A NONPARAMETRIC BAYESIAN PERSPECTIVE FOR MACHINE LEARNING IN PARTIALLY-OBSERVED SETTINGS

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Ferit Akova

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

August 2013

Purdue University

West Lafayette, Indiana

To my beloved mother.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

SYMBOLS

Variable	Description
$j = 1 : J$	index for groups
$k = 1 : K$	index for components
$i = 1 : n_{j.}$	index for samples in group j
n_{jt}	number of samples in cluster t of group j
$n_{j.}$	number of samples in group j
$n_{.k}$	number of samples generated by component k across all groups
$m_{j.}$	number of clusters in group j
$m_{.k}$	number of clusters associated with component k across all groups
$m_{..}$	number of clusters across all groups
x_{ji}	sample i in group j
t_{ji}	cluster indicator variable for sample i in group j
k_{jt}	component indicator variable for cluster t in group j
θ_{ji}	component parameters associated with cluster t_{ji}
ψ_{jt}	component parameters associated with cluster t in group j
ϕ_k	distinct set of component parameters
n_{jt}^{-i}	number of samples in cluster t in group j excluding sample i
$m_{.k}^{-jt}$	number of clusters sharing the same component ϕ_k excluding cluster t in group j
H	the base distribution for the HDPM model

Variable	Description
α and γ	precision parameters for the HDPM model
Σ_0	scale matrix of the inverse Wishart prior for covariance matrices
m	degrees of freedom of the inverse Wishart prior for covariance matrices
μ_0	mean of the Normal prior for component mean vectors
κ_0	scaling constant for the covariance matrix of the Normal prior for component mean vectors
D_{jt}	subset of samples in cluster t in group j
D_k	subset of samples in clusters associated with component k
μ_{jt} and Σ_{jt}	mean vector and the covariance matrix of cluster t in group j
$\bar{\boldsymbol{x}}_{jt}$ and S_{jt}	sample mean and covariance matrix of cluster t in group j
$\bar{x}_k$ and S_k	sample mean and covariance matrix of component k
$\hat{\mu}$	location vector for the student-t distribution
$\hat{\Sigma}$	scale matrix for the student-t distribution
v	degrees of freedom for the student-t distribution

ABSTRACT

Akova, Ferit Ph.D., Purdue University, August 2013. A Nonparametric Bayesian Perspective for Machine Learning in Partially-Observed Settings. Major Professor: Mehmet M. Dundar.

Robustness and generalizability of supervised learning algorithms depend on the quality of the labeled data set in representing the real-life problem. In many real-world domains, however, we may not have full knowledge of the underlying data-generating mechanism, which may even have an evolving nature introducing new classes continually. This constitutes a partially-observed setting, where it would be impractical to obtain a labeled data set exhaustively defined by a fixed set of classes. Traditional supervised learning algorithms, assuming an exhaustive training library, would misclassify a future sample of an unobserved class with probability one, leading to an ill-defined classification problem. Our goal is to address situations where such assumption is violated by a non-exhaustive training library, which is a very realistic yet an overlooked issue in supervised learning.

In this dissertation we pursue a new direction for supervised learning by defining self-adjusting models to relax the fixed model assumption imposed on classes and their distributions. We let the model adapt itself to the prospective data by dynamically adding new classes/components as data demand, which in turn gradually make the model more representative of the entire population. In this framework, we first employ suitably chosen nonparametric priors to model class distributions for observed as well as unobserved classes and then, utilize new inference methods to classify samples from observed classes and discover/model novel classes for those from unobserved classes.

This thesis presents the initiating steps of an ongoing effort to address one of the most overlooked bottlenecks in supervised learning and indicates the potential

for taking new perspectives in some of the most heavily studied areas of machine learning: novelty detection, online class discovery and semi-supervised learning.

1 INTRODUCTION

The goal of machine learning is to build robust models based on observed data that, when deployed in real-life applications, generalize well to as-yet unseen examples of the sample population. Two major paradigms heavily studied in machine learning are *supervised learning* and *unsupervised learning*. In supervised learning each data point is coupled with a label, generally indicating a class membership or a function output, where the goal is to infer a mapping from the data into labels and employ it in predicting labels of (existing or future) unlabeled data points. In unsupervised learning samples do not have labels, where the goal is to identify *patterns* or *substructures* in the data and to describe or represent the data by those patterns/substructures.

Among the many factors that influence the generalizability of a learning algorithm, an exhaustive training dataset is perhaps the most critical. A training dataset is exhaustive if it contains samples from all classes of informational value. When some of the classes are not yet known and hence not represented, the resulting training dataset is non-exhaustive and the associated learning problem is ill-defined. In this case, a sample from a class unknown at the time of training will be always incorrectly classified into one of the existing classes.

The easiest way to tackle with this problem is to ignore it, as most traditional algorithms do. This could be a viable option, when the prior probability of samples originating from an unrepresented class is low and/or the misclassification cost is negligible. However, for critical applications, this strategy could be potentially too costly to be a serious alternative.

