\Box,\hat{\Box}_k)$ is the logarithm of the maximized mixture likelihood for the model and d_{M_k} is the number of independent parameters to be estimated in the model. The term on the right in (2.31) is known as the *penalization term.* It penalizes the complexity of the model and this allows us to compare models with differing parameterizations, differing number of components or both. This is not possible by the log-likelihood alone, which increases as more terms are added to the model. A standard convention for interpreting BIC differences is given in Table 1 and the number of independent parameters for the six models considered in the previous section is given in Table 2.

Table 2.1
Indication of evidence for different BIC values [8].

BIC difference	Evidence
0-2	Weak
2-6	Positive
6-10	Strong
>10	Decisive

Table 2.2
Number of independent parameters (A=kd+k-1, B=d(d+1)/2).

Model	Symbol	Number of parameters
Σ_{jk}	U	A+kB
$\Sigma_{jk} = \Sigma_j I$	TE	A+1
$\Sigma_{jk} = \Sigma_{jk} I$	TV	A+k
$\Sigma_{jk} = D_j$	DE	A+d
$\Sigma_{jk} = D_{jk}$	DV	A+kd
$\Sigma_{jk} = \Sigma_j$	E	A+B

2.4 Experimental Results and Discussion

In this section, several experiments are conducted to test the performance of the proposed mixture classifier (MC) and compare it with that of a simple quadratic classifier (SQC). The simple quadratic classifier is designed by characterizing each class distribution by a normal density and estimating the corresponding parameters using the model-based scheme introduced in this chapter. We examined the performance of both classifiers with varying sizes of training set and dimensionality. For each experiment we randomly choose a portion (r) of the labeled set as the training set. The classifiers are tested using the remaining portion of the labeled set. Each experiment is repeated ten times each time with a different training set, and the average percent testing error rates as well their standard deviations are recorded.

2.4.1 Experiment 1: Flightline C1

The different r values considered for this data set are r=0.005 (~350), r=0.1 (~6950), r=0.2 (~19900), r=0.5 (~34750). The percent errors obtained using the two classifiers are recorded in Table 2.3. The values in parenthesis are the standard deviations. The number of subclasses identified in each case is available in Table 2.4. Finally the classification maps corresponding to the two classifiers for r=0.5 are shown in Figure 2.2 and 2.3.

Table 2.3
Testing error rates obtained for the Flightline C1 data.

	SQC	MC
r=0.005	12.0 (0.8)	11.1 (1.2)
r=0.1	5.0 (0.2)	4.7 (0.1)
r=0.2	4.9 (0.2)	4.2 (0.2)
r=0.5	4.7 (0.2)	3.7 (0.1)

Table 2.4
Number of subclasses identified for each class in the Flightline C1 data.

Classes	Number of modes in each class			
	r=0.005	r=0.1	r=0.2	r=0.5
Alfalfa	1.1 (0.3)	1.5 (0.5)	2.0 (0)	3.4 (0.5)
Corns	1.0 (0)	1.0 (0)	1.7 (0.9)	4.4 (0.8)
Rye	1.0 (0)	1.0 (0)	1.0 (0)	1.0 (0)
Oats	1.0 (0)	1.0 (0)	1.3 (0.5)	2.8 (0.4)
Bare Soil	1.0 (0)	1.0 (0)	1.0 (0)	1.3 (0.5)
Wheat	1.0 (0)	2.0 (0)	4.0 (0)	4.0 (0)
Red Clovers	1.0 (0)	1.6 (0.5)	2.3 (0.6)	5.9 (0.3)
Soybeans	1.2 (0)	3.7 (0.9)	5.3 (1.3)	5.5 (0.8)
Water	1.0 (0)	1.0 (0)	1.0 (0)	1.0 (0)

2.4.2 Experiment 2: Washington DC Mall Data

In order to avoid computational complexity and unavoidable numerical issues, for this data set we first reduced the dimensionality by feature extraction. At this point we avoid using parametric feature extraction techniques, as feature information relevant to mixture analysis may be lost when the class densities are assumed as Gaussian and a feature extraction is performed based on this assumption. We use the Nonparametric Weighted Feature Extraction (NWFE) technique recently introduced in [16]. We randomly choose a portion (r) of the labeled samples for each class as training samples. The r values considered for this data set are r=0.02 (430), r=0.1 (2150), r=0.2 (4300),

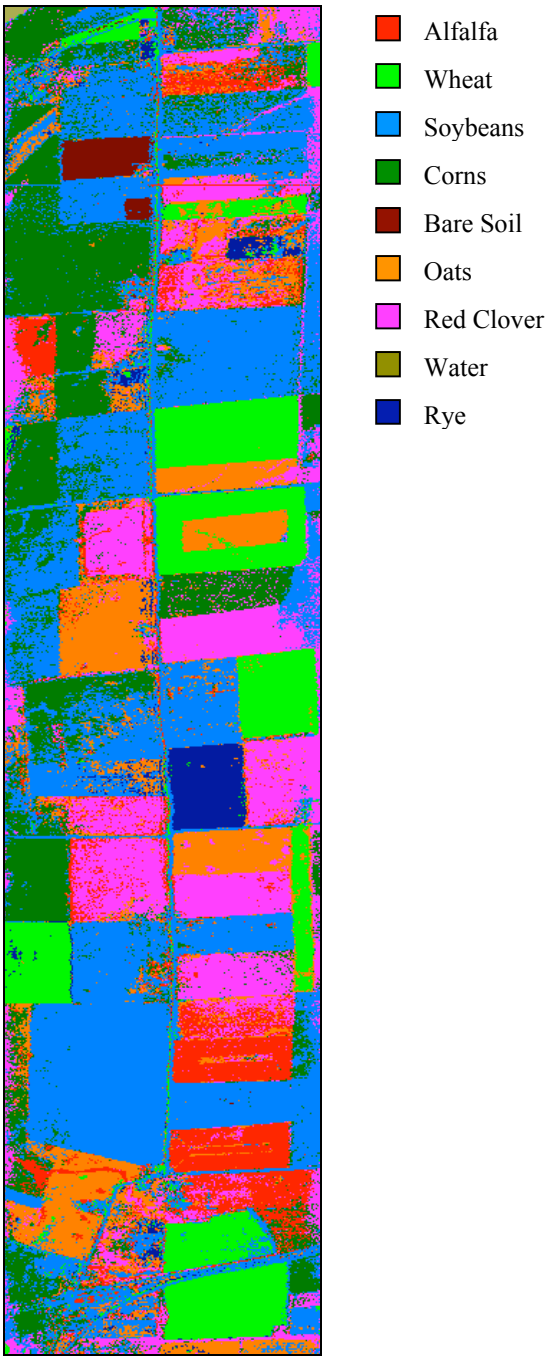


Fig. 2.2 Classification map obtained for the Flightline C1 data by SQC for r=0.5 (% error=5.7).

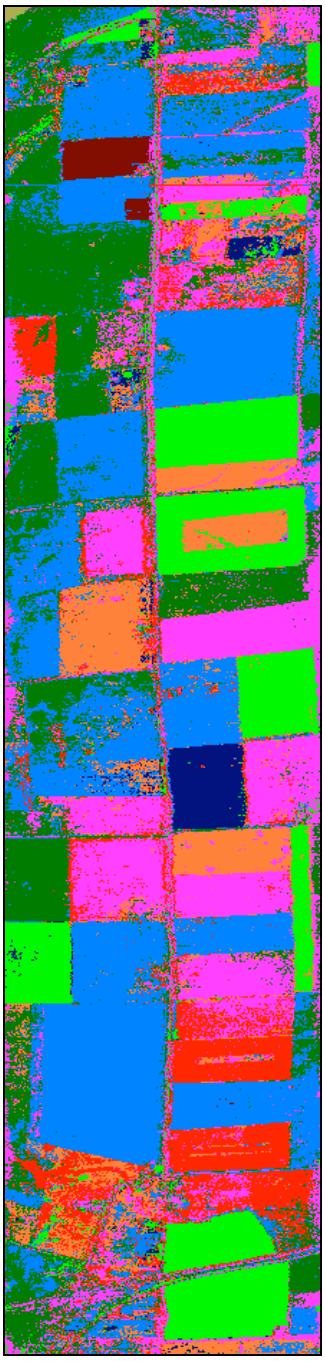


Fig. 2.3 Classification map obtained for the Flightline C1 data by MC for r=0.5 (% error=3.8).

r=0.5 (10750). The feature extraction algorithm is computationally very demanding. Therefore for large r values we used only a small portion of the training data during feature extraction. Then the best 10, 20, 30 and 40 features are selected respectively, and for each case a classification is performed. The classifiers are then tested using the remaining portion of the labeled samples. The percent error rates are shown in Table 2.5 and the numbers of subclasses identified in each class for each r considered are recorded in Table 2.6-2.9. The classification maps obtained for r=0.5 are shown in Figure 2.4 and 2.5.

Table 2.5
Testing error rates obtained for the Washington DC Mall data

	d=10		d=20		d=30		d=40	
	SQC	MC	SQC	MC	SQC	MC	SQC	MC
r=0.02	5.7 (0.4)	5.7 (0.5)	4.5 (0.4)	4.4 (0.4)	5.4 (0.8)	5.3 (0.9)	9.8 (2)	9.8 (2)
r=0.1	6.1 (0.2)	3.3 (0.4)	3.9 (0.3)	2.6 (0.3)	3.5 (0.2)	2.8 (0.4)	3.3 (0.2)	3.1 (0.5)
r=0.2	6.2 (0.4)	2.9 (0.3)	3.6 (0.3)	2.2 (0.2)	3.4 (0.2)	2.0 (0.1)	3.1 (0.2)	2.0 (0.1)
r=0.5	6.3 (0.3)	3.1 (0.3)	3.7 (0.0)	2.1 (0.1)	3.5 (0.0)	2.4 (1.0)	3.3 (0.0)	1.8 (0.1)

Table 2.6
Number of subclasses identified for each class in
Washington DC Mall data for r=0.02

Classes	Number of modes in each class			
	d=10	d=20	d=30	d=40
Roofs	1.0 (0)	1.0 (0)	1.0 (0)	1.0 (0)
Grass	1.3 (0.6)	1.0 (0)	1.0 (0)	1.0 (0)
Roads	1.4 (0.5)	1.2 (0.4)	1.0 (0)	1.0 (0)
Paths	1.0 (0)	1.9 (0.3)	1.1 (0.3)	1.1 (0.3)
Trees	1.0 (0)	1.1 (0.3)	1.4 (0.5)	1.3 (0.5)

Table 2.7
Number of subclasses identified for each class in
Washington DC Mall data for r=0.1.

Classes	Number of modes in each class			
	d=10	d=20	d=30	d=40
Roofs	2.1 (0.3)	1.0 (0)	2.1 (0.3)	1.7 (0.6)
Grass	1.7 (0.5)	1.0 (0)	1.7 (0.5)	1.0 (0)
Roads	3.2 (0.6)	2.0 (0)	1.1 (0.3)	1.0 (0)
Paths	1.6 (0.7)	2.9 (0.8)	1.4 (0.5)	1.0 (0)
Trees	1.3 (0.5)	2.3 (0.6)	2.0 (0)	2.0 (0)

Table 2.8
Number of subclasses identified for each class in
Washington DC Mall data for r=0.2

Classes	Number of modes in each class			
	d=10	d=20	d=30	d=40
Roofs	4.8 (0.4)	1.0 (0)	2.1 (0.3)	2.0 (0)
Grass	2.0 (0)	1.0 (0)	2.0 (0)	1.7 (0.5)
Roads	3.9 (0.5)	2.3 (0.5)	2.1 (0.3)	1 (0)
Paths	3.0 (0.5)	4.7 (1.2)	1.9 (0.3)	1 (0)
Trees	1.9 (0.3)	3.6 (0.9)	2.3 (0.5)	2.2 (0.4)

Table 2.9
Number of subclasses identified for each class in
Washington DC Mall data for r=0.5

Classes	Number of modes in each class			
	d=10	d=20	d=30	d=40
Roofs	7.7 (0.5)	7.6 (0.5)	7.2 (0.8)	7.4 (0.5)
Grass	7.6 (0.7)	5.9 (0.3)	4.0 (0.8)	2.2 (0.4)
Roads	5.7 (2.3)	5.2 (1.3)	3.6 (1.8)	2.8 (0.9)
Paths	4.3 (1.2)	2.9 (0.3)	2.0 (0)	2.0 (0)
Trees	4.3 (0.5)	2.7 (0.5)	2.0 (0)	2.0 (0)

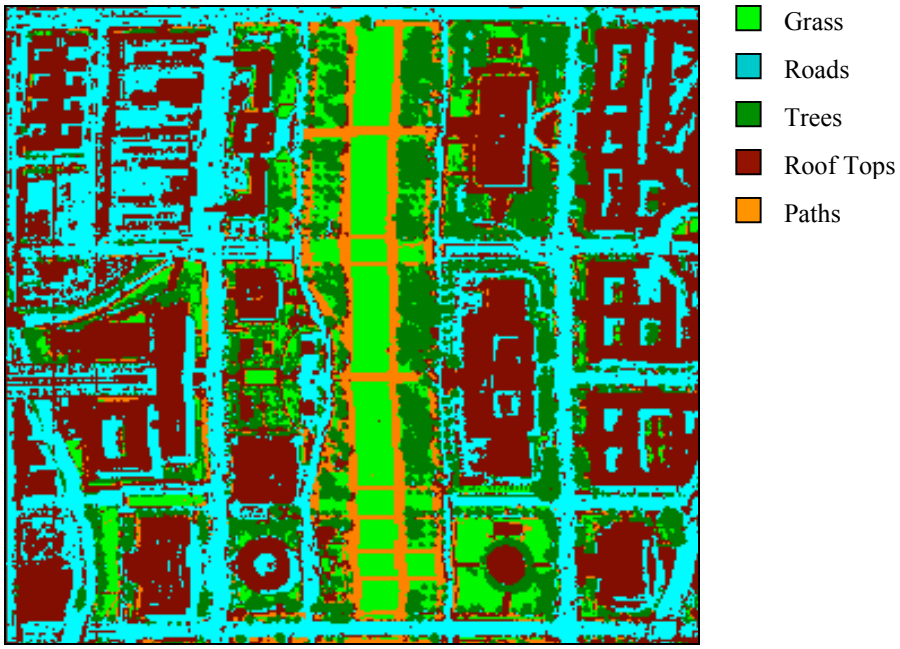


Fig. 2.4 Classification map obtained for the Washington DC Mall data
By SQC for r=0.5 (% error=3.0)

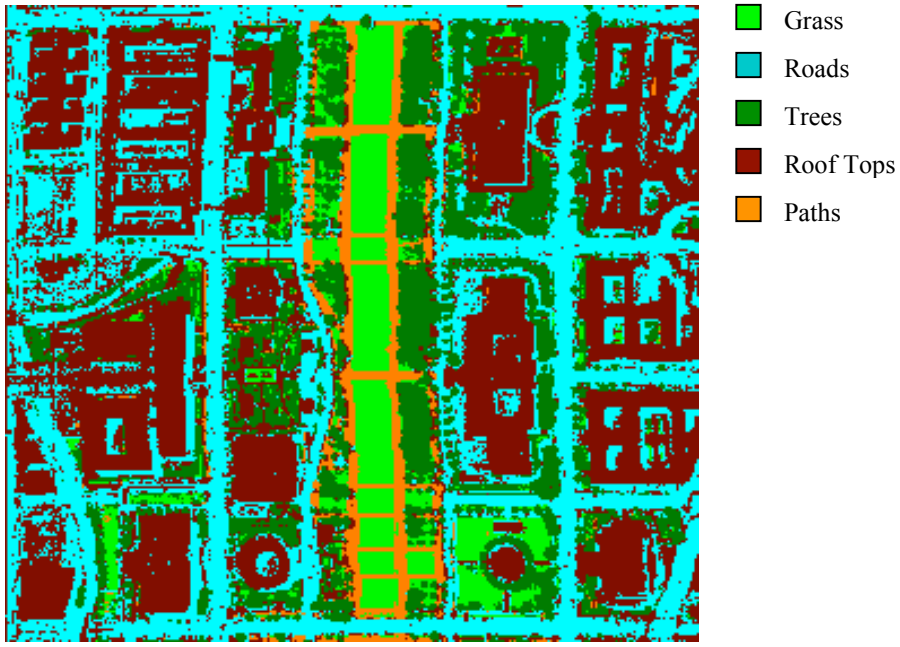


Fig. 2.5 Classification map obtained for the Washington DC Mall data
by MC for r=0.5 (% error=1.6)

2.4.3 Experiment 3: Purdue Campus Data

The classifiers for this data set are designed similar to section 2.4.2. The r values considered are r=0.03 (762), r=0.1 (2155), r=0.2 (4299), r=0.5 (10757). The percent error rates are noted in Table 2.10 and the numbers of subclasses identified in each class for each r considered are recorded in Table 2.11-2.14. The classification maps obtained for r=0.5 are shown in Figure 2.6 and 2.7.

Table 2.10
Testing error rates obtained for the Purdue Campus data.

	d=10		d=20		d=30		d=40	
	SQC	MC	SQC	MC	SQC	MC	SQC	MC
r=0.03	16.2 (2.7)	12.6 (1.5)	12.4 (0.7)	9.0 (0.4)	12.4 (0.4)	12.0 (2.2)	16.3 (0.6)	18.0 (1.2)
r=0.1	16.2 (0.4)	10.9 (0.5)	13.9 (0.3)	8.2 (0.4)	12.9 (0.1)	7.7 (0.2)	13.0 (0.0)	8.6 (0.6)
r=0.2	15.6 (0.5)	10.0 (0.4)	13.8 (0.6)	7.2 (0.1)	12.9 (0.6)	6.6 (0.1)	12.3 (0.3)	6.9 (0.2)
r=0.5	15.5 (0.1)	9.5 (0.1)	14.2 (0.0	6.8 (0.1)	13.6 (0.1)	6.0 (0.1)	13.2 (0.1)	5.9 (0.0)

Table 2.11
Number of subclasses identified for each class in
Purdue Campus data for r=0.03.

Classes	Number of modes in each class			
	d=10	d=20	d=30	d=40
Roofs	3.4 (1.0)	2.2 (0.4)	2.1 (0.3)	1.8 (0.4)
Grass	1.3 (0.5)	1.0 (0)	1 (0)	1.0 (0)
Roads	2.1 (0.3)	1.3 (0.5)	1.1 (0.3)	1.0 (0)
Shadow	1.5 (0.5)	1.0 (0)	1.0 (0)	1.0 (0)
Others	2.0 (0)	1.1 (0.3)	1.0 (0)	1.0 (0)
Paths	1.0 (0)	1.0 (0)	1.0 (0)	1.1 (0.3)
Cars	1.0 (0)	1.0 (0)	1.0 (0)	1.0 (0)
Trees	1.2 (0.4)	1.0 (0)	1.0 (0)	1.0 (0)

Table 2.12
Number of subclasses identified for each class in
Purdue Campus data for r=0.1.

Classes	Number of modes in each class			
	d=10	d=20	d=30	d=40
Roofs	6.1 (1.5)	4.8 (1.7)	4.2 (1.5)	3.2 (0.6)
Grass	2.3 (0.5)	1.0 (0)	1.0 (0)	1.0 (0)
Roads	2.6 (0.5)	2.4 (0.5)	1.3 (0.5)	1.0 (0)
Shadow	1.4 (0.5)	1.0 (0)	1.0 (0)	1.0 (0)
Others	2.0 (0)	1.4 (0.5)	1.1 (0.3)	1.2 (0.4)
Paths	1.1 (0.3)	1.0 (0)	1.0 (0)	1.0 (0)
Cars	1.1 (0.3)	1.0 (0)	1.0 (0)	1.0 (0)
Trees	2.1 (0.3)	1.0 (0)	1.0 (0)	1.0 (0)

Table 2.13
Number of subclasses identified for each class in
Purdue Campus data for r=0.2.

Classes	Number of modes in each class			
	d=10	d=20	d=30	d=40
Roofs	7.7 (0.5)	7.6 (0.5)	6.4 (1.4)	5.0 (1.2)
Grass	2.8 (0.4)	2.2 (0.7)	1.1 (0.3)	1.1 (0.3)
Roads	4.0 (0.9)	2.8 (0.6)	2.5 (0.5)	1.7 (0.5)
Shadow	2.1 (0.5)	1.1 (0.3)	1.0 (0)	1.0 (0)
Others	2.0 (0)	1.2 (0.4)	1.3 (0.5)	1.0 (0)
Paths	1.4 (0.5)	1.1 (0.3)	1.0 (0)	1.0 (0)
Cars	1.9 (0.5)	1.0 (0)	1.0 (0)	1.0 (0)
Trees	2.3 (0.5)	1.8 (0.4)	1.1 (0.3)	1.0 (0)

2.4.4 Discussion and analysis of results

For the simple quadratic classifier we would normally expect higher classification accuracy as the number of training samples increases. This is true to a certain extent. For a small set of training data a single normal distribution characterizes class densities within a good accuracy, therefore the performance of the simple quadratic classifier is comparable to that of a mixture classifier. However as the training data set enlarges,

Table 2.14
Number of subclasses identified for each class in
Purdue Campus data for r=0.5.

Classes	Number of modes in each class			
	d=10	d=20	d=30	d=40
Roofs	8.0 (0)	8.0 (0)	7.9 (0.3)	7.9 (0.3)
Grass	4.1 (0.9)	3.0 (0.4)	2.7 (0.5)	1.2 (0.4)
Roads	6.3 (1.1)	5.1 (0.9)	3.2 (0.4)	2.9 (0.3)
Shadow	2.5 (0.5)	2.0 (0)	1.8 (0.4)	1.0 (0)
Others	2.0 (0)	2.0 (0)	1.3 (0.5)	1.0 (0)
Paths	3.2 (0.4)	1.9 (0.3)	1.4 (0.5)	1.0 (0)
Cars	2.8 (0.4)	2.0 (0)	1.2 (0.4)	1.0 (0)
Trees	3.2 (0.4)	2.6 (0.5)	2.0 (0)	1.8 (0.4)

the simple quadratic characterization of class densities becomes more challenging, and the error between actual and estimated class densities increases. When this error exceeds a certain threshold the classifier performance deteriorates. Therefore a large set of training data does not necessarily mean a better classifier performance for a simple quadratic classifier.

On the other hand when a mixture density is used to estimate the class densities, this limitation is mitigated to a greater extent. As a matter of fact the larger the training data set, the more accurately the densities are estimated and hence the better the performance of the mixture classifier.

When the classification maps corresponding to both classifiers are compared one can notice that the speckle error in classification maps obtained by the mixture classifiers is slightly less than those obtained by the simple quadratic classifier. Apart from the scatter errors perhaps what is more significant in those thematic maps are the fact that some fields which are misclassified by the simple quadratic classifier are correctly classified by the mixture classifier. This difference is more dramatic especially for the campus data. To sum up, the improvements the quantitative error rates suggest comes in the form of both reduced scatter and classification errors. Indeed some of the fields that are correctly classified by the mixture classifier but misclassified by the simple quadratic

classifier are not included in the test field map. This implies the actual improvements might be slightly more than what the quantitative values suggest.

As the dimensionality increases we first get a lower rate but after a certain point the Hughes phenomena [17] comes into play and we see some deteriorations in the performances of both classifiers. Compared to the unconstrained mixture model the proposed technique significantly reduces the number of parameters to estimate. However the number of parameters for a mixture model with k components might still be greater than the number of parameters that needs to be estimated in a simple quadratic classifier. From that respect the mixture classifier is more prone to Hughes phenomena. Finally note that for small r both classifiers suffer from the scarcity of training data.

2.5 Conclusions

This technique significantly reduces the number of parameters to estimate by constraining the covariance matrix of each component to a less sophisticated covariance model. For the data we investigated we have obtained significant improvements over the simple quadratic classifier especially when the training sample size was larger.

One issue we were unable to address was to estimate the number of subclasses for a given class in the presence of limited labeled samples. This was partly due to the fact that the criterion we used to determine the best fit was only a crude approximation to a more sophisticated measure. But most importantly the normal mixture model was not competent enough to efficiently characterize class distributions in the presence of few labeled samples. Usually when the labeled samples are scarce it is hard to determine the number of components needed to efficiently estimate the class distribution. One can avoid this problem by assigning an excessive number of components for each class but in that case statistics estimation becomes extremely impractical.

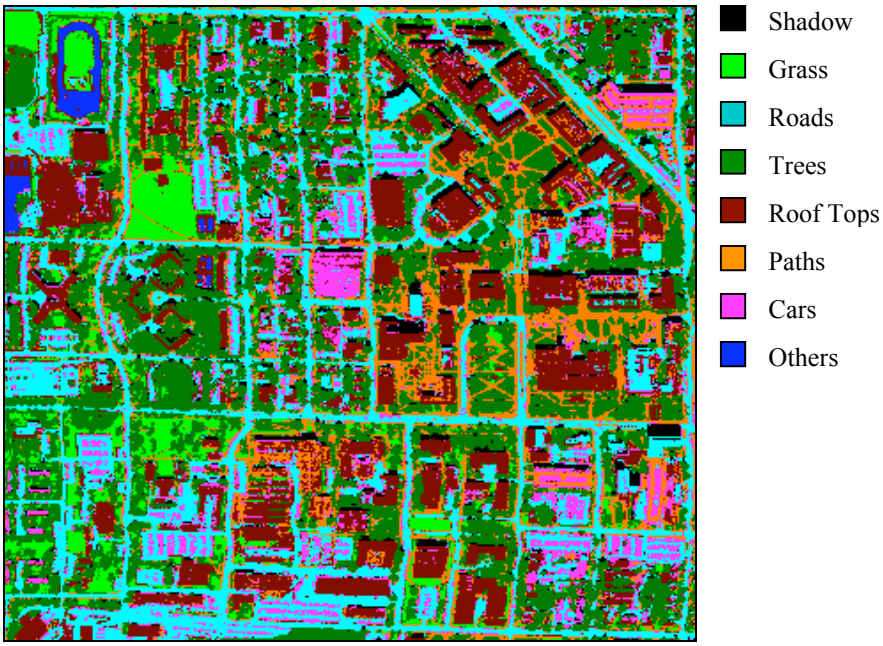


Fig. 2.6 Classification map obtained for the Purdue Campus data by SQC for r=0.5 (% error=15.4)

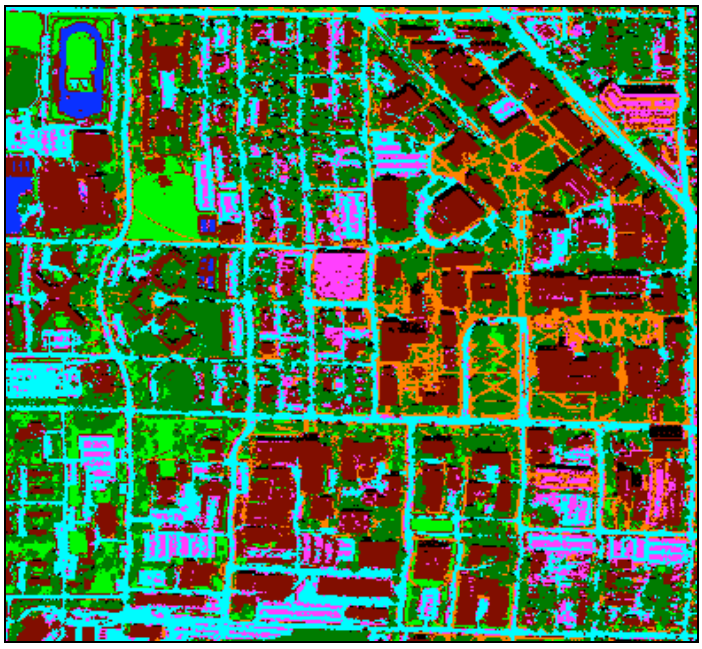


Fig. 2.7 Classification map obtained for the Purdue Campus data by MC for r=0.5 (% error=6.0).

Apart from the limited training size problem, which is one of the most basic problems of hyperspectral data analysis, there are some other stages in the design of the mixture classifier that requires further attention.

One of these is the model selection stage. The Bayesian Information Criterion doesn't provide reliable information on covariance structure when the number of samples is very small. As a matter of fact, it will overestimate the number of components and introduce redundant parameters that will make estimations less reliable. A possible solution might be to use a common covariance for all classes and update the EM equations accordingly.

Initial partitioning of the data is another stage that needs attention, as EM will converge to a local optimum that is closest to the initial starting point. We use nearest mean clustering to initialize EM but more efficient initializations can be found by trying different clustering algorithms.

Although there are several studies that address some of the above concerns within the domain of normal mixture models the mixture model might not be the most competent approach to deal with multi-mode class distributions. In the next chapter we will introduce a supervised classifier approach which addresses most of our concerns in this chapter to a greater extent and which will make us think we are moving toward obtaining an optimal classifier for the analysis of hyperspectral data.

3. TOWARD AN OPTIMAL SUPERVISED CLASSIFIER

3.1 Introduction

It is well known that in supervised classification problems the probability of error due to a Bayes classifier is the best one can achieve. The Bayes classifier compares the a posteriori probabilities of all classes and assigns the sample to the class with the highest probability. However for most classes of distributions designing a Bayes classifier is very difficult if not impractical. The most common way across this problem is to assume normal distributions for all classes. This way one arrives at a simple quadratic classifier which has proved effective in classification problems of interest to many different disciplines. The problem can be simplified even further by assuming equal covariance matrices for all classes. In this case the resulting decision boundary is linear. A linear classifier is often very desirable in that it is less prone to noise and most likely will not overfit.

Although by assuming normal distributions for all classes one can design a classifier based on Bayes rule this classifier will not be optimal unless the actual class distributions are normal. Clearly for most real-world data the normality assumption does not hold in practice. A single normal distribution is not flexible enough to capture complex data structures encountered in real-world settings.

One widely accepted way to mitigate this problem is to model each class data by a mixture of normal distributions [7, 18, 19, 20, 21]. The normal mixture model, which is the sum of one or more weighted normal components, combines much of the flexibility of nonparametric methods with certain of the analytic advantages of parametric methods. Under fairly weak conditions and given enough components, a mixture model can approximate a given density arbitrarily closely allowing great flexibility.

In Chapter 2 we proposed a model-based mixture-supervised classification approach for the analysis of hyperspectral data. In this approach we used normal mixture models to characterize class distributions by incorporating a model-based covariance estimator into the process of estimating the number of normal components and class statistics. Our proposed covariance estimator is a weighted sum of constrained and unconstrained covariance matrices. The constrained part is chosen among six other covariance models through clustering, expectation-maximization and model selection whereas the unconstrained part is the maximum likelihood estimate of the covariance matrix. This model-based approach, which utilizes the common and sample covariance matrices as well as their trace and diagonal forms, not only helps to identify the underlying class structure, providing better characterization but also produces robust estimates of component matrices. For the data we investigated we have obtained considerable improvements over the simple quadratic classifier especially when the number of training samples is large. However when the amount of training data is very limited the performance of the proposed classifier suffers from insufficient information about the underlying structures of class distributions. This poses a problem in identifying the number of subclasses and obtaining reliable estimates of their statistics.

A relatively new approach for characterizing class data is through kernel functions. The kernel concept was first applied in Support Vector Machines [22-24] and has since become common in the field of computational learning [25-28]. The underlying strength of this idea can be summarized as follows. Any linear algorithm which can be carried out in terms of dot products can be made nonlinear by substituting a dot product with an a priori chosen kernel operator k satisfying the Mercer theorem. This corresponds to mapping the data into a possibly high dimensional feature space F, by a nonlinear map $\Phi: R^d \rightarrow F$ and taking the dot product there, i.e. $k(\mathbf{x}, \mathbf{x}') = \Phi(\mathbf{x})^T \Phi(\mathbf{x}')$ where $\mathbf{x}, \mathbf{x}' \in R^d$. The kernel concept gives us an elegant way of implementing nonlinear algorithms in the input space without ever mapping the data explicitly to F [28].

This is a major breakthrough in that it allows us to accommodate complex data structures without sacrificing the simplicity of the classifier model. Indeed as simple as a linear classifier in the feature space corresponds to a powerful classifier with highly

nonlinear boundaries in the input space and one can easily adjust the complexity of this classifier by choosing a suitable kernel function among a large class of kernels that can be used.

Despite its widespread acceptance in various different disciplines the use of kernel-based methods in the remote sensing literature is quite limited [29-31].

In this study we will propose a supervised classifier based on implementation of Bayes rule with kernels for the analysis of hyperspectral data. The proposed technique first suggests an implicit nonlinear transformation of the data into some feature space and seeks to fit normal distributions into the mapped data. A key outstanding problem of this approach is the computation of posterior class probabilities. The posterior probabilities can not be fully expressed in dot-product form due to the presence of determinants. In order to get rid of these incomputable terms we further assume equal covariance matrices for all classes. A similar assumption is made in [26] to obtain a kernelized version of expectation-maximization and some encouraging results are obtained within the framework of that paper. Since the determinants now cancel out, posterior probabilities can be evaluated, and a linear classifier can be designed. At first sight it might seem unwise to further constrain the complexity of the classifier model by assuming equal covariance matrices for each class. However in return we are now able to kernelize the algorithm. This is a gain which will ultimately provide us more control over the complexity of the classifier model. By using different kernel functions and varying the parameters of a kernel function we can model a large variety of class conditional distributions.

The rest of this chapter is organized as follows. In the next section we review the theory of the Bayes decision rule for multiple hypotheses and discuss some of the problems that arise when the Bayes decision rule is implemented by normal models. In the third section we propose a novel technique for characterizing class conditional distributions using kernel functions and discuss why this technique can be a better alternative to normal and normal-mixture models. Finally we perform several experiments to test the performance of the proposed classifier and compare this with those of linear, simple quadratic and mixture classifiers.

3.2 The Bayes Decision Rule

Suppose each class data is distributed according to an unknown distribution function $f_k(\mathbf{x} \mid \theta_k)$ with prior probability π_k such that $\sum_{k=1}^{K} \pi_k = 1$, where θ_k is the unknown set of parameters and $0 \le \pi_k \le 1$, $1 \le k \le K$. We denote the posterior probability of class w_k given a d-dimensional vector $\mathbf{x}$ by $p_k(w_k \mid \mathbf{x})$. Then the decision rule that minimizes the classification error for any given sample $\mathbf{x}$ is

$$p_j(w_j \mid \mathbf{x}) = \max_k p_k(w_k \mid \mathbf{x}) \quad \Rightarrow \quad \mathbf{x} \in w_j \tag{3.1}$$

Since by the Bayes rule,

$$p_k(w_k \mid \mathbf{x}) = \frac{f_k(\mathbf{x} \mid \theta_k)\pi_k}{\sum_{k=1}^{K} f_k(\mathbf{x} \mid \theta_k)\pi_k} \tag{3.2}$$

(3.1) can be rewritten as

$$f_j(\mathbf{x} \mid \theta_j)\pi_j = \max_k f_k(\mathbf{x} \mid \theta_j)\pi_k \quad \Rightarrow \quad \mathbf{x} \in w_j \tag{3.3}$$

The initial problem of evaluating the *a posteriori* probability of class w_k given sample $\mathbf{x}$ now reduces to finding the likelihood of $\mathbf{x}$ coming from w_k.

In most real-world settings both f_k, θ_k and π_k are unknown and need to be estimated from a finite set of labeled samples. The most common way to deal with this problem is to assume a generic model for f_k and then estimate its parameters using labeled samples belonging to class w_k. This way one can obtain more reliable estimates with less bias and variance than when the same process is carried out without any specific assumption about the structure of the underlying class distribution.

For the simplicity it provides, the normal model has long been the mainstream choice for f_k in the literature. Compared to other parametric distributions, the normal model is usually more efficient in accommodating real-world data and can easily be characterized by two parameters, namely its mean and covariance. By assuming a normal model and estimating its mean and covariance using labeled samples one can obtain a closed form expression for f_k and arrive at a simple quadratic classifier by

evaluating the decision rule in (3.2). Although the simple quadratic classifier has proved efficient in some moderately challenging problems this scenario does not always give us satisfactory results.

It is well-known that not all real-world data can be sufficiently closely approximated by a normal model. The normal model usually lacks the flexibility and complexity required to accommodate complex data structures, e.g. data with multiple subclasses. When the generic model used to implement the decision rule in (3.2) differs from the actual model the classification error obtained might be substantially higher than the minimum achievable error. This problem is certainly not new. Indeed the literature is full of attempts to replace the normal model with more flexible models. One such attempt is the normal mixture model.

The normal mixture model is the sum of one or more weighted normal components. It is flexible because under fairly weak conditions and given enough components, a mixture model can approximate any given density arbitrarily closely. It is practical because we maintain the simplicity of the normal model. Since each class data is modeled by a mixture of normal distributions the posterior probabilities can still be computed with great ease.

Several studies reported [7, 18-21] shows that a normal mixture model can successfully replace the normal model when there are enough labeled samples to identify the right number of subclasses and to obtain reliable estimates of class statistics. Note that estimation of class statistics becomes more challenging with normal mixture models, because using the same number of labeled samples one must estimate mean vectors, covariance matrices and weighting factors of all the components associated with a class whereas in the normal model the parameters to estimate are limited to the mean vector, covariance matrix, and prior probability of that class. This certainly poses a problem when dealing with high dimensional data.