1.1 Partially-Observed/Nonexhaustive Training Data

In many supervised learning settings the labeled data is not only difficult and costly to obtain but also collected without full knowledge of the underlying components of the data-generating mechanism. The main challenge that arises in the mining of real-world data sets but is often overlooked in supervised as well as semi-supervised learning is that the data model is not only unknown at the time of training but may also have an evolving nature that makes learning with a fixed model impractical. Under such circumstances it would be unrealistic to assume that training and prospective data sets come from the same distribution, because certain aspects of the data-generating mechanism evident at the time the prospective data are observed may not be evident at the time the training data set is collected. As a result of this intrinsic difference between data sets observed at different time points, it is natural to have a training data set where the set of classes is non-exhaustively defined, i.e. partially observed. It is impractical, often impossible, to define a training data set with a complete set of classes and then collect samples for each class, mainly because: i) some of the classes may not be in existence at the time of training, ii) they may exist but are not known, or iii) their existence may be known but samples are simply not obtainable. A classifier trained with a non-exhaustive data set misclassifies all samples of unrepresented classes with a probability one, making the associated learning problem ill-defined.

In this thesis, we present a new perspective for supervised learning to address the described problem, bringing forth a number of advantages in handling data sets with evolving nature. Here the term evolving does not merely imply time-dependent or streaming data sets, but rather indicates the possibility of discovering other 'previously inaccessible' classes as well as formation of new ones. The proposed framework incorporates supervised classification with class discovery and modeling. The specific research goals that we targeted can be summarized as follows:

1. Training a classifier with a non-exhaustive training set to detect samples of unrepresented classes, i.e., novelties, with a high sensitivity while classifying future samples of represented classes with an acceptable accuracy.

2. Defining a prior model over the class distributions to enable easy incorporation of the domain knowledge to facilitate real-time class discovery and modeling.

3. Exploiting additional information introduced by the newly discovered classes to improve the predictive performance of the classifier for future samples.

All three problems are highly connected and we have to treat them jointly to develop a robust non-exhaustive learning system. More specifically, as we discover and accurately model more classes of informational value, the evolving model will gradually become more representative of the entire population. This, in turn, will improve the performance for detecting novelties as well as classifying future samples of previously discovered classes. In other words, it will result in a *self-adjusting model* for partially-observed settings to better accommodate more incoming data.

In what follows we describe three scientific applications with nonexhaustively defined training data sets in nature that motivated the presented research in this thesis.

1.2 Motivating Real-World Applications

Bacteria/Pathogen Detection

A global surge in the number of outbreaks together with elevated concerns about biosecurity has led to an enormous interest among scientific communities and government agencies in developing label-free, i.e., reagentless, techniques for rapid identification of pathogens. The core advantage of label-free methods is their ability to quantify phenotypes for which there are no available antibodies or genetic markers. This information can be used within a traditional supervised-learning framework in which knowledge discovered from independently tested and prelabeled samples is used for training. However, the quality of training libraries is potentially limited because

the sheer number of bacterial classes would not allow for practical and manageable training in a traditional supervised setting; for instance Salmonella alone has over 2400 known serovars. Additionally, microorganisms are characterized by a high mutation rate, which means that new classes of bacteria can emerge anytime. Thus, no matter how diligently the labeled data set is collected, the evolving nature of the problem does not allow for obtaining an exhaustively-defined training data set.

Hyperspectral Data Analysis

New sensor technology has made it possible to gather hyperspectral images in hundreds and potentially thousands of spectral bands. This increased spectral resolution has resulted in a tremendous increase in information density for remote-sensing imagery, facilitating the differentiation of land-cover types with only subtle structural differences and thus allowing for in-depth analysis of the scene. The widespread use of machine-learning techniques in the analysis of hyperspectral imagery is usually hindered by the lack of well-defined ground truth. Collecting ground-truth is a laborious task limited mainly by the manual labeling of the fields. The problem can get worse, especially when analyzing images of scenes that cannot be physically accessed, e.g., an enemy territory, or scenes with dynamic characteristics, e.g., urban fields. Under these circumstances, defining an exhaustive set of classes becomes impractical. The previously collected ground truth for similar scenes might allow for classification of broad land-cover types, but this comes at the expense of misclassifying fields belonging to undefined land-cover types into one of the existing types. Besides, this approach under-exploits the wealth of spectral information available in the imagery and does not allow for in-depth and high-level image analysis. In summary, the set of classes of informational value in hyperspectral image analysis is inherently non-exhaustive and like the pathogen detection problem presented above, robust analysis of hyperspectral data also requires new rigorous machine-learning approaches capable of addressing the non-exhaustiveness problem.

1.3 Previous Work Related to Nonexhaustive Learning

Early work related to the current study can be considered in 3 groups: i) Offline methods, ii) Online/Incremental learning methods and iii) Online Clustering with novelty detection.

Offline Methods - Anomaly/Novelty Detection

Offline methods related to nonexhaustive learning are anomaly detection [1–4] and novelty detection [5–7]. These techniques focus on the first problem stated in section 1.1, namely learning to detect novelties without any specific effort on differentiating them. So they do not possess the capability for online class discovery and modeling. Both anomaly detection and novelty detection deal with detecting samples not represented in the training set, however, anomalies are by definition samples that are peculiar, abnormal or difficult to classify. Since a group of detected anomalies do not necessarily possess informational value it is very difficult to model them. On the other hand, novelties originate from hidden, missing or not yet known classes and thereby have informational value. Novelty detection is also sometimes referred to in the literature as "novel class detection". Most of the early work on novelty detection is developed around one-class classification problems and uses either support estimation [8,9] or density-based models to identify novelties that are not represented in the training dataset.

Online/Incremental Learning

Online or incremental learning addresses the third problem in section 1.1 that is improving the prediction performance by combining the past and present data. Actually online learning develops sequential classification algorithms utilizing the current sample only to update the classifiers [10–17]. The focus is mainly on discriminative models with special emphasis on kernel based methods. However, discriminative

functions are modeled using all or a subset of the training samples so it is not trivial how to obtain update equations based on the current sample only. Also many of the studies in this field assume exhaustiveness of the initial training set.

Although there is similarity with nonexhaustive learning in terms of the sequential classification aspect, the difference lies in the way training updates occur: If a sample is classified to an existing class then the corresponding class parameters are updated. Otherwise, if the sample turns out to be a novelty then a new class is generated and the existing model is updated by augmenting that class into it.

Online Clustering with Novelty Detection

In this line of work Dirichlet process priors have been employed for online cluster modeling [18–25]. We benefited from the literature on online clustering with novelty detection. We used non-parametric prior models for novel class discovery in the same way as these techniques do so for online cluster modeling. However, a distinction arises in the way inference is performed. We work with data sets for which the structure is partially observed and implement inference techniques that take advantage of the observed part of the structure to discover the unobserved part. On the other hand, online cluster modeling deals with fully unobserved structures and performs inference in a fully unsupervised manner. In addition, most of the existing techniques in this line deal with text or streaming data. Thus, we found the overlap between the proposed work and early work on online clustering with novelty detection to be minimal.

In traditional novelty detection algorithms no immediate action is taken for novelties. Once detected, they are left for a follow-up analysis. However, novelties originate from classes of informational value which were not known at the time of training. Pooling novelties showing similar characteristics into individual clusters may potentially recover some of these classes and as more classes of informational value are introduced, the training model becomes more representative of the real population. This helps improve the predictive performance of the system not only for detecting

novelties but also for classifying future samples of newly discovered classes. In this thesis we take a Bayesian learning view to achieve this dynamic behavior by putting some trust on prior domain knowledge but relying more on the way that the data lead to.

1.4 Background in Bayesian Learning/Inference

Bayesian learning aims to capture the data generating mechanism underlying the observed data by incorporating any prior belief about the generative model, which can include class distributions, independence assumptions, auxiliary parameters etc. Assuming a good prior model that accurately represents the generative process for the data is crucial for solving complex real-world problems. The model structure, M, consists of some unknown *parameters*, Θ, as random variables and the usual *Bayesian* recipe to estimate the true parameters is: i) define *prior* probability distributions over model parameters—possibly in a hierarchical fashion, ii) acquire some real data, D, representative of the sample population, iii) infer the *posterior* probability distribution for the parameters given the observed data, $P(\Theta|D, M)$. Once we obtain the posterior distribution we can, for instance, make predictions for future data by averaging over the posterior, which yields the posterior mean; or make decisions by minimizing the expected loss using a loss function (Bayesian decision making).

Specifically, the prior distribution and the likelihood of the parameters on the available data are combined using the *Bayes' Rule* as:

$$P(\Theta|D, M) = \frac{P(\Theta|M)P(D|\Theta, M)}{P(D|M)} \tag{1.1}$$

where the denominator, i.e. *evidence*, is obtained by integrating over the parameters:

$$P(D|M_k) = \int_{\Theta} P(D|\Theta, M_k)P(\Theta|M_k) \tag{1.2}$$

It is also called the *marginal likelihood* since it does not depend on any parameters and serves as a normalizing constant in the actual computation, which is usually ignored to express it simply as a proportionality: $Posterior \propto Prior$ x $Likelihood$.

Usually, working with simple models and/or using conjugate distributions for the likelihood and the prior makes it possible to analytically obtain posteriors in closed-form. However, conjugate models can seldom describe the observed data and usually more flexible models are needed. In this case more complex distributions are employed, which cannot be represented using tractable formulas and are approximated by more computational *sampling methods* (randomly drawing a large number of values from a distribution), or by deterministic approximation methods.

In situations where choosing the best fitting model is difficult, *model selection* is performed by comparing different models based on their marginal likelihoods. To do so, for each model M_k, the marginal likelihood of the observed data is computed using (1.2). As a result, a single model can be chosen according to the trade-off between the computed value and the model complexity (i.e. the number of model parameters) to avoid over-fitting the observed data. Alternatively, predictions from multiple models can be combined as a weighted average based on the marginal likelihood values times the prior probability of each respective model.