In [20] we proposed a classifier where each class of data is modeled by a normal mixture model, and covariance matrices corresponding to each component are estimated by a model-based covariance estimator. This technique significantly reduces the number of parameters to estimate by constraining the covariance matrix of each

component to a less sophisticated covariance model. For the data we investigated we obtained significant improvements over the simple quadratic classifier especially when the number of training sample size was large. One issue we were unable to address was to estimate the number of subclasses for a given class in the presence of limited labeled samples. This was partly because the criterion used to determine the best fit was only a crude approximation to a more sophisticated measure, but most importantly because the normal mixture model was not competent enough to efficiently characterize class distributions in the presence of limited labeled samples. Usually when labeled samples are scarce, it is hard to determine the number of components needed to efficiently estimate the class distribution. One can avoid this problem by assigning an excessive number of components for each class, but in that case statistics estimation becomes extremely impractical.

In the next section we will propose a novel technique that addresses the above problems associated with normal models as well as normal mixture models to a greater extent.

3.3 Normal Model in the Feature Space

We assume an implicit transformation of the data into some feature space F by a nonlinear map $\Phi: R^d \to F$ and assume a normal model for all class conditional distributions $f_k(\Phi(\mathbf{x}) \mid \theta_k)$. Let $\Phi_k = [\Phi(\mathbf{x}_1^k), \ldots, \Phi(\mathbf{x}_{l_k}^k)]$ be the labeled samples belonging to class w_k with $\Phi(\mathbf{x}_i^k) \in F$ and $\sum_{k=1}^{K} l_k = l$. By definition the prior probability, the mean vector and the covariance matrix of class w_k are

$$\pi_k = \frac{l_k}{l} \tag{3.4}$$

$$\mathbf{m}_k^{\Phi} = \frac{1}{l_k} \Phi_k \mathbf{1}_{n_k} \tag{3.5}$$

$$\Sigma_k^{\Phi} = \frac{1}{l_k} \underbrace{(\Phi_k - \mathbf{m}_k^{\Phi} \mathbf{1}_{n_k}^T)(\Phi_k - \mathbf{m}_k^{\Phi} \mathbf{1}_{n_k}^T)^T}_{\bar{\Phi}_k \bar{\Phi}_k^T} \tag{3.6}$$

where $\mathbf{1}_{l_k}$ is an l_k-dimensional vector of ones and Φ is used to distinguish the parameters in the feature space from their input-space counterparts. In order to evaluate the decision rule (3.1) in F we need to find the *a posteriori* probabilities of all classes for any given sample $\Phi(\mathbf{x})$. In doing this we will greatly benefit from the kernel concept. Here the kernel concept will allow us to efficiently compute the *a posteriori* probability of a given class in the input space without ever mapping the data into the feature space once we can express equation (3.2) in the dot-product form. Unfortunately equation (3.2) can not be fully expressed in the dot-product form unless we further constrain the normal model by assuming equal covariance matrices for all classes. That is $\Sigma_1^{\Phi} = \Sigma_2^{\Phi} = \ldots = \Sigma_K^{\Phi} = \Sigma^{\Phi}$. This way the determinants in the numerator and denominator of equation (3.2) cancels out and since the denominator is equal for all classes the initial problem of evaluating the posterior probability in the feature space reduces to evaluating the following discriminant function for all classes

$$g_k(\Phi(\mathbf{x})) = (\Phi(\mathbf{x}) - \mathbf{m}_k^{\Phi})^T \Sigma^{\Phi -1} (\Phi(\mathbf{x}) - \mathbf{m}_k^{\Phi}) + \log(\pi_k) \tag{3.7}$$

and the decision rule in (3.1) now becomes

$$g_j(\Phi(\mathbf{x})) = \max_k g_k(\Phi(\mathbf{x})) \quad \Rightarrow \quad \mathbf{x} \in w_j \tag{3.8}$$

Note that (3.7) is still not in the dot-product form as it requires the inverse of the common covariance matrix Σ^{Φ} which is usually impractical if not impossible to compute explicitly. In what follows using the idea introduced within the framework of kernel principal component analysis [23] we will fully express (3.7) in the dot-product form. First we define the common covariance matrix as

$$\Sigma^{\Phi} = \frac{1}{K} \sum_{k=1}^{K} \frac{1}{l_k} \bar{\Phi}_k \bar{\Phi}_k^T \tag{3.9}$$

Let $\mathbf{v}$ be an eigenvector of Σ^{Φ} and λ its corresponding eigenvalue then

$$\Sigma^{\Phi} \mathbf{v} = \lambda \mathbf{v} \tag{3.10}$$

From the theory of reproducing kernels we know that $\mathbf{v}$ can be expressed as an expansion of all the labeled samples,

$$\mathbf{v} = \underbrace{[\Phi(\mathbf{x}_1), \ldots, \Phi(\mathbf{x}_l)]}_{\Phi} \mathbf{a} \tag{3.11}$$

where $\mathbf{a}$ is some l_k-dimensional vector. Substituting (3.9) and (3.11) into (3.10) and multiplying both sides of the resulting equation by Π^T we get

$$\frac{1}{K}\sum_{k=1}^{K}\frac{1}{l_k}\Pi^T\bar{\Phi}_k\bar{\Phi}_k^T\Pi\mathbf{a} = \Pi\Pi^T\Pi\mathbf{a} \tag{3.12}$$

Equation (3.12) can be evaluated by replacing all the dot-products by a kernel function. Let $k(\mathbf{x},\mathbf{x}_i)$ be a kernel function satisfying the Mercer theorem [28] such that $k(\mathbf{x},\mathbf{x}_i) = \Phi(\mathbf{x})^T\Phi(\mathbf{x}_i)$, then we denote the kernel matrix for class w_k as $\mathbf{K}_k = \Phi^T\Phi_k$, the gram matrix as $\mathbf{K} = \Pi^T\Pi$, and express (3.12) in terms of $\mathbf{K}_k$ and $\mathbf{K}$

$$\frac{1}{K}\sum_{k=1}^{K}\frac{1}{l_k}\bar{\mathbf{K}}_k\bar{\mathbf{K}}_k^T\mathbf{a} = \mathbf{G}\mathbf{a} \tag{3.13}$$

where $\bar{\mathbf{K}}_k = \mathbf{K}_k(\mathbf{I}_{l_k} - \mathbf{1}_{l_k}\mathbf{1}_{l_k}^T)$, $\mathbf{I}_{l_k}$ is an l_k-dimensional identity matrix and $\mathbf{1}_{l_k}$ is an l_k-dimensional vector of ones. Although equation (3.13) can be solved using generalized eigenvalue problem solvers the above setting is ill-conditioned. Note that $\mathbf{G}$ is an l-dimensional matrix estimated using l samples. Therefore a small regularization term is added to this matrix to make the algorithm numerically more stable, i.e. $\mathbf{G}_\alpha = \mathbf{G} + \alpha\mathbf{I}_n$.

Now we can express Π^α in terms of all the $\mathbf{a}$'s and λ's that satisfies (3.13). Since $\mathbf{v}$ is related to $\mathbf{a}$ by equation (3.11) we first scale $\mathbf{a}$ such that $\mathbf{v}^T\mathbf{v} = 1$.

$$\bar{\mathbf{a}} = \frac{\mathbf{a}}{\sqrt{\mathbf{a}^T\mathbf{K}\mathbf{a}}} \tag{3.14}$$

Let $\mathbf{a}_1 ... \mathbf{a}_m$ be all the solutions of (3.13) ($m \le n$) with corresponding eigenvalues $\lambda_1,...,\lambda_m$ all greater than zero. Then

$$\Pi^\alpha = \Pi\mathbf{A}\Pi\mathbf{A}^T\Pi^T \tag{3.15}$$

where $\mathbf{A} = [\bar{\mathbf{a}}_1 ... \bar{\mathbf{a}}_m]$ and $\Lambda = \mathrm{diag}(\lambda_1 ... \lambda_m)$. Since $\bar{\mathbf{a}}_i$'s are orthogonal with respect to $\mathbf{K}$: that is,

$$\bar{\mathbf{a}}_i^T\mathbf{K}\bar{\mathbf{a}}_j = \begin{cases} 1 & i = j \\ 0 & i \ne j \end{cases} \tag{3.16}$$

the inverse of Π^α can be expressed as,

$$\Pi^{\alpha-1} = \Pi\mathbf{A}\Pi^{-1}\mathbf{A}^T\Pi^T \tag{3.17}$$

Now that we obtained Φ^{-1} we substitute (3.5) and (3.17) into the equation for $g_k(\Phi(\mathbf{x}))$ to get

$$g_k(\Phi(\mathbf{x})) = (\Phi(\mathbf{x})^T \Phi \mathbf{K}_k^T \mathbf{1}_{l_k} / l_k)\mathbf{A}\Lambda^{-1}\mathbf{A}^T(\Phi^T \Phi(\mathbf{x}) \Phi \mathbf{K}_k \mathbf{1}_{l_k} / l_k) + \log(\pi_k) \quad (3.18)$$

Note that (3.18) can be evaluated for any class and any given sample **x** as the term $\Phi(\mathbf{x})^T \Phi$ can be evaluated by the kernel function and the rest of the equation is computable.

Equation (3.18) results in piece-wise linear decision boundaries between classes in the feature space. This is a direct result of assuming a common covariance matrix for all class distributions. At first sight it might look unwise to sacrifice from the complexity of the classifier just to obtain a kernelizable form of the decision rule. However this is not really the case. On the contrary we now have more flexibility to model a large variety of class conditional distributions which will ultimately give us more control over the complexity of the classifier.

Although there are several kernel functions to choose from, possible choices that have proven useful are Gaussian radial basis functions (RBF), $k(\mathbf{x}, \mathbf{x}_i) = \exp(-\|\mathbf{x} - \mathbf{x}_i\|^2 / \sigma^2)$ and polynomial kernels, $k(\mathbf{x}, \mathbf{x}_i) = (\mathbf{x}^T \mathbf{x}_i)^p$ for some positive constants σ and p respectively. Throughout this study we used Gaussian RBF, which has proved more effective in the analysis of hyperspectral images.

The parameter σ of the kernel function and the regularization term added to **G** are estimated by cross-validation. This process ensures that the nonlinear function used to implicitly transform the data into feature space is not an ordinary map. In contrast it is a map that reduces the cross-validation error. This has several important implications. Theoretically speaking the feature space induced by the Gaussian RBF kernel function is infinite dimensional (although the data lies in an m-dimensional subspace) so in many cases this results in the data being linearly separable[1]. As noted in [6] when the distributions are unimodal and separated by the scatter of means, a linear classifier is competent compared to more sophisticated classifiers. Therefore intuitively

[1] For two-class problems as noted in [46] it is possible to change any kernel function such that the data in the feature space is linearly separable.

speaking this transformation makes the normal model with a common covariance matrix a better candidate for the data by providing more separability between classes. Finally although the decision boundaries in the feature space are piecewise linear, they correspond to powerful nonlinear boundaries in the original space.

3.4 Experimental Data and Classification Tasks

3.4.1 Classification tasks

For each of the data sets described in chapter 1, the performances of four classifiers are compared. These are a linear classifier (LC), a simple quadratic classifier (SQC), a mixture classifier (MC) and the feature space counterpart of the linear classifier as proposed in this study, which we will call the kernel linear classifier (KLC). Classifiers are designed with a randomly chosen training set. The labeled samples which are not chosen as training data are used for testing. For each data set two different sizes of training set are considered. Our goal here is to test the performance of the proposed classifier first with a large and then with a relatively smaller training set. Each experiment is repeated 10 times and the testing error rates for all classifiers are plotted in each case.

3.4.2 Estimation of parameters

First we scale all the data sets to between 0 and 1. Although not very crucial, this step makes the algorithm numerically more stable and also allows us to use a more consistent range of values for all the data sets during parameter estimation. Both the width of the Gaussian RBF function σ and the regularization term λ added to **G** are estimated using 10-fold cross-validation (this is implemented with a training set not using a different validation set). For large n cross validation is performed with **v** (see equation 3.9) expressed using one third of the training samples. This reduces the dimensionality

and provides some time savings. A discrete set of values are considered for each parameter. More specifically for α the range we consider is [0.01 1] and for β, $[10^{-15}\ 10^{-1}]$.

3.4.3 Experimental results and discussion

Flightline C1:

The two different l values considered for this data set are $l = 2085$ (3 % of the entire labeled set) and $l = 348$ (0.5 % of the entire labeled set). Although the training samples are chosen randomly we also make sure each class gets at least 15 labeled samples in order to avoid numerical problems. Results obtained for all the four classifiers are shown in Figure 3.1 and 3.2. The classification maps obtained using the MC and KLC for $l = 2085$ are shown in Figure 3.7 and 3.8. For $l = 2085$ SQC, MC and KLC all performed equally well with KLC slightly better. For $l = 348$ KLC clearly outperformed all the other classifiers.

Washington DC Mall:

The two different l values considered for this data set are $l = 2025$ (10 % of the entire labeled set) and $l = 440$ (2 % of the entire labeled set) but at the same time we make sure each class gets at least 50 labeled samples. Also for SQC and MC the best 30 features obtained through the nonparametric weighted feature extraction approach introduced in [16] are used. We preferred this feature extraction algorithm over discriminant analysis feature extraction because the number of features we can choose is not restricted by the number of classes in this technique. As the experimental results in [16] suggests the best 30 features are more than sufficient for this data set and using more features doesn't further improve the performance of the SQC and MC. For LC and KLC no feature extraction step is needed. These classifiers operate over the entire feature set. Results obtained for all the four classifiers are shown in Figure 3.3 and 3.4. The classification maps obtained using the MC and KLC for $l = 2025$ are shown in Figure

3.9 and 3.10. For both $l = 2025$ and $l = 440$, KLC significantly outperforms all the rest of the classifiers.

Purdue Campus Data:

The two different n values considered for this data set are $l = 2155$ (10 % of the entire labeled set) and $l = 762$ (3 % of the entire labeled set). Again we make sure each class gets at least 50 labeled samples. For SQC and MC parallel to our results in [3] we use the best 30 features obtained through the nonparametric weighted feature extraction. For LC and KLC the entire feature set is used. Results obtained for all the four classifiers are shown in Figure 3.5 and 3.6. The classification maps obtained using the MC and KLC for $l = 2155$ are shown in Figure 3.11 and 3.12. For $l = 2155$ KLC and MC performed equally well. For $l = 762$ KLC is considerably better.

3.4.4 Discussion and analysis of results

The results for the data considered look very encouraging. We observed that the proposed kernel linear classifier performed equally well with or sometimes slightly better than the mixture classifier (the most competent among the other three) for large l and substantially better for small l.

Compared to the mixture classifier the proposed classifier also has lower variance throughout the results; a desirable characteristic especially when dealing with small training sets. The underlying strength of the kernel linear classifier stems from the implicit nonlinear transformation we suggested for the data and the way this transformation is carried out. This transformation ensures that in the space where the data is mapped to, the classes are more separable and class data is more suitable for modeling with normal distributions having a common covariance matrix.

The classifier is designed in the feature space but the computations are done in the input space. As a result of this the classifier in the feature space has piecewise linear boundaries which correspond to nonlinear boundaries in the input space. As being a linear classifier this also makes the proposed classifier less prone to noise and over

fitting to some extent. Although we used a regularization constant for N, the statistics estimation with this approach is expected to be more robust than the mixture classifier, as we are utilizing all the training samples in estimating N. Besides, the regularization constant added to N is very small (for the data considered it was usually less than 10^{-8}).

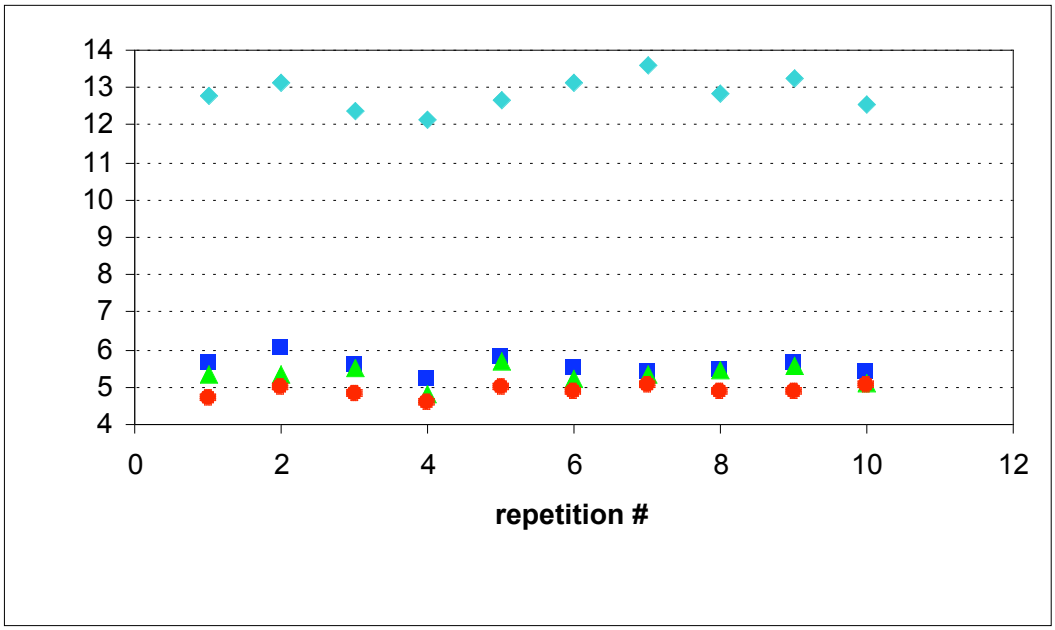


Fig. 3.1 Testing error rates obtained in each repetition for the Flightline C1 data set for $|I| = 2085$.

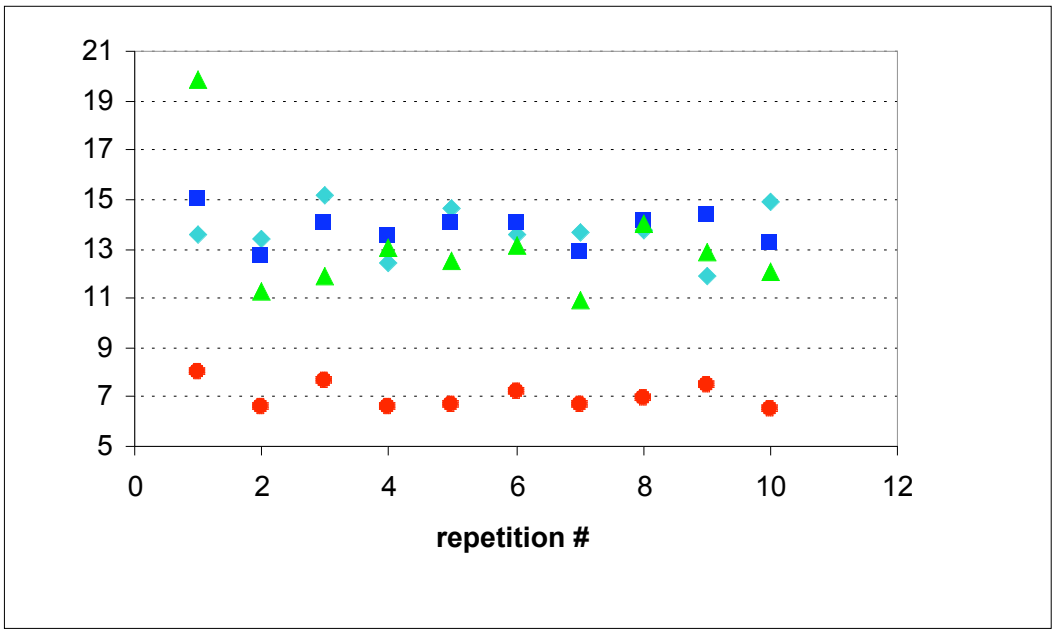


Fig. 3.2 Testing error rates obtained in each repetition for the Flightline C1 data set for $|I| = 348$.

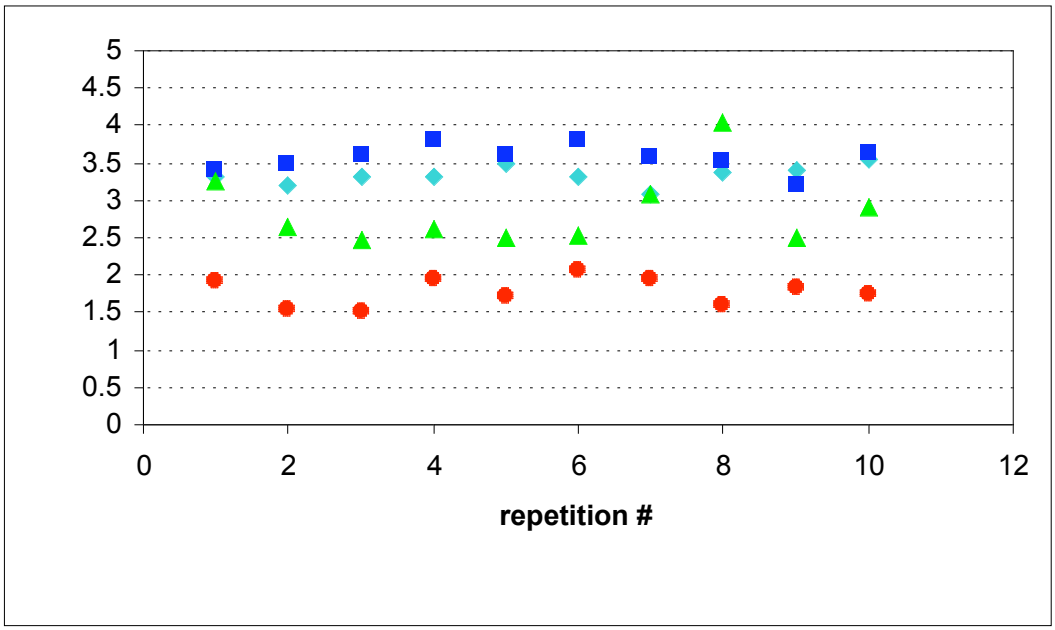


Fig. 3.3 Testing error rates obtained in each repetition for the Washington DC Mall data set for $l = 2025$.

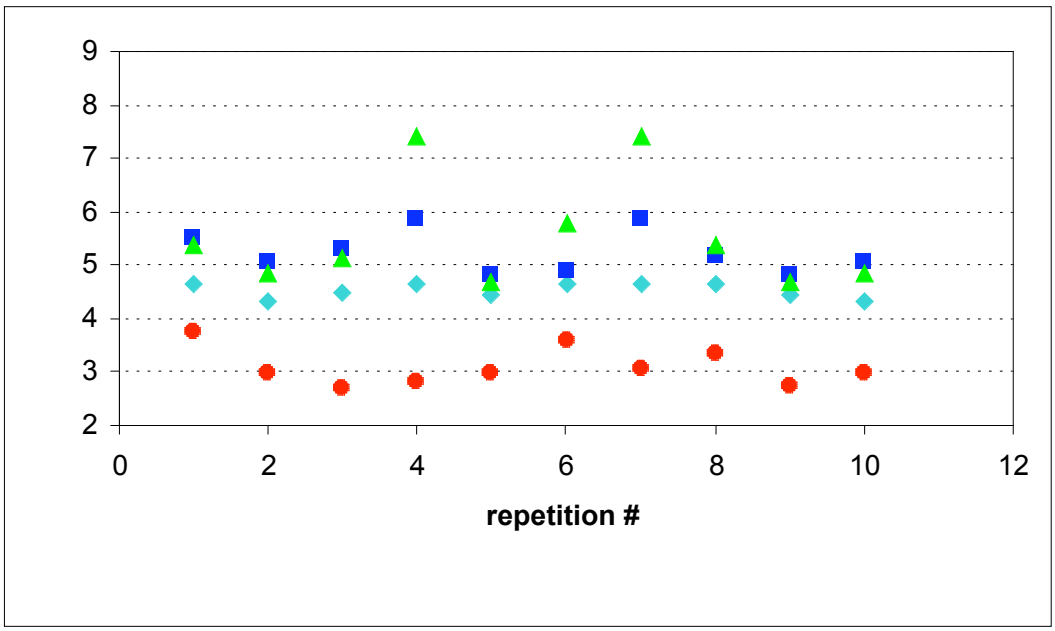


Fig. 3.4 Testing error rates obtained in each repetition for the Washington DC Mall data set for $l = 440$.

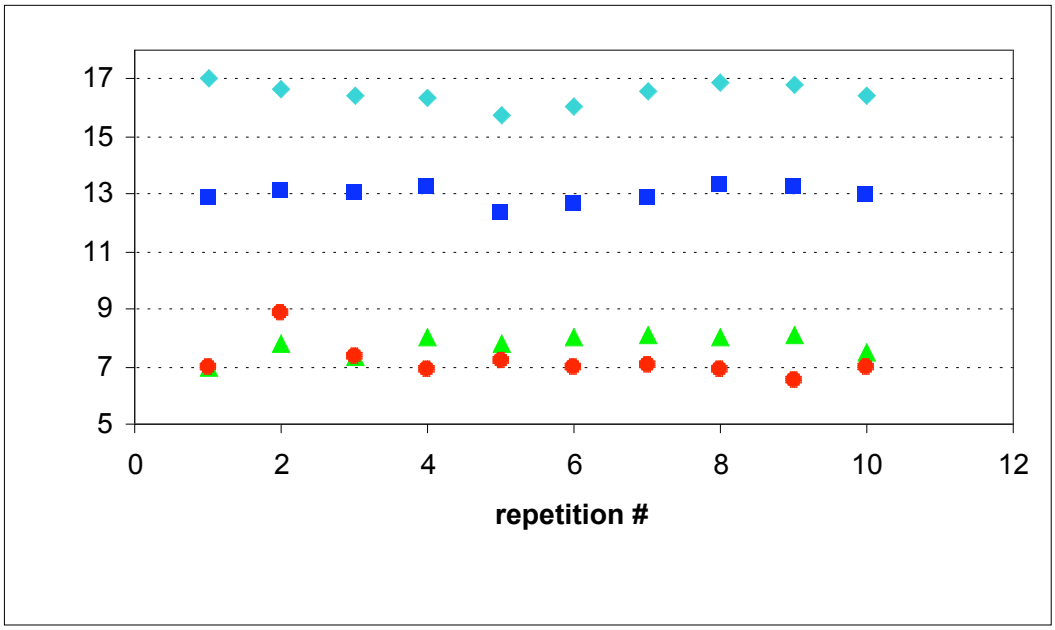


Fig. 3.5 Testing error rates obtained in each repetition for the Purdue Campus data set for $l = 2155$.

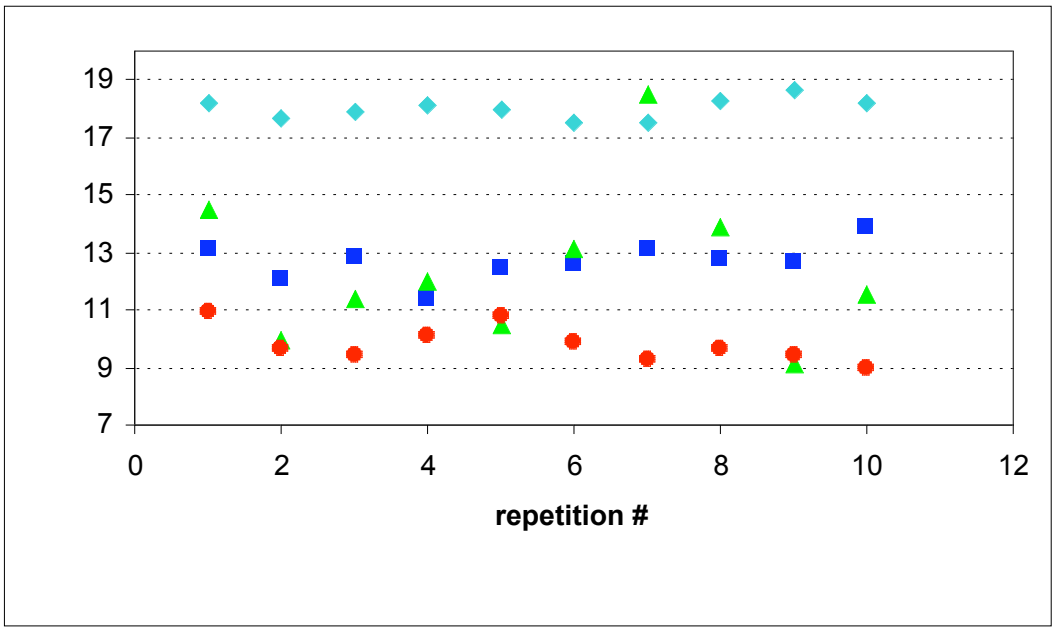


Fig. 3.6 Testing error rates obtained in each repetition for the Purdue Campus data set for $l = 762$.

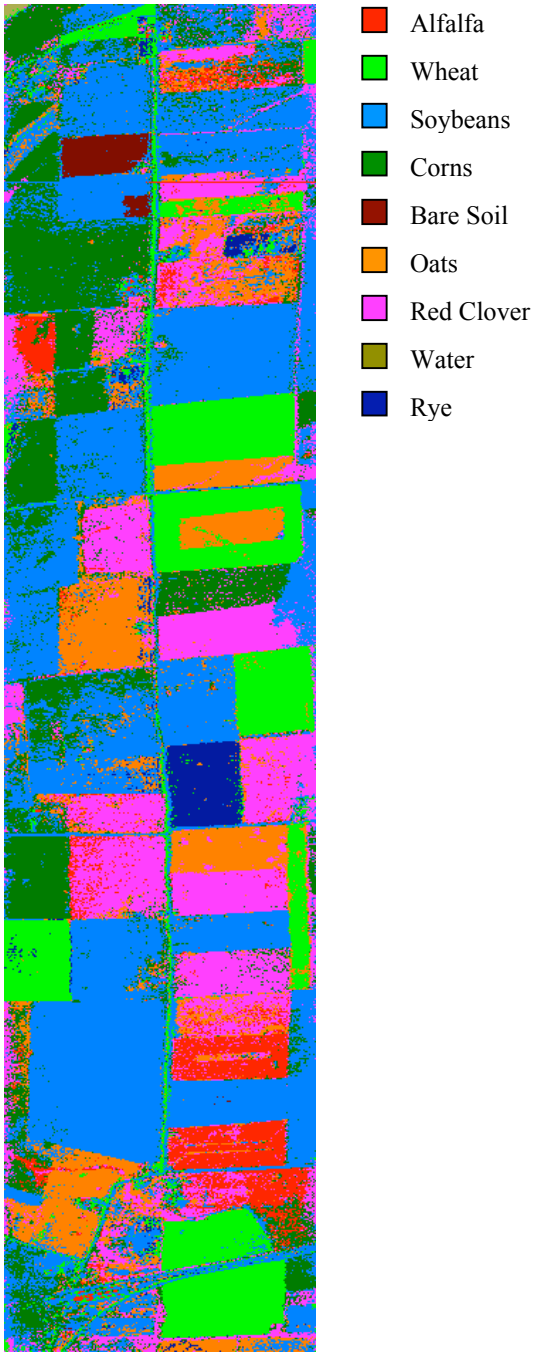


Fig. 3.7 Classification map obtained for the Flightline C1 data by MC for r=0.1 (% error=5.1).

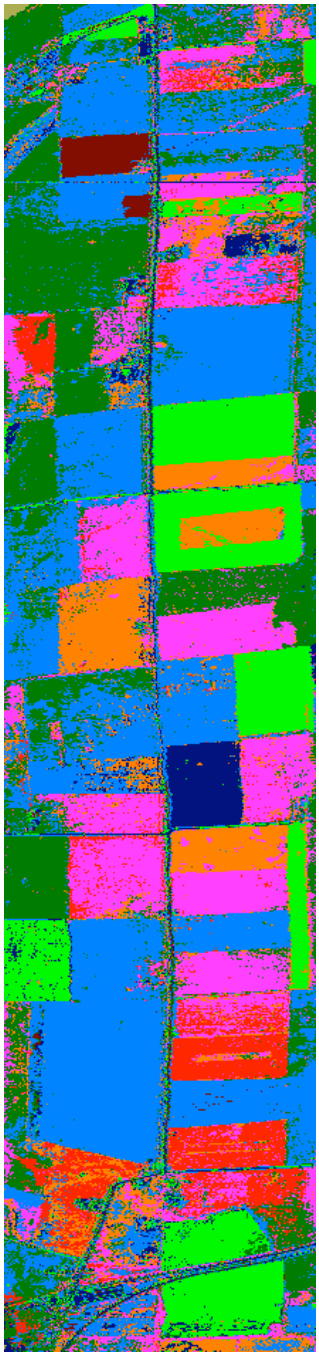


Fig. 3.8 Classification map obtained for the Flightline C1 data by KLC for r=0.1 (% error=5.3).

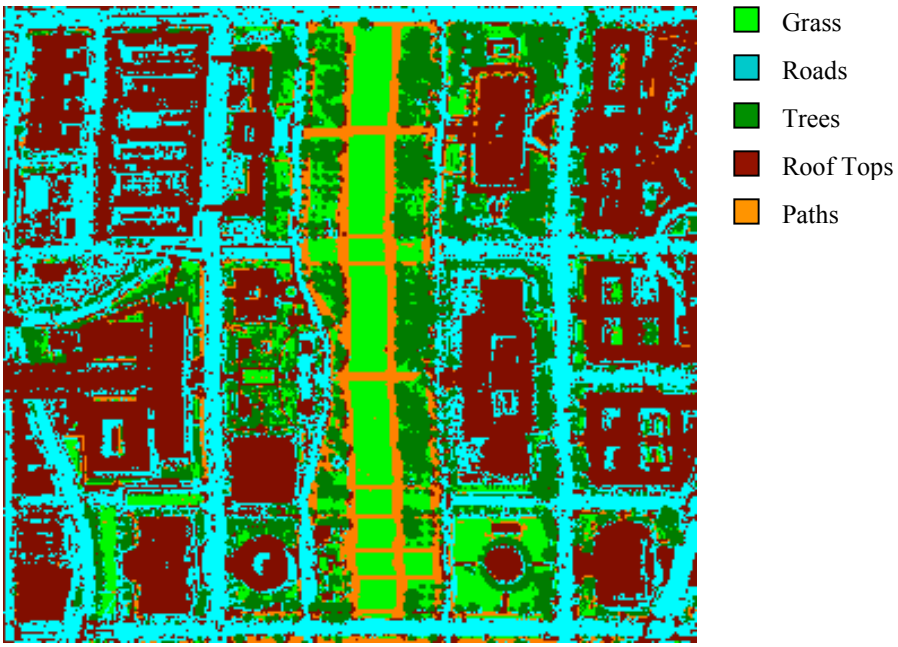


Fig. 3.9 Classification map obtained for the Washington DC Mall data by MC for r=0.1 (% error=2.7).

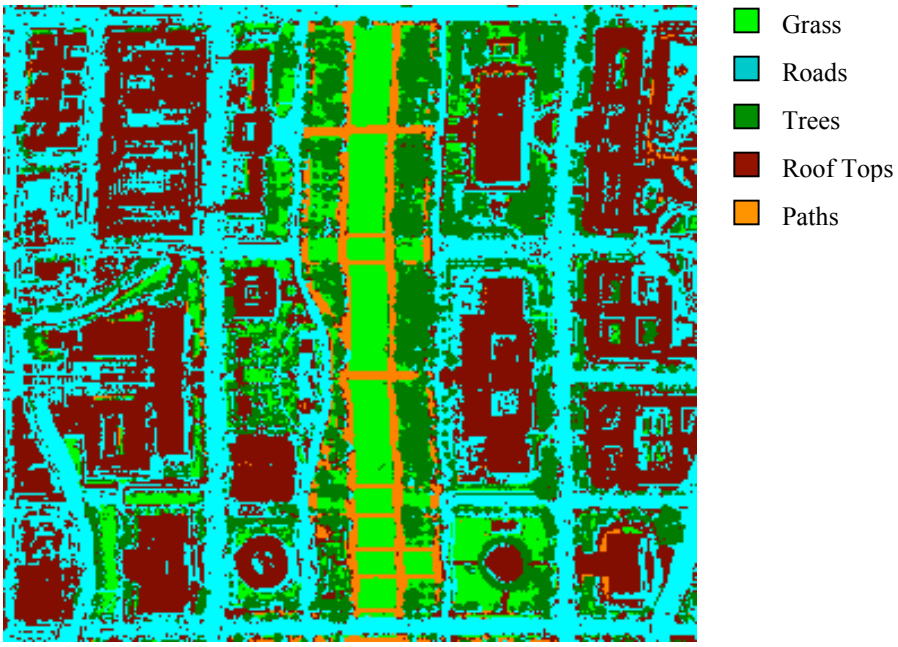


Fig. 3.10 Classification map obtained for the Washington DC Mall data by KLC for r=0.1 (% error=1.6).

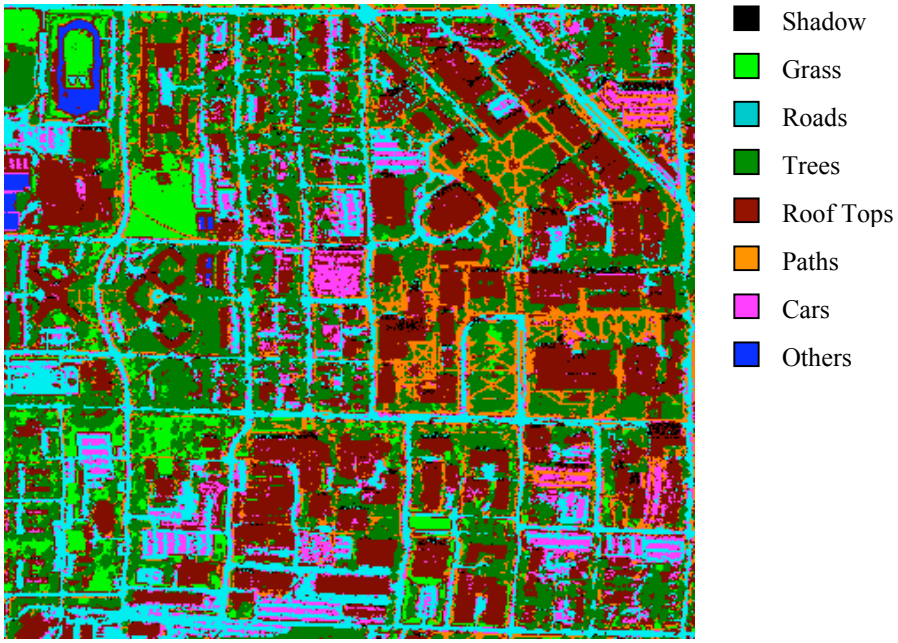


Fig. 3.11 Classification map obtained for the Purdue Campus data by MC for r=0.1 (% error=7.7).