Parametric and Nonparametric Models

Many real world problems require very flexible models and *parametric models* are limited in that sense since they are represented by a fixed number of parameters. Although we can get some more flexibility through *hierarchical models* by placing hyper-priors on the priors of the parameters themselves, the assumptions on the distributions may be too restrictive for accurately modeling the data. Nonetheless, the theory of finite mixture models [26] states that given enough components and under fairly weak assumptions, a mixture model can approximate a given density arbitrarily closely, allowing better flexibility. For example, even if the initially known

classes belong to a much complex distribution, the class-conditional distributions can still be estimated arbitrarily closely, using a mixture of Gaussians. A mixture of Gaussian subclasses can be learned for each class data through a process involving expectation maximization (EM) [27] and model selection. As an aside, in a wide range of applications it is customary to treat data of unknown nature by assuming *Normal (Gaussian) distribution* for all classes. Normal distribution is generally preferred due to its simplicity and analytical tractability as well as its suitability in modeling many natural and social phenomena—as it can be justified via the Central Limit Theorem. In this thesis we treated the aforementioned scientific problems by assuming single Gaussians or mixtures of Gaussians for the initial classes.

However, going back to learning with a partially-observed data set, the major problem is that regardless of how accurately the initial parametric model matches the available data, a clear mismatch with the unknown classes is inevitable. The huge variability due to emerging classes in future data cannot be modeled using traditional learning algorithms. As an alternative to parametric models there are *Nonparametric models* in Bayesian learning, which allow defining more flexible models capable to capture the variability in the data by using nonparametric distributions for the priors. Nonparametric models are not free of parameters; on the contrary, they have (countably) infinitely many parameters. In other words, in nonparametric models the number of parameters grow as the data demand. Dirichlet Processes (DP) [28] are among the most popular nonparametric distributions. DP is a distribution over distributions, which makes it suitable for using it as a prior over the distribution defined for parameters. A DP itself is defined by two parameters, a base distribution and a precision parameter. We will be using DPs in our nonexhaustive learning algorithms that we present in this thesis and will provide a more detailed description in Chapter 3.

Our research focuses primarily on supervised and semi-supervised learning in partially observed settings where the set of classes in the training data set is nonexhaustively defined and prospective data may originate from observed as well as unobserved

classes. We define nonparametric priors over class distributions and couple this with parametric data models to obtain semi-parametric models. We implement Markov Chain Monte Carlo (MCMC) inference techniques for predicting the class label of future samples in the presence of labeled and unlabeled data. Investigating this learning problem in the offline setting relates the proposed research to semi-supervised learning. However, unlike traditional semi-supervised learning problems where all classes are observed and labeled and unlabeled samples originate only from the observed classes, the proposed research addresses a more realistic scenario in which unlabeled samples can also originate from unobserved classes. We demonstrate the utility of the proposed semi-parametric models in handling unlabeled data to improve learning even when there is a clear mismatch between the models that generated the labeled and unlabeled data sets. In short, the main emphasis of the proposed research is on class discovery, modeling and also as a continuing effort class association in a higher level hierarchy.

2 FEASIBILITY STUDY FOR NONEXHAUSTIVE LEARNING WITH A PARAMETRIC MODEL

2.1 Bayesian Learning for Novel Class Detection in Multi-Class Settings

In this preliminary approach to nonexhaustive learning we assume that there is a common pattern among all class distributions (both known and unknown). Then as long as there is a sufficiently large number of known classes we can capture this pattern by means of Bayesian parameter estimation. In this approach, first we define a common prior over class parameters, θ, for all classes (known and unknown) with hyperparameters β, and then obtain the posterior estimate for θ in terms of a weighted mixture of β and the sample estimate, $\hat{\theta}$. The existing known classes play an important role for a reliable estimation of the sample estimates. When a new class is generated we estimate its parameters using the posterior mean for θ given $\hat{\theta}$s for current set of classes.

This initial work on nonexhaustive learning was motivated by the pathogen detection problem where we developed a real-time detection and classification algorithm that works in a multi-class setting, incorporates supervised classification with novelty detection and evaluates samples sequentially. Our approach evaluates each sample for the class conditional likelihood for all classes and compares the maximum likelihood value against a threshold to determine a novelty. If a sample x_i is identified as novelty we generate a new class and merge it into the current set of known classes. Otherwise, we assign the sample to class ω_j that maximizes the likelihood. If that class is a previously discovered class then we update the sample estimate, $\hat{\theta}_j$. If it is one of the initial training classes we do not perform any updates. The reason for this is that the initial classes might be acquired and validated by thorough procedures involving

manual processing, so we need to avoid updating class parameters with potentially incorrectly classified samples.

A newly generated class contains one sample initially, considered as a seed point for defining the new class, where the sample estimate may be ill-conditioned or even undefined. In this case the posterior mean for theta provides a reliable one as long as we have reliable estimates for the hyperparameter, β. The major assumption based on domain knowledge is that the parameter sets defining the class-conditional likelihoods share a common prior distribution and the labeled classes for training is large enough to obtain a robust estimate. As the sequential classification procedure iterates, some discovered classes may gradually reach to a certain size which may signal emergence of new types (pathogens). In such cases (biological) follow-up analyses are needed to identify the characteristics of such classes and assign labels to them if possible. Until such analysis occurs all newly discovered classes are considered as *unlabeled* classes and class parameters keep updated with each sample assigned to them. However, the parameters for the initial *labeled* classes in the training set are estimated only once at the beginning and they are considered as labeled throughout.

2.2 BayesNoDe: Bayesian Novelty Detection Algorithm

Density-based approaches use class-conditional likelihoods of samples to detect novelties. In short, if the maximum of the class-conditional likelihoods is above a designated threshold, then the sample belongs to one of the classes in the training library and is assigned the corresponding class label; otherwise the sample is identified as belonging to an unrepresented class, hence a novelty.

More formally, let Ω, Δ, and Γ denote the set of *all*, *known* and *unknown* bacteria classes, respectively, with $\Omega = \Delta \cup \Gamma$; A, K, and M are their corresponding cardinalities with $A = K + M$. The decision that minimizes the Bayes risk under the

0/1 loss-function assumption assigns a new sample x^* to the class with the highest posterior probability. More specifically,

$$x^* \in \omega_i^* \text{ s.t. } p(\theta_i|x^*) = \max_i \{p(\theta_i|x^*)\} \tag{2.1}$$

where $i = \{1, \ldots, A\}$. Here ω_i represents the i^{th} class and θ_i the parameters of its distribution. The classifier obtained by evaluating this decision rule is known as a maximum a posteriori classifier (MAP) [29].

Using Bayes' rule the above decision rule can be rewritten as follows:

$$x^* \in \omega_i^* \text{ s.t. } p(\theta_i|x^*) = \max_i \left\{ \frac{p(x^*|\theta_i)p(\theta_i)}{p(x^*)} \right\} \tag{2.2}$$

Ignoring the *evidence*, $p(x^*)$, and assuming all classes *a priori* likely, we evaluate only the class conditional likelihoods, which leaves us with the maximum likelihood (ML) decision function for classifying x^*:

$$x^* \in \omega_i^* \text{ s.t. } p(x^*|\theta_i) = \max_i \{p(x^*|\theta_i)\} \tag{2.3}$$

where x^* is considered a novelty if $\omega_i^* \in \Gamma$, and a sample of a known class if $\omega_i \in \Delta$.

Since the set of classes is nonexhaustive $p(x^*|\theta_i)$ cannot be computed for all classes and the decision function in (2.3) cannot be evaluated explicitly. We express (2.3) in terms of ω_i^* and rewrite it by separating $p(x^*|\theta_i)$ of *known* and *unknown* classes as:

$$h(x^*) = \begin{cases} x^* \textit{ is known} & \text{if } \psi \geq \gamma \\ x^* \textit{ is novelty} & \text{if } \psi < \gamma \end{cases} \tag{2.4}$$

where $\psi = \max_{\{i:\omega_i \in \Delta\}} \{p(x^*|\theta_i)\}$ and $\gamma = \max_{\{i:\omega_i \in \Gamma\}} \{p(z|\theta_i)\}$. This simply means that if the conditional likelihood of a known class for a sample x^* is less than γ, then x^* is considered a sample from an unrecognized class; otherwise x^* is a sample from a known class and thus can be assigned a known class label.

Since no data are available for unknown classes, γ cannot be explicitly estimated. One way to treat γ is to consider it as a tuning parameter to optimize sensitivity at a desired specificity or vice versa. In other words, γ can play a role to adjust for the compromise between sensitivity and specificity of the system.

2.2.1 Gaussianity Assumption and Covariance Estimation

A common and effective way to treat data of unknown nature is to assume Gaussian distributions for all classes, $\omega_i \sim N(\mu_i, \Sigma_i)$, $\theta_i = \{\mu_i, \Sigma_i\}$. With this assumption in place, (2.4) becomes:

$$h(x^*) = \begin{cases} x^* \textit{ is known} & \text{if } \min_{\{i:\omega_i \in \Delta\}} g_i(x^*) \leq \gamma \\ x^* \textit{ is novelty} & \text{if } \min_{\{i:\omega_i \in \Delta\}} g_i(x^*) > \gamma \end{cases} \tag{2.5}$$

where $g_i(x^*) = \log(|\Sigma_i|) + (x^* - \mu_i)^T \Sigma_i^{-1} (x^* - \mu_i)$ is the negative log-likelihood of class ω_i given x^* and $|\Sigma_i|$ is the determinant of Σ_i. For $\{i : \omega_i \in \Delta\}$, μ_i and Σ_i can be estimated from class-conditional data available in the training set.

When dealing with datasets containing limited numbers of training samples and high dimensionality, the covariance estimator plays an important role in the modeling of the class conditional distributions. The covariance estimate, $\hat{\Sigma}_i$, can be obtained by the sample covariance: $S_i = \frac{1}{n_i - 1} \left(X_i - \hat{\mu}_i e_{n_i}^T\right) \left(X_i - \hat{\mu}_i e_{n_i}^T\right)^T$, where n_i is the number of samples in class ω_i, e_{n_i} is a vector of ones of size n_i and $\hat{\mu}_i$ are estimated by the sample mean vectors: $\bar{x}_i = \frac{1}{n_i} X_i e_{n_i}$. Here for notational simplicity all samples belonging to class ω_i are denoted in the matrix form as $X_i = [x_{i1} \ldots x_{in_i}]$.

When the number of samples available for a given class is less than $d + 1$, d being the dimensionality, the sample covariance becomes ill conditioned, i.e. the inverse does not exist. In practice, a robust sample covariance requires many more samples than $d + 1$ because the number of parameters to estimate in a covariance matrix increases as the square of the dimensionality. This phenomenon is known as *the curse of dimensionality* [30].