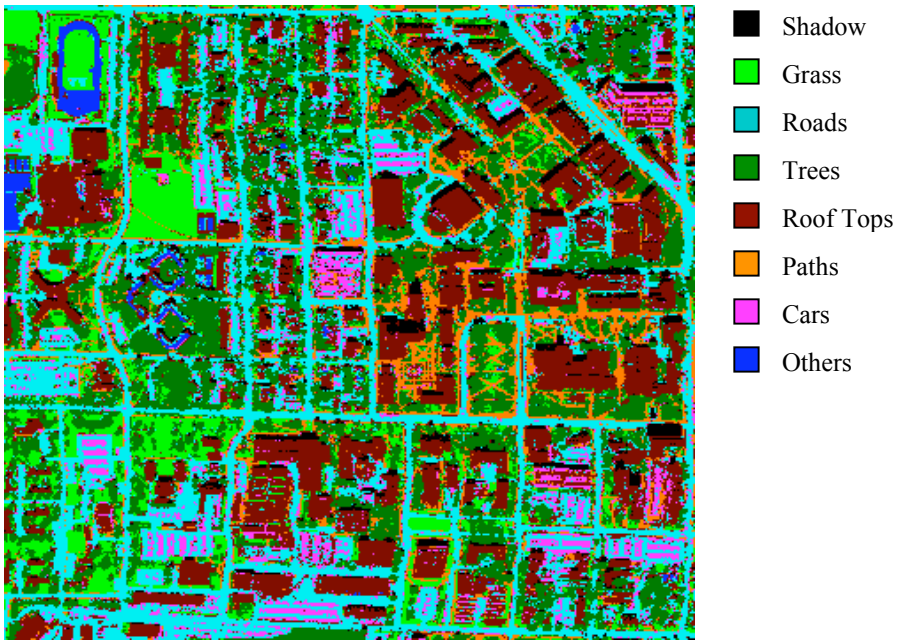


Fig. 3.12 Classification map obtained for the Purdue Campus data by MC for r=0.1 (% error=6.7).

3.5 Conclusions

For the data we have investigated the results favoring the proposed kernel linear classifier over all the other classifiers considered clearly look very promising. Although for large l the mixture classifier performed equally well, the performance of the proposed kernel linear classifier was significantly better for small l. Considering the prohibitively expensive price we pay for training samples in remote sensing applications, we think this improvement is a remarkable step toward obtaining an optimal supervised classifier.

The proposed technique gives us the flexibility required to model complex data structures that originate from a wide-range of class conditional distributions. Like its linear space counterpart the statistics estimation is performed at full dimensionality (no feature extraction is needed) allowing us to exploit all the separability the data provides without stumbling on numerical issues. As the results suggest, seemingly simple linear piece-wise decision boundaries in the feature space correspond to powerful nonlinear boundaries in the input space.

Although it is very crucial, the effect of the implicit nonlinear transformation on the data is not very clear. However it is intuitive to think that the high dimensional feature space induced by this transformation provides more separability allowing a normal model to fit the data more efficiently.

Like most other kernel-based approaches the proposed technique also suffers from the computational complexity of working with kernel functions to some extent. We need to evaluate the kernel function l^2 times during the design of the classifier to compute $\mathbf{G}$ and $l \cdot n_t$ times during the testing stage to evaluate $g(\mathbf{x})$ where l is the number of training and n_t is the number of testing samples. Apart from this we need to solve a generalized eigenvalue problem for n by n dimensional matrices as in equation (3.11). For large n these issues may become quite problematic. One way across this problem is to express $\mathbf{v}$ in (3.9) in terms of a portion of the training samples rather than the full set to reduce the dimensionality (as we did in this study during parameter

estimation for large l). Nonetheless we were able to implement each run of this algorithm including the parameter estimation stage in less than 45 minutes using a Linux based Pentium 4 machine.

4. A COST-EFFECTIVE SEMI-SUPERVISED CLASSIFIER WITH KERNELS

3.1 Introduction

In attacking today's challenging classification problems, it is very desirable to have as much labeled data as possible. Unfortunately for most real-world settings such as text categorization and remotely sensed data analysis, acquiring labeled data requires a tedious, time-consuming process of human labeling. Therefore the price one must pay for labeled data is often prohibitively expensive. This has raised interest in the readily available large amount of unlabeled data. Various methods taking advantage of unlabeled data to improve classification performance have already been proposed. Such methods include but are not limited to use of the expectation-maximization (EM) algorithm with finite mixture models [18, 19, 32], transductive learning [33-35] and a co-training framework [36]. Despite the ongoing discussion [37-38] about whether these attempts are successful and unlabeled data are truly useful, it has been theoretically proven that under a zero bias assumption unlabeled data reduces the variance of the estimator and helps classification [19], and under various assumptions, classification error decreases exponentially with the number of labeled samples and linearly with the number of unlabeled samples [39-40].

Among other techniques for incorporating unlabeled data into the design of a classifier, the EM algorithm has drawn more attention. EM is an iterative algorithm for maximum likelihood estimation with incomplete data [41]. Here unlabeled data is considered incomplete because they come without class labels. In the context of EM the unlabeled data problem is usually attacked with a generic model such as finite mixture models. When the generic model assumption for the classifier is "correct" that is, the

model used to build the classifier is identical to the model that generated the data; early work [42] has proved that under the additional assumption of identifiability a large set of unlabeled data alone is sufficient to identify the mixture components. In this case labeled data are only used to associate these components with classes.

While these results are encouraging, it is well known that real-world data are unlikely to be this easily modeled. Most approaches adopt normal mixture models to characterize class distributions. However normal densities are clearly not flexible enough to capture complex data structures encountered in real-world settings. Empirical studies [18, 37], indeed have shown that when there is a mismatch between approximating a distribution and the true underlying distribution, unlabeled data may actually degrade the accuracy of the classifier. In order to mitigate this problem several extensions to the basic EM have been proposed, most of which are centered on two main ideas. The first one is to introduce a weighting factor that will dynamically adjust the contribution of unlabeled data to parameter estimation [18, 32] and the second is to model each class distribution with multiple mixture components [18,19]. In what follows these two approaches will be discussed with reference to the techniques introduced in [19, 32].

In [32], an iterative framework based on the Maximum Likelihood (ML) rule is proposed to enhance the statistics estimation by using unlabeled data in addition to labeled data. In this technique, unlike EM, unlabeled data only contributes to the statistics of the class to which it has been classified. The classifier assigns full weight to labeled samples, but the amount of contribution of each unlabeled sample is automatically weighted by the sample's posterior probability (equal class prior probabilities are assumed). This technique doesn't directly deal with the model mismatch problem. It is assumed that each class distribution can be approximated by a normal density within good accuracy and the samples which don't quite follow a normality pattern are penalized by assigning reduced weight in estimating the statistics of the class to which they have been classified.

A major concern with this technique is that when the class distributions are multimodal and hence cannot be characterized by a single normal density the posterior probabilities obtained through initial classification of the unlabeled data might be totally

misleading. Since the model doesn't represent unlabeled data very well most unlabeled data will be incorrectly classified into one of the classes available for the mere reason that it generated the highest posterior probability for that class. Therefore when there is model mismatch, unlabeled samples even when they are weighted and contribute only to the class to which they have been classified will likely have an adverse effect on statistics estimation and thereby degrade the performance of the classifier in the succeeding iterations. The authors attempt to get around this problem by manually introducing subclasses.

When faced with data that does not fit into the generic model, the above approach addresses the problem by limiting the effect of the unlabeled data. A more direct approach is to use a more flexible model that can capture more complex data structures. In [19], this is achieved by allowing each class to have multiple normal components. It is well known that given enough components any given distribution can be approximated within a good accuracy using a mixture of normal densities. The EM algorithm is used to fit the data into the augmented mixture model. However the expectation stage is slightly modified such that a labeled sample only contributes to the components belonging to its class of origin whereas an unlabeled sample can contribute to any component. Intuitively speaking characterizing each class distribution using a mixture of normal densities might seem the right thing to do. Indeed several studies have shown that classifiers designed this way can outperform those using a single normal density per class. In the presence of moderately sized labeled data sets, these studies have also addressed the standard issues of finding the right number of mixture components and avoiding ill-conditioned statistics estimation to a greater extent. However clearly the need for unlabeled data arises when the amount of labeled data is very limited in which case the underlying structures of class distributions are mostly hidden and cannot be easily captured. This poses a problem in identifying the subclasses and estimating their statistics. One could avoid this problem by assigning an excessive number of components for each class, but in that case statistics estimation becomes very impractical.

Even though they are far from solving the unlabeled data problem completely, early work definitely provides insight into the problem. In dealing with unlabeled data it

is very important that the model should be flexible enough to accommodate the wide range of class distributions. In [25] a nonlinear classification technique called Kernel Fisher's Discriminant (KFD) has been proposed. The main ingredient of this approach is the kernel concept that is originally applied in Support Vector Machines [22-24] and allows the efficient computation of Fisher's Discriminant in the kernel space. The linear discriminant in the kernel space corresponds to a powerful nonlinear decision function in the input space. Furthermore different kernels can be used to accommodate the wide-range of nonlinearities possible in the data set.

In [25], KFD is compared to some state-of-the-art classifiers and Support Vector Machines using 13 artificial and real world datasets. The experiments show that KFD is competitive or in some cases even superior to the other algorithms on almost all data sets. It might be premature to say that KFD is a powerful alternative to classical techniques but it certainly has some characteristics which when explored may lead to improved algorithms. Indeed in [26] an algorithm for constructing a Kernel Fisher's discriminant (KFD) from training samples with noisy labels is proposed. This problem is very similar to the unlabeled data problem in that the labels assigned to some of the training samples are assumed to be incorrect. However the fact that which of these samples might have incorrect labels is unknown makes this problem more difficult. The authors attack this problem by implementing the EM algorithm in some kernel space. The E-step uses class conditional probabilities estimated as a by-product of the KFD algorithm and the M-step updates the flip probabilities and determines the parameters of the discriminant. They test the proposed approach on an image problem where the task is to find whether an image contains 'sky' or not. Even though this problem is usually dealt with in a different context the proposed approach still gives some clue to in what ways implementing Fisher's discriminant with kernels can be useful for the unlabeled data problem.

In iterative semi-supervised learning algorithms, labeled data apart from determining the initial structure of the classifier have another significant role. At eachiteration, the number of misclassified labeled samples gives us an idea on how well the unlabeled data is incorporated into the classifier model. When the generic model used to build the classifier is not close enough to the actual model the large amount of

unlabeled data will certainly have an adverse effect on the classifier performance by causing gradual deterioration in the error rate of labeled data. Most semi-supervised learning algorithms with EM tackle this problem by adjusting the posterior probabilities of the data during the "expectation" stage such that labeled samples are always assigned to their class of origin with full weight and unlabeled data are assigned to the class to which they are classified by reduced weight. However this rather brute-force post-processing approach might fail when the amount of unlabeled data is very large compared to labeled data, which is typically the case in semi-supervised settings. Unlabeled data no matter how much they are weighted might eventually become the dominant factor in determining the classification boundary.

In this study we attack the unlabeled data problem by finding the Fisher's Discriminant in the feature space using labeled and unlabeled data together. There is a large class of kernels that can be used, leading to a large class of available feature spaces. This provides us the flexibility we need for the classifier model. At the same time since we are computing the discriminant in some feature space we maintain the simplicity and efficiency of the Fisher's discriminant.

The proposed technique is unique in that the decision function is obtained through an optimization of a quadratic programming problem that minimizes the total cost of misclassified labeled data while maximizing the Rayleigh coefficient in the feature space. The within-class and between class scatter matrices are estimated using labeled as well as unlabeled data. However unlike semi-supervised learning approaches using the EM algorithm the labeled data are incorporated into the process of finding the best decision function not only through scatter matrices but also through a more direct measure of in-sample error rate. Minimizing the total cost of misclassified labeled samples and maximizing the Rayleigh coefficient in the presence of unlabeled data might be two conflicting goals. However by adjusting the cost of each misclassified labeled sample one can obtain an efficient trade-off between these two.

The idea of minimizing the total cost of misclassified labeled samples when determining the decision function is not new. As a matter of fact Support Vector Machines (SVM's) inherently have this characteristic and there have been several recent

attempts [33-35] to generalize the performance of SVM's through transductive learning. However as argued in [38] unlabeled data are not expected to help much for SVM's because the data distribution does not carry any parameter information. Even worse, finding the optimal boundary in the presence of unlabeled data (i.e. test data) is a combinatorial optimization problem and the most efficient algorithm can only accommodate up to a few hundred test samples.

4.2 Kernel Concept and Fisher's Discriminant

We know that the probability of error due to the Bayes classifier is the best we can achieve. A major disadvantage of the Bayes error as a criterion is that a closed-form analytical expression is not available in general. However by assuming normal distributions for classes, standard classifiers using quadratic and linear discriminant functions can be designed.

The famous Fisher's Linear Discriminant (FLD) [6], arises in the special case when classes have a common covariance matrix. FLD is a classification method that projects the high-dimensional data onto a line and performs classification in this one-dimensional space. This projection is chosen such that the ratio of between and within class scatter matrices or the so called *Rayleigh coefficient* is maximized.

More specifically let $X_k = [\mathbf{x}_1^k, \ldots, \mathbf{x}_{l_k}^k]$ be the training data matrix and l_k the number of labeled samples for class k, $\mathbf{x}_i \in R^d, k \in \{\pm\}$. FLD is the projection $\mathbf{w}$, which maximizes,

$$J(\mathbf{w}) = \frac{\mathbf{w}^T S_B \mathbf{w}}{\mathbf{w}^T S_W \mathbf{w}} \tag{4.1}$$

where,

$$S_B = (\mathbf{m}_+ - \mathbf{m}_-)(\mathbf{m}_+ - \mathbf{m}_-)^T \tag{4.2}$$

$$S_w = \sum_{k \in \{\pm\}} \frac{1}{l_k} (X_k - \mathbf{m}_k \mathbf{1}_k^T)(X_k - \mathbf{m}_k \mathbf{1}_k^T)^T \tag{4.3}$$

are the between and within class scatter matrices respectively and

$$\mathbf{m}_k = \frac{1}{l_k} X_k \mathbf{1}_{l_k} \tag{4.4}$$

is the mean of class k and $\mathbf{1}_k$ is an l_k dimensional vector of ones. There are several ways of maximizing (4.1). One way is to solve the generalized eigenvalue problem $S_B\mathbf{w} = \lambda S_w\mathbf{w}$. However for large data sets this becomes very inefficient. On the other hand by transforming (4.1) into a convex quadratic programming problem one can benefit from several algorithmic advantages. First, notice that if a vector $\mathbf{w}$ is a solution to (4.1), then so is any scalar multiple of it. Therefore to avoid multiplicity of solutions, we further impose the constraint $\mathbf{w}^T S_B \mathbf{w} = b^2$, which is equivalent to $\mathbf{w}^T(\mathbf{m}_+ - \mathbf{m}_-) = b$ where b is an arbitrary scalar. Then the optimization problem of (4.1) becomes,

$$\begin{aligned} \min_{\mathbf{w}} \quad & \mathbf{w}^T S_W \mathbf{w} \\ \text{subj. to} \quad & \mathbf{w}^T(\mathbf{m}_+ - \mathbf{m}_-) = b \end{aligned} \tag{4.5}$$

Using Lagrange method it is straightforward to show that the solution $\mathbf{w}^*$ is,

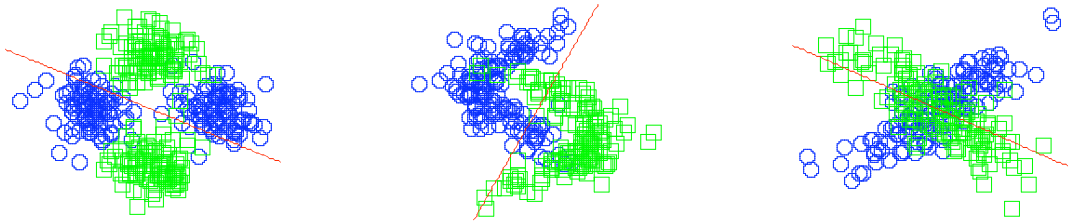
$$\mathbf{w}^* = \frac{b S_w^{-1}(\mathbf{m}_+ - \mathbf{m}_-)}{(\mathbf{m}_+ - \mathbf{m}_-)^T S_w^{-1}(\mathbf{m}_+ - \mathbf{m}_-)} \tag{4.6}$$

When classes are normally distributed with equal covariance, $\mathbf{w}^*$ are in the same direction as the discriminant in the corresponding Bayes classifier. Hence for this special case FLD is equivalent to the Bayes optimal classifier. Although it relies on heavy assumptions that are not true in most real world problems, FLD has proven very powerful. Generally speaking when the distributions are unimodal and separated by the scatter of means FLD becomes very appealing compared to a simple quadratic classifier. One reason why FLD can be preferred over a simple quadratic classifier is that as a linear classifier it is less prone to noise and most likely will not overfit.

Clearly for most real world data a linear discriminant is not complex enough. Typical examples where FLD will perform poorly are shown in Figure 4.1. In Figure 4.1.a and Figure 4.1.b the distributions are multimodal, and in Figure 4.1.a and Figure 4.1.c the distributions share the same mean.

Classical techniques tackle these problems by using more sophisticated distributions in modeling the optimal Bayes classifier, however these often sacrifice the closed form solution. A relatively new approach is to look for nonlinear directions. Mika

et. al. [25] has proposed a nonlinear classification technique based on Fisher's Discriminant which they called Kernel Fisher's Discriminant (KFD). The main ingredient of their approach is the kernel concept, which is originally applied in Support Vector Machines [22-24] and allows the efficient computation of Fisher's Discriminant in the kernel space. The linear discriminant in the kernel space corresponds to a powerful nonlinear decision function in the input space. Furthermore different kernels can be used to accommodate the wide-range of nonlinearities possible in the data set. Kernel Fisher's Discriminant constitutes the core component of this study. Hence in what follows we will derive Fisher's discriminant in the kernel space which is mostly due to [25].



(a) (b) (c)

Fig. 4.1 Examples for which FLD doesn't work. Class 1 data is depicted by circles, Class 2 data is depicted by squares, and the solid lines are the decision boundaries obtained by FLD[2].

We first assume an implicit mapping of the data into a possibly high dimensional feature space F by a nonlinear map $\Phi: R^d \rightarrow F$. Once we can express our algorithm in the dot-product form, the kernel concept gives us an elegant way of implementing the algorithm in the input space without ever mapping the data explicitly to F. There is a wide-range of kernel functions we can use leading to many different feature spaces. This gives us the flexibility to control the complexity of the classifier model. Possible choices for k which have proven useful are Gaussian radial basis functions (RBF),

[2] These toy examples are adapted from [6].

$k(\mathbf{x}, \mathbf{x}_i) = \exp(-\|\mathbf{x} - \mathbf{x}_i\|^2 / \sigma^2)$, or polynomial kernels, $k(\mathbf{x}, \mathbf{x}_i) = (\mathbf{x}^T \mathbf{x}_i)^p$ for some positive constants σ and p respectively.

In order to find the Fisher's discriminant in some kernel space F we need to solve,

$$\begin{aligned} &\min_{\mathbf{w}} \quad \mathbf{w}^T S_W^{\Phi} \mathbf{w} \\ &\text{subj. to} \quad \mathbf{w}^T (\mathbf{m}_+^{\Phi} - \mathbf{m}_-^{\Phi}) = a \end{aligned} \tag{4.7}$$

where $\mathbf{w} \in F$ and S_B^{Φ} and $\mathbf{m}_k^{\Phi}$ are now defined as,

$$S_w^{\Phi} = \sum_{k \in \{\pm\}} \frac{1}{l_k} (\Phi_k - \mathbf{m}_k^{\Phi} \mathbf{1}_k^T)(\Phi_k - \mathbf{m}_k^{\Phi} \mathbf{1}_k^T)^T \tag{4.8}$$

$$\mathbf{m}_k^{\Phi} = \frac{1}{l_k} \Phi_k \mathbf{1}_k \tag{4.9}$$

and $\Phi_k = [\Phi(\mathbf{x}_1^k), \ldots, \Phi(\mathbf{x}_{l_k}^k)]$. Here the superscript Φ is used to distinguish nonlinear parameters from their linear counterparts. Before we can benefit from the kernel concept we should express (4.7) in the dot-product form.

An important result from the theory of reproducing kernels will help us achieve this goal. Any solution **w*** of (4.7) must lie in the span of all training samples in F. Therefore **w** can be expanded in the form,

$$\mathbf{w} = \underbrace{[\Phi_+ \; \Phi_-]}_{\Phi} \mathbf{a} \tag{4.10}$$

where **a** is an l-dimensional vector and $l = l_+ + l_-$. After substituting (4.8), (4.9) and (4.10) into (4.7) we get,

$$\begin{aligned} &\min_{\mathbf{a}} \quad \mathbf{a}^T N \mathbf{a} \\ &\text{subj. to} \quad \mathbf{a}^T \mathbf{d} = b \end{aligned} \tag{4.11}$$

where N, **d** are defined as,

$$N = \sum_{k \in \{\pm\}} \frac{1}{l_k} \left\| \Phi^T \Phi_k (I - \frac{1}{l_k} \mathbf{1}_k \mathbf{1}_k^T) \right\|^2 \tag{4.12}$$

$$\mathbf{d} = (\frac{1}{l_+} \Phi^T \Phi_+ \mathbf{1}_+ - \frac{1}{l_-} \Phi^T \Phi_- \mathbf{1}_-) \tag{4.13}$$

Now that (4.12) and (4.13) are in dot-product form, N and **d** can be evaluated and hence (4.11) can be solved. We replace $\Phi^T \Phi_k$ in (4.12) and (4.13) by K_k where K_k is the kernel

matrix for class k whose $(i,j)^{th}$ element is obtained by $k(\mathbf{x}_i,\mathbf{x}_j^k)$. Finally we end up with the following optimization problem,

$$\begin{aligned} \min_{\mathbf{a}} \quad & \mathbf{a}^T N \mathbf{a} \\ \text{subj. to} \quad & \mathbf{a}^T \mathbf{d} = b \end{aligned} \tag{4.14}$$

where N and $\mathbf{d}$ are now,

$$N = \sum_{k \in \{\pm\}} \frac{1}{l_k} \left\| K_k \left(I - \frac{1}{l_k} \mathbf{1}_k \mathbf{1}_k^T \right) \right\|^2 \tag{4.15}$$

$$\mathbf{d} = \left(\frac{1}{l_+} K_+ \mathbf{1}_+ - \frac{1}{l_-} K_- \mathbf{1}_- \right) \tag{4.16}$$

The initial problem of finding the Fisher's discriminant in the kernel space has now turned into a problem of finding $\mathbf{a}$ in the l-dimensional vector space, i.e. $\mathbf{a} \in R^l$. Similar to (4.6) the solution $\mathbf{a}$ of (4.16) is found as,

$$\mathbf{a}^* = \frac{b N^{-1} \mathbf{d}}{\mathbf{d}^T N^{-1} \mathbf{d}} \tag{4.17}$$

Once $\mathbf{a}^*$ is obtained $\mathbf{w}$ can be expressed in terms of $\mathbf{a}^*$ as in (4.10) and the projection of a new pattern $\mathbf{x}$ onto $\mathbf{w}$ can be computed by,

$$\phi(\mathbf{x})^T \mathbf{w}^* = \sum_{i=1}^{l} a_i^* k(\mathbf{x}_i, \mathbf{x}) \tag{4.18}$$

As a final side note; the above setting is ill-posed because N, which is an l-dimensional matrix, is estimated using l samples, thereby N is ill-conditioned. In [1] this problem is tackled by adding a multiple of the identity matrix to N,

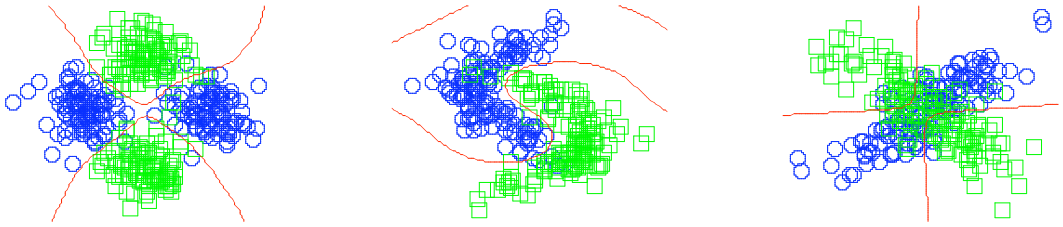
$$N_\mu = N + \mu I \tag{4.19}$$

Apart from making the algorithm numerically more stable μ also serves as a capacity control in F.

In Figure 4.2 the decision boundaries obtained by KFD for the toy examples introduced in Figure 4.1 are shown. In Figure 4.2.a and Figure 4.2.b the normality assumption and in Figure 4.2.c the common covariance assumption are not met, yet unlike its linear counterpart KFD is able to find the optimal boundaries in all three cases.

4.3 Estimating the Discriminant in the Semi-supervised Framework

Kernelizing the classifier introduces the flexibility required to accommodate a large variety of class conditional distributions. The complexity of the classifier can be easily controlled by varying the type of the kernel function as well as its parameters. Moreover using KFD one can handle complex data structures without sacrificing the theoretical



(a) (b) (c)

Fig. 4.2 The decision boundaries obtained by KFD for the examples of Fig. 4.1. A Gaussian radial basis kernel with a width of 0.1 is used in all three cases. Class 1 data is depicted by circles, Class 2 data is depicted by squares, and the solid lines are the decision boundaries obtained by KFD.

beauty and simplicity of Fisher's Discriminant. Also, apart from its relationship to Bayes classifier as a linear classifier Fisher's Discriminant is less prone to noise and most likely will not overfit. These inherent characteristics make KFD an ideal candidate classifier for exploiting unlabeled data.

In what follows we propose a modified version of KFD for semi-supervised learning. A typical semi-supervised KFD can be designed as follows. In the first step, a kernel function is determined and, using labeled samples, kernel parameters are estimated (possibly by cross validation). In the second step, using this kernel function and its parameters, N and **d** in (4.12) and (4.13) are estimated, and a supervised KFD is designed. In the third step, unlabeled data is classified by the designed classifier.

Unlabeled samples after classification are usually called semi-labeled samples in the literature. They are different than labeled samples in that labels assigned to them are not necessarily correct. In the fourth step, semi-labeled samples and labeled samples together are used to estimate N and **d**. In estimating N and **d** one can give more weight to labeled samples and reduced weight to semi-labeled samples using some designated criterion. Steps 3 and 4 are repeated until some designated convergence criterion is met. One issue needs further consideration in this procedure. Kernel parameters are estimated using a small number of labeled samples. However these parameters may not be good enough to capture the additional nonlinearities that might be introduced by a large set of semi-labeled samples. More specifically, the initial classifier model determined by a limited number of labeled samples may not be fully representative of the entire data set. This is almost unavoidable in most real world problems where the ratio of labeled samples available to unlabeled samples is too small. One could tackle this problem by estimating the kernel parameters at each iteration using labeled and semi-labeled samples together. However this is not feasible for two reasons. First, the dimensionality of N and **d** would be $l + s$ which is usually impractically large (l is the number of labeled samples and s is the number of semi-labeled samples). Second, some of the semi-labeled samples are no doubt misclassified. When these samples dominate the process of estimating the kernel parameters we may lose the initial flexibility we obtained through labeled samples let alone improving it. For these reasons we use only labeled samples to estimate the kernel parameters. Hence estimating kernel parameters using labeled and semi-labeled samples together is not a very practical solution for dealing with an unrepresentative labeled data problem. However there is more we can do. We can control how well the unlabeled data are being fit into the classifier model by monitoring the classifier error for labeled samples.

Although slight deterioration in this error rate might be tolerated a constant increase is definitely not desirable. When this happens unlabeled data are making an adverse effect on the classifier performance. An ideal solution would be to determine the discriminant such that no labeled samples are misclassified. However in that case not only the potential impact of unlabeled data is inhibited but the classifier might overfit as

well. Most semi-supervised learning algorithms with EM tackle this problem by adjusting the posterior probabilities of the data during the "expectation" stage such that labeled samples are always assigned to their class of origin with full weight and unlabeled data are assigned to the class to which they are classified by reduced weight. This rather post-processing brute-force approach doesn't necessarily lead to a better error rate for labeled samples in the next iteration, because the number of labeled samples is usually very small compared to the number of unlabeled samples and hence is not a very efficient way of controlling the effect of unlabeled data. A more practical solution is to allow some labeled data to be on the wrong side of the classification boundary and minimize the total cost associated with such samples while maximizing the Rayleigh coefficient. The optimization problem of (4.7) can be updated to accommodate this change in the cost function as follows,

$$\begin{aligned} &\min_{\mathbf{w}} \quad \mathbf{w}^T S_W^{\Phi} \mathbf{w} + \sum_{i=1}^{l} c_i \xi_i \\ &\text{subj to} \quad \mathbf{w}^T (\mathbf{m}_+^{\Phi} - \mathbf{m}_-^{\Phi}) = b \\ &\xi_i \geq 0, \quad y_i (\mathbf{w}^T \Phi(\mathbf{x}_i) + w_0) \geq 1 - \xi_i \end{aligned} \tag{4.20}$$

Here ξ_i's are the "slack" variables, i.e. the amount by which a misclassified labeled sample $\Phi(\mathbf{x}_i)$ is on the wrong side and c_i is the cost associated with that sample. This way, unlike most other EM based iterative semi-supervised learning algorithms labeled data are more directly and effectively used in determining the discriminant. The higher the ξ_i for a given sample the more that sample will dominate in the cost function.

Using (4.10), (4.20) can be restated as follows,

$$\begin{aligned} &\min_{\mathbf{a},a_0} \quad \mathbf{a}^T N \mathbf{a} + \sum_{i=1}^{l} c_i \xi_i \\ &\text{subj to} \quad \mathbf{a}^T \mathbf{d} = a \\ &\xi_i \geq 0, \quad y_i (\mathbf{a}^T \mathbf{k}_i + a_0) \geq 1 - \xi_i \end{aligned} \tag{4.21}$$

where $K = [\mathbf{k}_1, \ldots, \mathbf{k}_l]$ is the l-dimensional kernel (gram) matrix whose $(ij)^{th}$ element is obtained by $k(\mathbf{x}_i, \mathbf{x}_j)$. The corresponding Lagrange equation for (4.21) is,

$$L(\mathbf{a}, a_0, \xi_i) = \mathbf{a}^T N \mathbf{a} + \sum_{i=1}^{l} c_i \xi_i + \lambda (\mathbf{a}^T \mathbf{d} - b) - \sum_{i=1}^{l} \lambda_i^1 \left(y_i (\mathbf{a}^T \mathbf{k}_i + a_0) + \xi_i - 1 \right) - \sum_{i=1}^{l} \lambda_i^2 \xi_i \tag{4.22}$$

where γ is the Lagrange multiplier and μ_i^1 and μ_i^2 are the Karush-Kuhn-Tucker (KKT) multipliers [48]. Differentiating the Lagrange equation with respect to $\mathbf{a}$, a_0, ξ_i and equating the resultant equations to zero we get,

$$\mathbf{a} = \frac{1}{2}N^{-1}\left[\sum_{i=1}^{l} \mathbf{k}_i y_i \alpha_i^1 - \gamma \mathbf{d}\right] \tag{4.23}$$

$$\sum_{i=1}^{l} y_i \alpha_i^1 = 0 \tag{4.24}$$

$$c_i = \mu_i^1 + \mu_i^2 \tag{4.25}$$

Next using the equality constraint $\mathbf{a}^T\mathbf{d} = b$ and (4.23) we express the Lagrange multiplier in terms of α_i^1 as follows,

$$\gamma = \frac{\mathbf{d}^T N^{-1}\sum_{i=1}^{l} \mathbf{k}_i y_i \alpha_i^1 - 2b}{\mathbf{d}^T N^{-1}\mathbf{d}} \tag{4.26}$$

Then we substitute (4.23), (4.24), (4.25) and (4.26) into (4.21) and obtain the following cost function,

$$J(\alpha^1) = -\frac{1}{2}\alpha^{1T} H \alpha^1 - \mathbf{f}^T \alpha^1 \tag{4.27}$$

where,

$$H = \left(\frac{1}{2}\Omega^T N^{-1}\Omega\right).*\left(\mathbf{y}\mathbf{y}^T\right) \tag{4.28}$$

$$\mathbf{f} = \frac{b\mathbf{d}N^{-1}(\Omega + K)}{\mathbf{d}^T N^{-1}\mathbf{d}}.*\mathbf{y} \tag{4.29}$$

$$\Omega = \left(I - \frac{\mathbf{d}\mathbf{d}^T N^{-1}}{\mathbf{d}^T N^{-1}\mathbf{d}}\right)K \tag{4.30}$$

$\alpha^1 = [\alpha_1^1, K\ , \alpha_l^1]^T$, $\mathbf{y} = [y_1, K\ , y_l]^T$ and $.*$ denotes term by term multiplication.

Finally the dual of the optimization problem in (4.21) can be stated as follows,

$$\begin{aligned} \min_{\alpha^1} \quad & J(\alpha^1) \\ \text{subj. to} \quad & \sum_{i=1}^{l} y_i \alpha_i^1 = 0 \\ & 0 \le \alpha_i \le c_i \\ & \alpha_i^1 \left(y_i \left(\mathbf{a}^T \mathbf{k}_i + a_0 \right) + \xi_i \right) = 0 \\ & \alpha_i^2 \xi_i = 0 \end{aligned} \tag{4.31}$$

In solving the optimization problem stated in (4.31) we benefited from the efficient algorithm introduced in [43]. This algorithm is introduced in the context of Support Vector Machines (SVMs) but it can easily be adapted to solve (21). The main idea is to partition the labeled data into two sets, namely working and nonworking sets. Then α_i^1's corresponding to the nonworking set is fixed and the optimization problem is solved on the working set only. This process is repeated until all α_i^1's satisfies the Karush-Kuhn-Tucker (KKT) conditions stated in (4.31). For details and proof on the convergence properties of this algorithm please see [43]. Once α^{1*} is found $\mathbf{a}^*$ is obtained by (4.23) and (4.26) and $\mathbf{w}^*$ is obtained by (4.10).

4.4 Classification Tasks and Experimental Results

4.4.1 Classification tasks

In this study we used the Washington DC Mall and Purdue Campus Data Sets shown in Figure 1.3 and 1.5 respectively. In these data sets the two classes we are particularly interested are "roof tops" and "roads and parking lots". These features are chosen because of their particular attributes in remotely-sensed data. Roof tops generally encompass quite broad area land-cover distinction. Compared to other features, roads are relatively difficult to classify by automated algorithms.

The main emphasis in this section will be on the effect of unlabeled data on the classifier performance. We will conduct several experiments to show that unlike Semi-supervised Kernel Fisher's Discriminant (S^2KFD) where the cost of a misclassified

labeled sample is inherently zero, the Cost-effective Semi-supervised Classifier (CES^2C) as proposed in this chapter can successfully incorporate unlabeled data and result in an iteratively improved classifier performance.

For each classification task a certain ratio of labeled data is randomly chosen and the entire data set is used as unlabeled data. In the first iteration using labeled training samples supervised-only counterparts of both classifiers are designed and the unlabeled data is classified by each classifier separately. In the next iteration both semi-labeled and labeled samples are used together to estimate statistics and CES^2C and S^2KFD are designed and unlabeled data is classified again. This process is iterated until almost no change in class membership of semi-labeled data is observed. This algorithm usually converges in less than twenty iterations.

3.4.2 Choosing the parameters

Throughout the experiments in this study we have used the Gaussian radial basis kernel function (RBF), which is defined by $k(\mathbf{x}, \mathbf{x}') = \exp(-\|\mathbf{x} - \mathbf{x}'\|^2 / \sigma^2)$. Apart from σ two other parameters to be estimated are the regularization constant λ in (4.19) and the cost factor, c_i. We considered different costs for each class. More specifically using a simple heuristic the cost values assigned to classes are set to be proportional with one minus the corresponding class prior probabilities, i.e. $c_k=C(1-p_i)$, where C is some scalar. This simply implies we are putting more emphasis on a sample originating from a class with a less number of labeled samples. The value of C and σ are estimated by cross-validation over the set of labeled training samples. The discrete range of values considered for these parameters are $C=[1\ 10^5]$, $\sigma=[0.01\ 10]$ respectively. The regularization constant λ is set to 0.5. Note that we only need σ in the first iteration as the number of samples used in statistics estimation in the succeeding iterations will be much higher than the dimensionality and hence no regularization will be needed.