Covariance Estimation

Although the research in covariance estimators using a limited number of samples with high dimensionality has a long history with relatively well-established techniques, two main approaches dominate the field. These are, regularized discriminant analysis

(RDA) [31] and empirical Bayes estimators [32]. RDA considers the mixture of sample and pooled covariance and an identity matrix as an estimator, with their weights empirically estimated by cross-validation. On the other hand, the Bayesian approach defines a pair of conjugate prior distributions over the sample and true covariance matrices, and uses the mean of the resulting posterior distribution as an estimator. In RDA, multiple samples from each class are required to estimate the mixing weights by cross-validation, and thus to estimate the covariance matrix, whereas in the Bayesian approach, the covariance estimator is a function of the parameters of the prior distribution, which are estimated using samples of the known classes.

Creating a new class for each detected novelty and defining the class by its mean and covariance matrix form the core component of the described approach. The Bayesian approach assumes a common prior for all classes (known and unknown) and estimates the covariance matrix using the posterior mean. In that regard, the use of the Bayesian approach makes intuitive sense in the nonexhaustive setting, mainly because we assume that there is a common pattern among the class distributions of all classes and that it can be captured with known classes only, provided that a sufficiently large number of them are available for training. In the bacterial detection problem, for instance, although our training dataset represents only a small portion of a potentially very large number of bacterial serovars, unlike traditional machine learning problems, the number of available classes is still large enough to allow for a reasonably robust estimation of the prior distribution. This facilitates the estimation of the covariance matrices for the new classes, which is especially important when defining a class for the first time using the sample detected as novelty. In the following section we describe the special family of conjugate priors for covariance estimation under the Bayesian framework.

2.2.2 Family of Wishart and Inverted-Wishart Conjugate Priors

The assumption of Gaussian samples, i.e., $\omega_i \sim N(\mu_i, \Sigma_i)$, implies that the sample covariance matrices S_i, $i = \{1, ..., K\}$, where K is the number of known classes, are mutually independent with $f_i S_i \sim W(\Sigma_i, f_i)$. Here $f_i = n_i - 1$ and $W(\Sigma_i, f_i)$ denotes the Wishart distribution with f_i degrees of freedom and a parameter matrix Σ_i. The inverted-Wishart distribution is conjugate to the Wishart distribution and thus provides a convenient prior for Σ_i.

We assume that Σ_i is distributed according to an inverted-Wishart distribution with m degrees of freedom as: $\Sigma_i \sim W^{-1}((m-d-1)\Psi, m),\ m > d+1$. The scaling constant $(m-d-1)$ before Ψ is chosen to satisfy $E\{\Sigma_i\} = \Psi$. Under this setting, the posterior distribution of Σ_i given $\{S_1, ..., S_K\}$, is obtained as described in [33]: $\Sigma_i|(S_1, ...S_K) \sim W^{-1}(f_i S_i + (m-d-1)\Psi, f_i + m)$. The mean of this posterior distribution is [33]:

$$\hat{\Sigma}_i(\Psi, m) = \frac{f_i}{f_i + m + d - 1} S_i + \frac{m - d - 1}{f_i + m + d - 1} \Psi \tag{2.6}$$

Under squared-error loss, the posterior mean is the Bayes estimator of Σ_i. The estimator is a weighted average of S_i and Ψ, and it shifts toward S_i for large f_i and approaches Ψ for large m. For a class with just one sample, the estimator yields Ψ, which implies that no matter what the dimensionality is a nonsingular covariance estimate can be obtained using this estimator, provided that Ψ is nonsingular. The estimator is a function of Ψ and m, which are the parameters of the inverted-Wishart prior for Σ_i, and their closed-form estimates do not exist. The study in [32] suggests estimating Ψ with the unbiased and consistent estimate S_p, i.e., the pooled covariance, and maximizing the marginal likelihood of S_i for $m > d+1$ numerically to estimate m. In this study we set Ψ to S_p but estimate m to maximize the classification accuracy for the known classes by cross-validating over the training samples. Here, S_p is the pooled covariance matrix defined by $S_p = \frac{f_1 S_1 + f_2 S_2 + \cdots + f_K S_K}{N - K}$ where N is the total number of samples available in the training dataset.

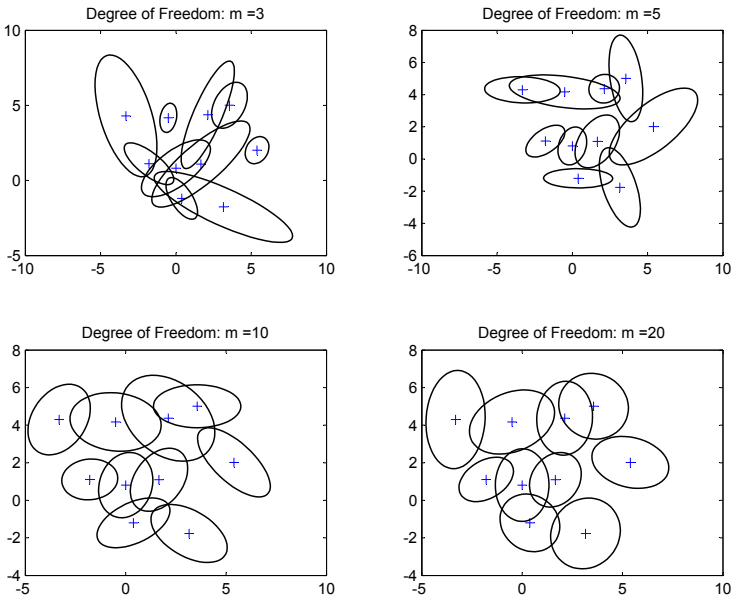


Figure 2.1. Simulated classes illustrating the impact of the degree of freedom, m, in the inverted-Wishart distribution.