3.4.3 Experimental results

The number of total labeled samples used in the training for the Campus dataset is 1 % (~200) and for the Washington DC Mall data set is 0.2 % (~56) of the total labeled samples. The classification tasks are carried out as outlined in section 4.4.1. This process is repeated ten times each time with a randomly chosen training set. In Figure 4.3a-4.3b the training and testing error rates obtained during a typical run of the algorithm is shown for both CES^2C and S^2KFD. In Figure 4.3c-4.3f the average testing error together with the standard deviation at each iteration is plotted for both classifiers for all the features considered. Finally the classification maps obtained during a single run of the algorithm (CES^2C) for the first and final iterations are shown in Figure 4.4 to Figure 4.15.

4.4.4 Discussion and analysis of experimental results

Using the proposed semi-supervised classifier and a very small training set for the features considered we have obtained noteworthy improvements over the supervised-only counterpart. This improvement comes in the form of reduced scatter as well as reduced classification error. When the classification maps are examined we see that most class fields, which are barely identifiable due to a large number of misclassified pixels in the first iteration, are readily seen with relatively well-defined and clear boundaries in the final iteration. Indeed these classification maps suggest the actual improvement in the final iteration over the first iteration might be more than what the quantitative values imply. Also observe how the semi-supervised Kernel Fisher's Discriminant, which also has the flexibility to model a wide-range of class conditional distributions but lacks the cost-effective term CES^2C has, performed poorly in all the cases considered. As Figure 4.3a-4.3b show, the training error rate corresponding to CES^2C is forced to be bounded within a certain range whereas there is no such mechanism that limits the training error in S^2KFD.

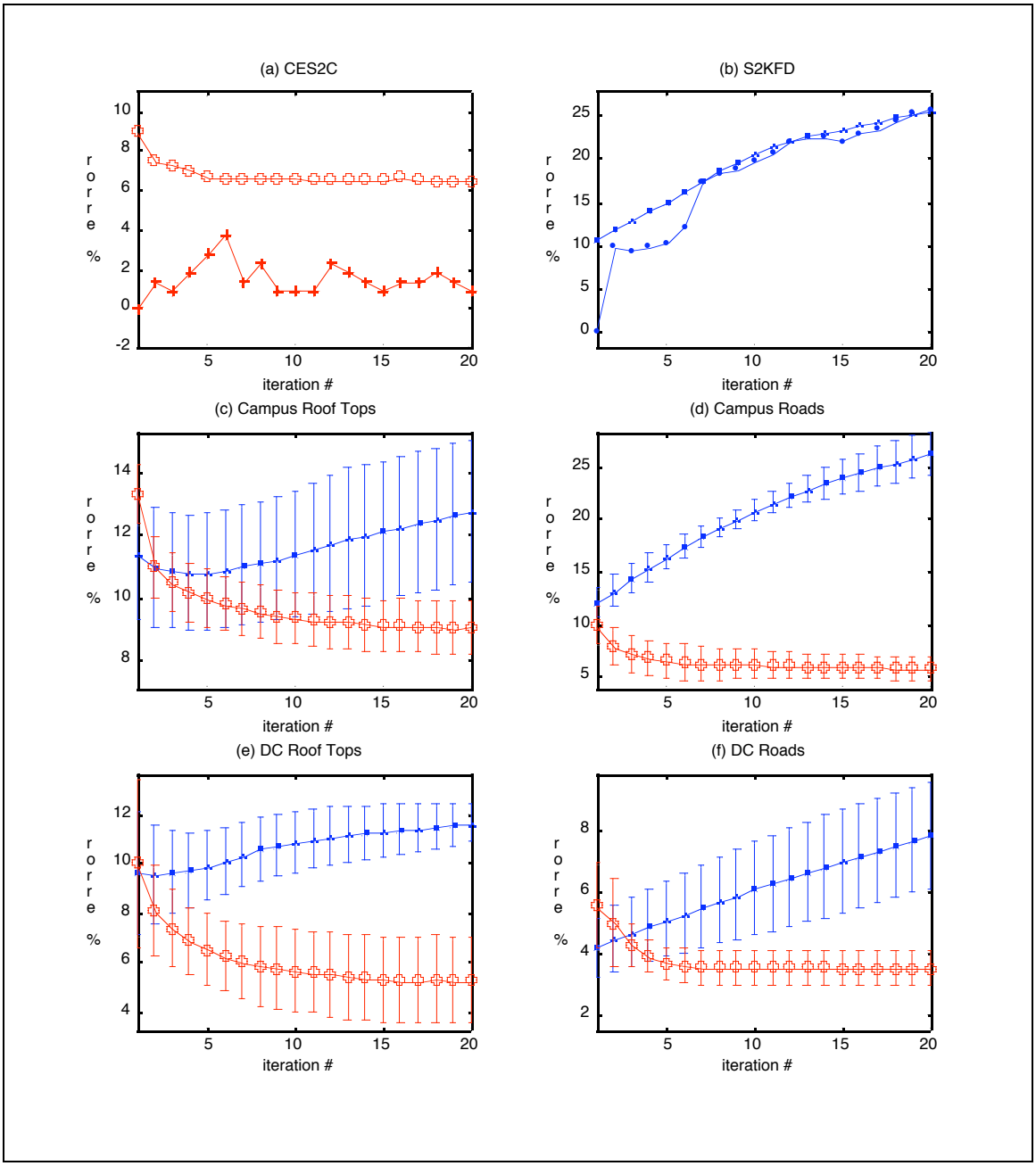


Fig. 4.3 Comparison of CES^{2}C and S^{2}KFD. (a)-(b) Testing and training error rates obtained during a typical run of the algorithm for CES^{2}C and S^{2}KFD respectively. (c)-(f) Testing error rates corresponding to CES^{2}C and S^{2}KFD for 'roof tops', 'roads' in Purdue Campus Data and 'roof tops', 'roads' in DC Mall Data respectively. (o – testing error for CES^{2}C, + – training error for CES^{2}C, x – testing error for S^{2}KFD, • – training error for S^{2}KFD, vertical bars depict error ranges)

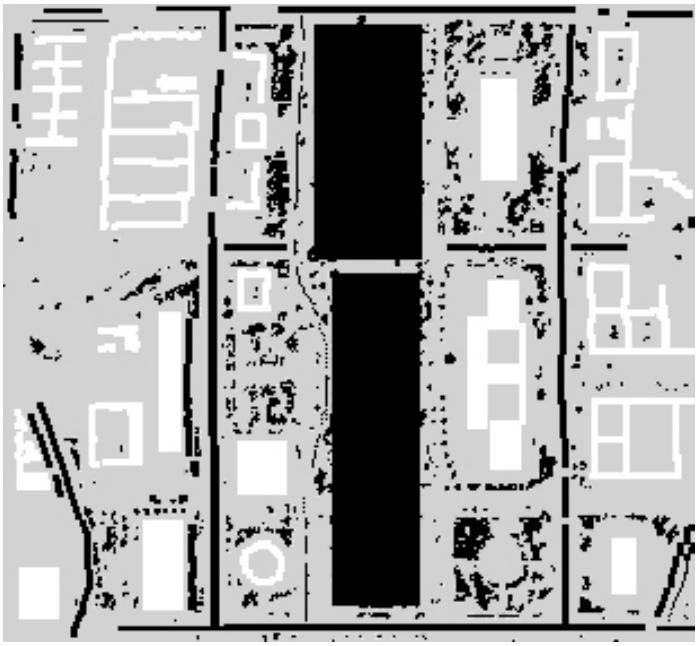


Fig. 4.4 Available ground truth data for "roof tops" in DC Mall Data Set
(White=roof tops, Black=Not roof tops, Gray= No Assertion)

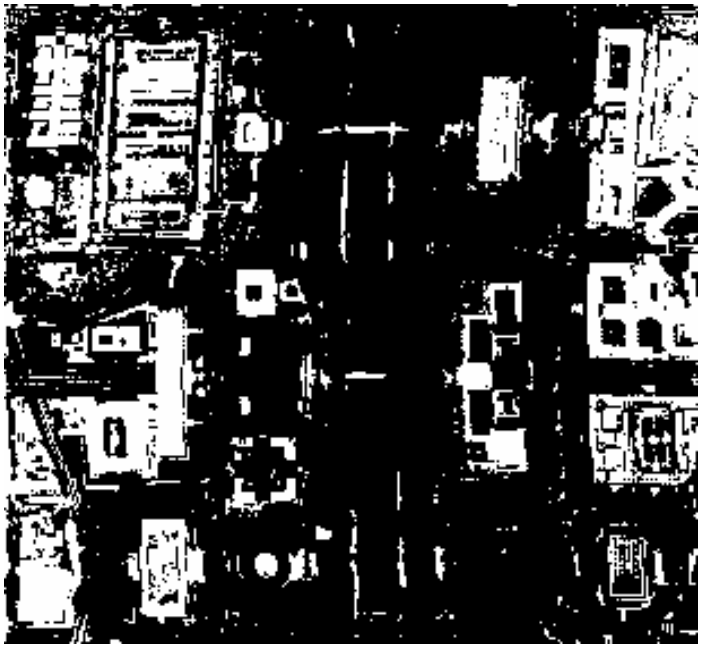


Fig. 4.5 Classification map obtained for "roof tops" in DC Mall
Data Set during the 1st iteration of CES^{2}C (error=12.9 %)

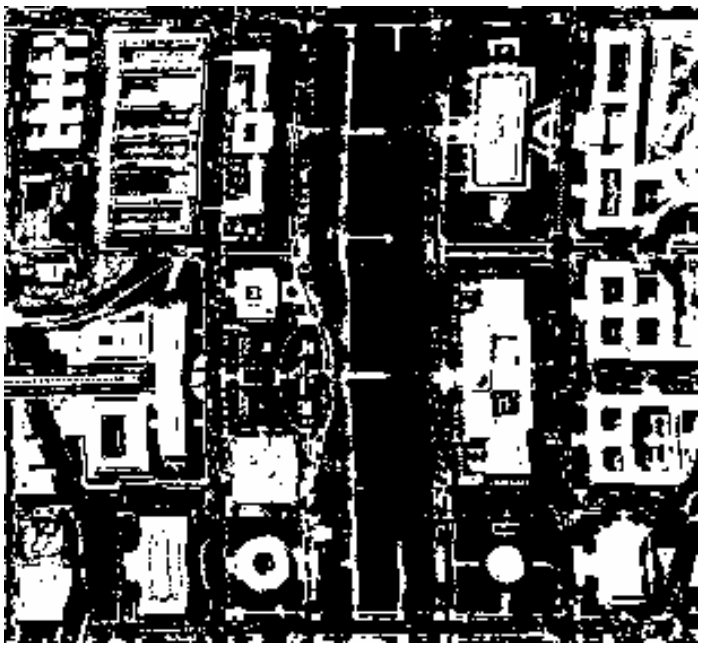


Fig. 4.6 Classification map obtained for "roof tops" in DC Mall Data Set during the 20th iteration of CES^{2}C (error=5.0 %)

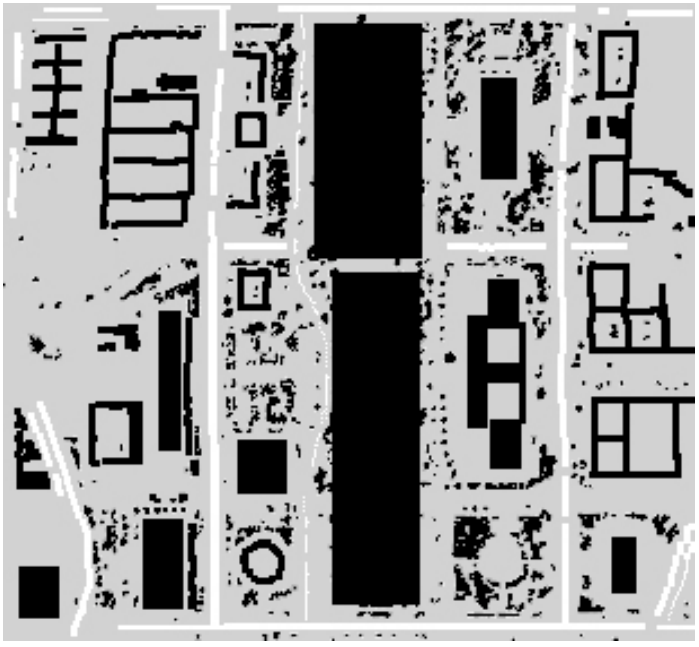


Fig. 4.7 Available ground truth data for "roads and parking lots" in DC Mall Data Set (White= roads and parking lots, Black=Not roads and parking lots, Gray= No Assertion)

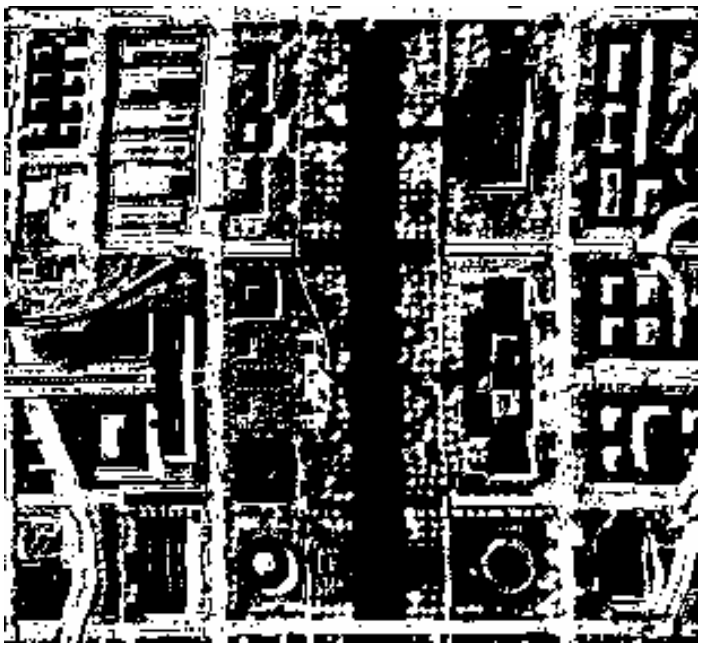


Fig. 4.8 Classification map obtained for "roads and parking lots" in DC Mall Data Set during the 1st iteration of CES^{2}C (error=8.5 %)

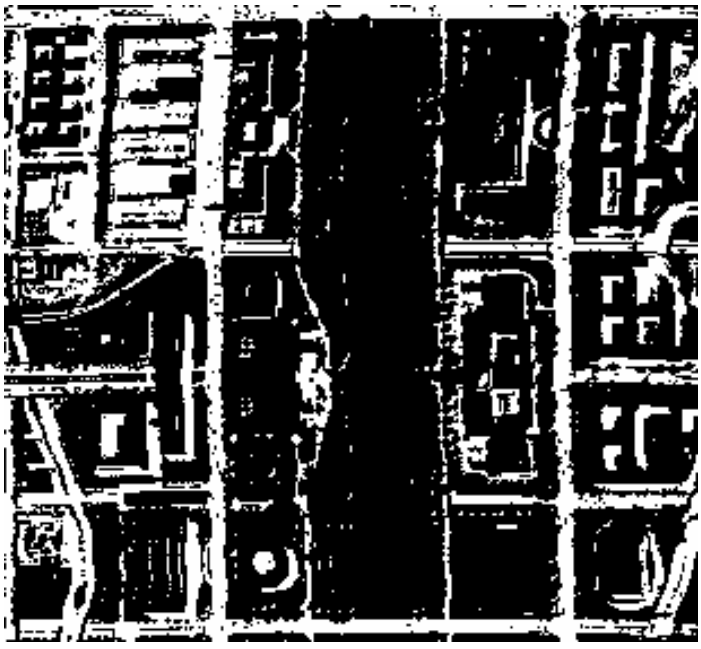


Fig. 4.9 Classification map obtained for "roads and parking lots" in DC Mall data during the 20th iteration of CES^{2}C (error=5.8 %)

Fig. 4.10 Available ground truth data for "roof tops" in Purdue Campus data (White= roof tops, Black=Not roof tops, Gray= No Assertion)

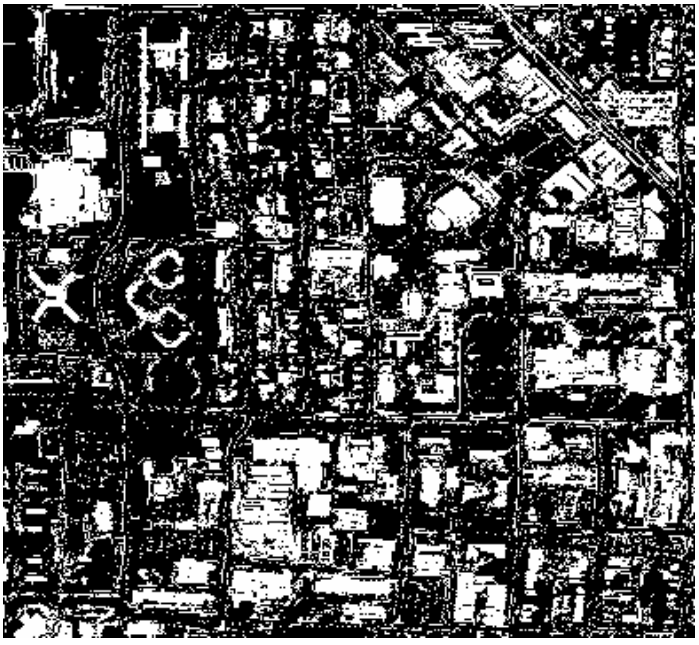


Fig. 4.11 Classification map obtained for "roof tops" in Purdue Campus data during the 1st iteration of CES^{2}C (error=12.2 %)

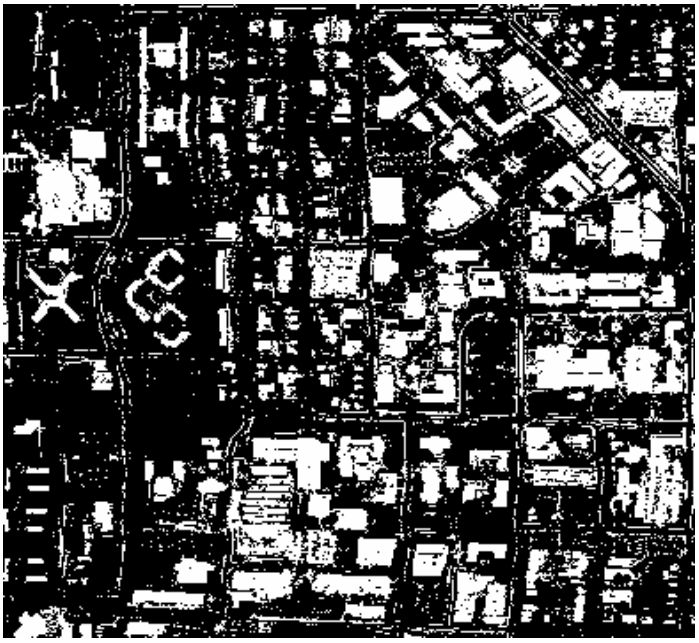


Fig. 4.12 Classification map obtained for "roof tops" in Purdue Campus data during the 20th iteration of CES^{2}C (error=8.3 %)

Fig. 4.13 Available ground truth data for "roads and parking lots" in Purdue Campus data (White= roads and parking lots, Black=Not roads and parking lots, Gray= No Assertion)

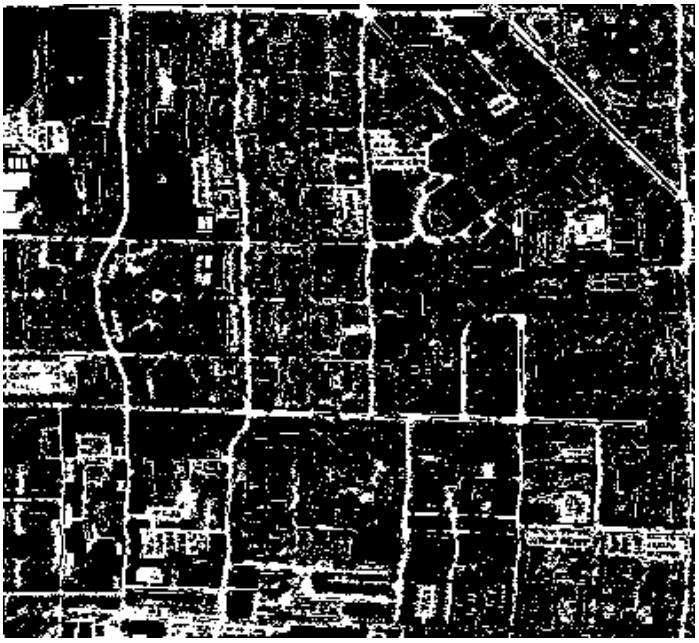


Fig. 4.14 Classification map obtained for "roads and parking lots" in Purdue Campus data during the 1st iteration of CES^{2}C (error=9.0 %)

Fig. 4.15 Classification map obtained for "roads and parking lots" in Purdue Campus data during the 20th iteration of CES^{2}C (error=6.5 %)

4.5 Conclusions

When there is only a limited number of labeled samples and a huge set of semi-labeled samples, semi-labeled samples might eventually dominate the statistics estimation. This is fine when the classifier is tolerable to some error in the classification of semi-labeled samples. However this is hardly the case in real-world problems. The semi-labeled samples that are misclassified at a given iteration are usually the ones closest to the classification boundary and hence are the ones that will make the most dramatic effect on the statistics estimation in the next iteration. For this reason classifiers are not very tolerant of even a small error in semi-labeled samples unless a perfect match between the classifier model and the model that generated the data exists.

The Kernel concept provides the flexibility needed to capture a variety of nonlinearities that might be available in real-world data sets. However with a limited number of labeled samples the amount of flexibility obtainable is also limited as the kernel parameters are estimated using labeled samples. The best we can hope is to capture the nonlinearities disclosed by the current set of labeled samples. However a large set of semi-labeled data introduced will likely demand additional complexity in the classifier model. At this point one can choose to estimate the kernel parameters using the labeled and semi-labeled samples together, but this is not practical as aforementioned. The proposed cost-effective semi-supervised classifier assigns a certain cost to a misclassified labeled sample and tries to balance the effect of unlabeled data by directly controlling the number of and the amount by which labeled samples are misclassified whereas this cost in the semi-supervised kernel Fisher's discriminant is inherently zero implying that no extra attention is paid to how the labeled samples are being classified in the semi-supervised framework.

In this study rather than focusing our attention on how the unlabeled samples should be weighted we closely examined how the role of labeled samples in determining the classification boundary can be enriched. Towards this end we propose an efficient technique which successfully incorporates unlabeled data into the design of the classifier by counter balancing any possible adverse effect of unlabeled data on the classifier

performance using a more direct and efficient way of controlling the number of misclassified labeled samples as well as the amount by which they are on the wrong side of the boundary. For the data sets considered we observed a considerable amount of improvements in the performance of the classifier. This improvement is often observed in the form of reduced scatter as well as reduced classification error.

The proposed technique doesn't require a large labeled data set. Throughout the experiments we have used limited numbers of training samples and with this many samples we were able to obtain on the average 25-50% improvement in the final iteration over the first iteration. Even though in most real world problems it is not very hard to acquire this many labeled samples the proposed approach might still perform well with a smaller set of labeled samples provided that the cost associated with each misclassified labeled sample is increased. The less the number of labeled samples the more valuable they are, and hence assigning a higher cost to each misclassified labeled sample is justified.

From a computational perspective things don't look so bright. Both **N** and **d** in equation (4.11a) and (4.11b) are l-dimensional where l is the number of labeled samples. Hence as we increase the number of labeled samples the algorithm becomes computationally more demanding. However with a few hundred labeled and a few thousand unlabeled samples the algorithm is tractable to a greater extent. A single run of the algorithm (a cycle of 20 iterations) in the previous section took on the average 40 minutes on a PC with a processor clock of 1.2 GHz. When the number of labeled samples is very large one can still tackle this problem by expanding the solution in the kernel space in terms of only a small portion of labeled samples rather than all the samples as in equation (4.7). As long as labeled samples used in the expansion are chosen efficiently one can still obtain a comparable performance with much less computational effort.

Both labeled and semi-labeled samples were given full weight during the statistics estimation. Assigning reduced weight to semi-labeled samples might help improve the classifier performance further. In [32] this is done by assigning posterior probabilities as the weighting factors. However as in Support Vector Machines (SVM) for the proposed classifier converting the output of the discriminant function to posterior

probabilities is not easy if not impractical. Towards this end no widely accepted method is available but the approaches introduced in [22, 44] are worth further consideration.

5. CONCLUSIONS AND FUTURE RESEARCH SUGGESTIONS

In Chapter 2 we modeled each class distribution using a mixture of normal distributions while constraining the covariance matrix of each component to a less sophisticated covariance model. Apart from providing us the flexibility to characterize a wide range of distributions this technique significantly reduces the number of parameters to estimate in a mixture model and helped us to obtain more robust estimates of the statistics. For the data we investigated we have obtained significant improvements over the simple quadratic classifier, especially when the training sample size was large. However in the presence of limited labeled samples results were not very promising. The initial training data size should be large enough to allow us to identify an efficient set of mixture components and at the same time to estimate their statistics without stumbling on numerical issues. Having this fact in mind we proposed another methodology in Chapter 3 to characterize the class distributions.

The Kernel Linear Classifier of Chapter 3 gives us the flexibility required to model complex data structures that originate from a wide-range of class conditional distributions. The fact that the statistics estimation is performed at full dimensionality (no feature extraction is needed) allows us to exploit all the separability the data provides without stumbling on numerical issues. As the results suggest, seemingly simple linear piece-wise decision boundaries in the feature space correspond to powerful nonlinear boundaries in the input space. Unlike the Mixture Classifier of Chapter 2 this technique performs well even when we have only a few training samples.

The theoretical support and the strong empirical results suggest this classifier can be an optimal candidate classifier for the analysis of hyperspectral data. However before introducing this classifier as a powerful alternative to other state-of-the-art classifiers some improvements should be accomplished from the computational perspective. The algorithm is currently computationally too demanding to be used with a large training data set. A significant amount of time is spent on determining the classifier parameters

through cross-validation. Perhaps a competitive technique with less computational burden can be found. Toward this end structural risk minimization [22] is definitely worth consideration.

Exploring some alternative ways a given vector is expanded in the feature space is yet another way the computational complexity of the algorithm can be reduced. Unfortunately the technique introduced in Chapter 3 is not "sparse", i.e. the eigenvectors are expressed in terms of every training sample. For very large sizes of training sets this becomes quite problematic and makes the algorithm intractable. A simple heuristic solution to this problem would be to express the eigenvectors of the common covariance matrix only in terms of a small portion of the training set. This definitely saves us substantial amount of computational time. By doing so we a priori assume all but a small number of expansion parameters to be zero. However in order not to sacrifice the performance to significantly, a theoretically sound approach is needed. In [45] a technique for obtaining sparse kernel principal component analysis is proposed. This methodology is worth exploiting for its possible generalization to kernel-based classifiers.

In Chapter 4 we propose a cost-effective semi-supervised binary classifier which, for the data we investigated, showed encouraging performance. This is yet another technique where we successfully benefited from the kernel concept. Indeed the implicit nonlinear transformation we suggest, and which we implemented using the kernel concept, is the core part of Chapter 3 and 4.

In a typical semi-supervised setting when there is a limited number of labeled samples and a huge set of semi-labeled samples, semi-labeled samples might eventually dominate the statistics estimation. This is fine when the classifier is tolerant to some error in the classification of semi-labeled samples. However this is hardly the case in real-world problems. The flexibility we acquired through using the kernel concept mitigates this problem to some extent, but as the empirical results with the semi-supervised kernel Fisher's discriminant suggest, this alone is not sufficient for improvement in the classifier performance.

Another problem with a typical semi-supervised classifier using the kernel concept is that the amount of flexibility obtainable using a limited number of labeled samples is also limited as the classifier parameters are estimated using labeled samples. The best we can hope for by doing so is to capture the nonlinearities disclosed by the current set of labeled samples. The question that arises in this consideration is how close this classifier model to the actual model is. A large set of semi-labeled data introduced will likely demand additional complexity in the classifier model. Therefore another question in this setting is how efficient estimating the classifier parameters using only labeled samples would be.

In this study rather than focusing our attention on how the unlabeled samples should be weighted or how a perfect match between the classifier and the actual models can be obtained we closely examined how the role of labeled samples in determining the classification boundary can be enriched. Towards this end we propose an efficient technique which successfully incorporates unlabeled data into the design of the classifier by counter balancing any possible adverse effect of unlabeled data on the classifier performance using a more direct and efficient way of controlling the number of misclassified labeled samples as well as the amount by which they are on the wrong side of the boundary. The proposed cost-effective semi-supervised classifier assigns a certain cost to a misclassified labeled sample whereas this cost in the semi-supervised kernel Fisher's discriminant is inherently zero implying that no extra attention is paid to how the labeled samples are being classified in the semi-supervised framework. For the data sets considered we observed a considerable amount of improvement in the performance of the classifier. This improvement is often observed in the form of reduced scatter as well as reduced classification error.

REFERENCES

[1] P.H. Swain and Davis S. M., eds., *Remote Sensing: The Quantitative Approach,* McGraw Hill, 1978, Chapter 2.

[2] L. Rickard et. all., "HYDICE: An airborne system for hyperspectral imaging," *SPIE* Vol. 1937, 1993.

[3] F. A. Kruse, J. W. Boardman, A. B. Lefkoff, J. M. Young, K. S. Kierein-Young, T. D. Cocks, R. Jensen, P. A. Cocks, HyMap: An Australian Hyperspectral Sensor Solving Global Problems - Results from USA HyMap Data Acquisitions: *In Proc. of the 10th Australasian Remote Sensing and Photogrammetry Conference*, 2000.

[4] Purdue University Campus Map (http://www.eas.purdue.edu/map/campus/)

[5] D. A. Landgrebe, "On the Relationship Between Class Definition Precision and Classification Accuracy in Hyperspectral Analysis", *Proceedings of the International Geoscience and Remote Sensing Symposium*, Honolulu, Hawaii, 2000.

[6] K. Fukunaga, *Introduction to Statistical Pattern Recognition,* San Diego, CA: Academic Press, 1990.

[7] J. P. Hoffbeck and D. A. Landgrebe, "A Method for Estimating the Number of Components in a Normal Mixture Density Function", *Proceedings of the International Geoscience and Remote Sensing Symposium*, Honolulu, Hawaii, 2000.

[8] R. E. Kass and A. Raftery, "Bayes Factors", *Journal of the American Statistical Association,* 90:773-795, ? 1995.

[9] G. J. McLachlan and T. Krishnan, *The EM Algorithm and Extension*, Wiley, 1997.

[10] Warren L. G. Koontz, Patrenahalli M. Narendra, Keinosuke Fukunaga, "A Branch and Bound Clustering Algorithm", *IEEE Trans. Computers*, V. 24, September 1975.

[11] J.P. Hoffbeck and D.A. Landgrebe, "Covariance Matrix Estimation and Classification with Limited Training Data" *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol 18, NO. 7, pp. 763-767, July 1996.

[12] Saldju Tadjudin and David Landgrebe, "Covariance Estimation With Limited Training Samples," *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 37, No. 4, pp. 2113-2118, July 1999.

[13] Bor-Chen Kuo and David A. Landgrebe, "A Covariance Estimator for Small Sample Size Classification Problems and Its Application to Feature Extraction**,"***IEEE Transactions on Geoscience and Remote Sensing,* Vol. 40, No. 4, pp 814-819, April 2002.

[14] C. Fraley and A. E. Raftery, "How Many Clusters? Which Clustering Method? Answers via Model-Based Cluster Analysis", *Computer Journal,* 41:578-588, ? 1998.

[15] C. Fraley and A. E. Raftery, "Model-Based Clustering, Discriminant Analysis, and Density estimation", *Journal of the American Statistical Association 97:611-631, 2002*.

[16] Bor-Chen Kuo and David A. Landgrebe "Hyperspectral Data Classification Using Nonparametric Weighted Feature Extraction," International Geoscience and Remote Sensing Symposium, Toronto, Canada, 2002.

[17] G. F. Hughes, "On the mean accuracy of statistical pattern recognition," *IEEE Trans. on Information Theory*, 1968, Vol. IT-14, No. 1, pp 55-63.

[18] K. Nigam, A. McCallum, S. Thrun, and T. Mitchell,"Text Classification from Labeled and Unlabeled Documents using EM," *Machine Learning*, vol. 39 (2/3), pp. 103-134, 2000.

[19] B.M. Shahshahani and D.A. Landgrebe, "The effect of unlabeled samples in reducing the small sample size problem and mitigating the Hughes Phenomenon," *IEEE Trans. Geoscience and Remote Sensing*, vol. 32, no. 5, pp. 1087-1095, 1994.

[20] M. M. Dundar and D. Landgrebe, "A model based mixture classification approach in hyperspectral data analysis," to be published in *IEEE Trans Geoscience and Remote Sensing*.

[21] T. Hastie and R. Tibshirani, "Discriminant Analysis by Gaussian Mixtures," *Journal of the Royal Statistical Society*, series B, 1996.

[22] V. Vapnik, S*tatistical Learning Theory*, New York: Wiley, 1998.

[23] B. Scholkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond*, Cambridge, MA: MIT Press, 2002, pp. 189 – 212.

[24] C. J. C. Burges. “A Tutorial on Support Vector Machines for Pattern Recognition,” *Data Mining and Knowledge Discovery*, vol. 2, pp. 955-974, 1998.

[25] S. Mika, G. Ratsch, J. Weston, B. Scholkopf, and K. R. Muller, “Fisher discriminant analysis with kernels,” *Neural Networks for Signal Processing IX IEEE*, 1999, pp. 41-48.

[26] N. Lawrence and B Scholkopf, "Estimating a Kernel Fisher Discriminant in the Presence of Label Noise," *in Proc. of 18th International Conference on Machine Learning*, 2001, pp. 306-313.

[27] J. Ma et. all., "Modified Kernel-Based Nonlinear Feature Extraction," *Proc. International Conference on Machine Learning and Applications*, Las Vegas, June 24-27, 2002.

[28] B. Scholkopf et al., "Input Space vs. Feature Space in Kernel-Based Methods," *IEEE Trans. on Neural Networks*, vol. 10, no. 5, pp. 1000-1017, 1999.

[29] J. A. Gualtieri, S. R. Chettri, R. F. Cromp, and L. F. Johnson, “Support Vector Machine Classifiers as Applied to AVIRIS Data,” Eighth JPL Airborne Earth Science Workshop, Feb 8-11, 1999.

[30] S. Perkins et. all., Support Vector Machines for Broad Area Feature Extraction in Remotely Sensed Images. *Proc. SPIE 4381*, 2001.

[31] A. J. Gualtieri and R.F. Cromp, “Support Vector Machines for Hyperspectral Remote Sensing Classification,” *Proc. SPIE 3584*, 221-232, 1999.

[32] Q. Jackson and D. A. Landgrebe, "An Adaptive Classifier Design for High-Dimensional Data Analysis with a Limited Training Data Set,” *IEEE Trans. Geoscience and Remote Sensing*, vol. 39, no. 12, pp. 2664-2679, 2001.

[33] T. Joachims, "Transductive Inference for Text Classification using Support Vector Machines," in Proc. of the International Conference on Machine Learning (ICML), 1999.

[34] A. Demiriz, K. P. Bennett, "Semi-Supervised Support Vector Machines," *Advances in Neural Information Processing Systems*, Cambridge, MA: MIT Press, 1998, pp. 368-374.

[35] G.Fung and O.L. Mangasarian, "Semi-Supervised Support Vector Machines for Unlabeled Data Classification,” Data Mining Inst., Tech. Rep. 99-05, Oct, 1999.

[36] A. Blum and T. Mitchell, "Combining Labeled and Unlabeled Data with Co-Training," *in Proc. of the 11th Annual Conference on Computational Learning Theory*, 1998, pp. 92-100.

[37] F. G. Cozman and I. Cohen, "Unlabeled Data Can Degrade Classification Performance of Generative Classifiers," *in Proc. of 15th International FLAIRS Conference*, 2002, pp. 327-331.

[38] T.Zhang and F. Oles, "A probability analysis on the value of unlabeled data for classification problems," *in Proc. of the 17th International Conference on Machine Learning*, 2000, pp. 1191-1198.

[39] V. Castelli and T.M. Cover, "The Relative Value of Labeled and Unlabeled Samples in Pattern Recognition with an Unknown Mixing Parameter," *IEEE Trans Information Theory*, vol. 42, no.6, pp.2102-2117, 1996.

[40] V. Castelli and T. M. Cover, "On the exponential value of labeled samples," *Pattern Recognition Letters*, vol. 16, pp.105-111, 1995.

[41] A. O. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," Journal of the Royal Statistical Society, Series B, vol. 39, no. 1, pp. 1-38, 1977.

[42] G. J. McLachlan and K. E. Basford, *Mixture Models*, New York: Marcel Dekker, 1988.

[43] E. Osuna, R. Freund, F. Girosi, "An improved training algorithm for Support Vector Machines," *in the Proc. of IEEE NNSP*, 1997, pp. 24-26.

[44] G. Wahba, "Support vector machines, reproducing kernel Hilbert spaces and the randomized GACV," *Advances in Kernel Methods*, Cambridge, MA: MIT Press, 1999, pp. 69-88.

[45] M. E. Tipping, "Sparse kernel principal component analysis," *Advances in Neural Information Processing Systems 13*, MIT Press, 2001.

[46] J. Shawe-Taylor, N. Christianini, "Further results on the margin distributions," *In Proc. of the 12th Annual Conference on Computational Learning Theory*, pp. 278-285, 1999.

[47] D. A. Landgrebe, *Signal Theory Methods in Multispectral Remote Sensing*, John Wiley and Sons, 2003.

[48] E. K. P. Chong and S. H. Zak, *An Introduction to Optimization*, New York, NY: Wiley, 1996