Figure 2 illustrates the effect of m on the modeling of the classes. In this example 10 classes are generated. Their mean vectors are chosen randomly from a normal distribution with mean at the origin and covariance matrix equal to $10I$, where I denotes the 2-D identity matrix. The covariance matrices of the classes are obtained from an inverted-Wishart distribution with the first parameter $\Psi = 0.3I$, which is designed to yield relatively circular distributions. The parameter m, the degree of freedom, takes the values 3, 5, 10 and 20, respectively, in the four cases shown in figure 2. As m increases, initially the classes transform from more elongated distributions to more circular ones but only slight changes in shape and orientation are observed beyond a certain m value.

So far, we have discussed a framework for detecting novelties in real time based on maximum likelihood (ML) evaluation of samples using known classes. The approach employs a pair of conjugate Wishart priors to estimate the covariance matrices of known classes and detects novelties by thresholding the maximum likelihood evaluated with known classes. We refer to this approach as *ML-Wishart* in the experiments section. Next part presents the online class discovery component that is combined with ML-Wishart. The resulting approach is referred to as *BayesNoDe* in the rest of this chapter.

2.2.3 Real-Time Discovery of New Classes

As formulated in (2.5), for a new sample $x^* \in \Re^d$, if $\min_{\{i:\omega_i \in \Delta\}} g_i(x^*) > \gamma$ we consider it as a novelty. In other words, if the negative log-likelihoods of known classes given x^* are all greater than the designated threshold γ, then the sample is considered a novelty.

When a new sample is detected as a novelty, a new class is generated and defined by the parameters, (μ, Σ), both of which are not known. With just one sample, since S is not defined and $f = 0$, the posterior mean in (2.6) is equivalent to Ψ and thus

the Bayesian estimator for Σ becomes $\hat{\Sigma} = \Psi$. The mean vector, μ is estimated by $\hat{\mu} = x^*$, i.e. the sample itself.

The set of known classes is augmented with this new class. So for the next sample available, the decision function in (2.5) is evaluated for classes known initially as well as for the newly created classes. If the sample is detected as a novelty, the above procedure is repeated to generate another class. Otherwise, if the sample is classified into one of the existing classes, then we check for the class that minimizes the negative log-likelihood. If the sample is assigned to a previously discovered class, then we update the class parameters μ by the new $\bar{x}$ and Σ using equation (2.6). Since, there is more than one sample available now, $\hat{\Sigma}$ becomes a mixture of the sample covariance and Ψ. If, on the other hand, the sample is assigned to a class known initially, then no class update is necessary.

As an aside, in case the Gaussianity assumption does not perfectly fit for either the set of initially known classes or the newly discovered ones it can be addressed through defining mixture models. The theory of finite mixture models [26] states that given enough components and under fairly weak assumptions, a mixture model can approximate a given density arbitrarily closely, allowing great flexibility. In other words, even if the initially known classes are not Gaussian, the class-conditional distributions can still be estimated arbitrarily closely, using a mixture of Gaussians. A mixture of Gaussian subclasses can be learned for each class data through a process involving expectation maximization [27] and model selection. Once the Gaussian subcomponents are identified for each class data, the described approach can be implemented at the subclass level by considering each subclass as an independent Gaussian class. Similarly, when discovering new classes, only clusters with Gaussian patterns will be created for novelties. However, true classes with informational value can still be recovered by grouping newly discovered clusters under a higher-level class using domain/expert knowledge.

2.2.4 An Illustrative Example

Next, we demonstrate the algorithm detecting novelties and creating classes on a simple 2-D dataset. Similar to the previous example we generate ten classes with their covariance matrices obtained from an inverted-Wishart distribution with parameters $\Psi = 0.3I$ and $m = 10$ and their mean vectors are chosen randomly from a normal distribution with mean at the origin and covariance matrix equal to $10I$. Here, I denotes the 2-D identity matrix.

Figure 3a shows all ten classes. Known classes are depicted by solid lines, unknown classes by dashed lines. The square sign locates the mean of each class. The ellipses represent the class boundaries as defined by the three standard deviation distance from the class means. A total of 80 samples are generated from the ten classes: 5 from each of the known classes and 20 from each of the unknown classes. Test samples are classified sequentially using the BayesNoDe algorithm. Figures 3b, 3c, and 3d illustrate cases where 16/80, 48/80 and 80/80 samples are classified, respectively. Red solid lines indicate the estimated distribution contours for newly discovered classes in each subfigure with the diamond signs locating their estimated means. The blue $+$ signs and red $\times$ signs in each subfigure show the samples classified to known and unknown classes, respectively. Figure 3e demonstrates novelty detection using ML-Wishart, i.e., with a fixed set of classes in the training dataset, and figure 3f illustrates the case where no novelty detection is performed at all. In these two figures the samples marked with red circles indicate samples from the unknown classes misclassified as known. Also in figure 3e blue solid lines correspond to $g(z) = \gamma$ as defined in (2.5) and indicate the classification boundaries for the unknown samples.

As figures 3b, 3c, and 3d demonstrate, the algorithm gradually recovers the unknown classes as more test samples are introduced, converging to almost ideal distributions after all 80 test samples are classified.

Comparing figures 3d and 3e shows the improvement achieved by the BayesNoDe algorithm over the ML-Wishart as a result of the dynamically updated training

dataset. When no novelty detection is used, all samples are misclassified as illustrated in figure 3f.

2.3 Experiments

2.3.1 Bacteria Detection

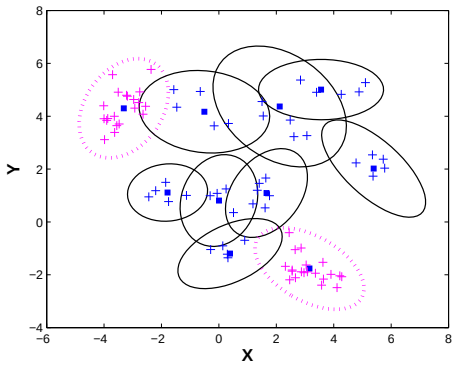
A total of 28 strains (subclasses) from five different bacteria species were considered in this study. The species available are *Escherichia coli, Listeria, Salmonella, Staphylococcus and Vibrio*. Table A.1 in the Appendix shows the list of 28 strains from 5 species considered in this study together with the number of samples collected for each one using an optical scattering system described in Section 1.2. In our experiments we treated each strain as a separate class and used the number of samples listed in Table A.1 from each class for training.

Feature Selection

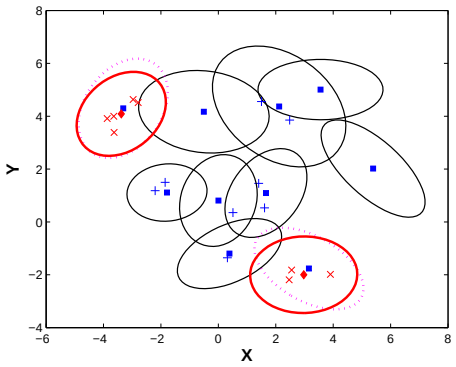
Scatter patterns of the bacteria are characterized by a total of 50 features involving moment invariants and Haralick texture descriptors. Details of the feature extraction process are available in [34].

Classifier Design

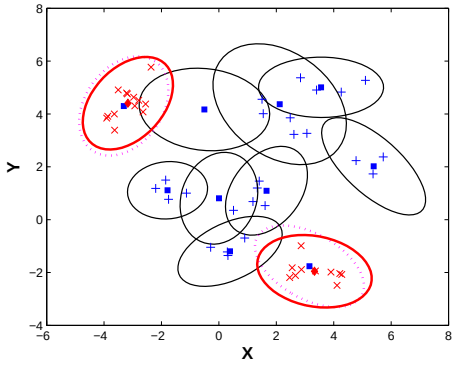
The classification methods considered are the support vector domain description (SVDD) [8], which is the benchmark technique for detecting anomalies and novelties, maximum likelihood (ML) using common covariance (ML-Common), ML using common covariance with simulated subclass generation (MLS) [35], ML with the covariance matrix estimated by the posterior mean of the inverted-Wishart distribution (ML-Wishart), and the BayesNoDe algorithm. The maximum likelihood classifier using sample covariance is not considered here, because sample covariances were ill conditioned for most classes.



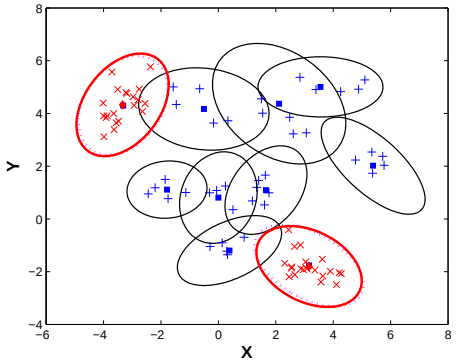
(a) Unknown classes w/ dashed line



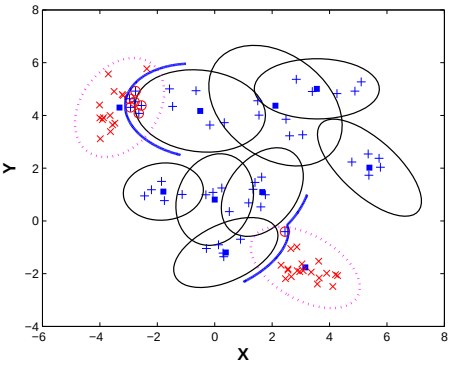
(b) 16/80 samples classified



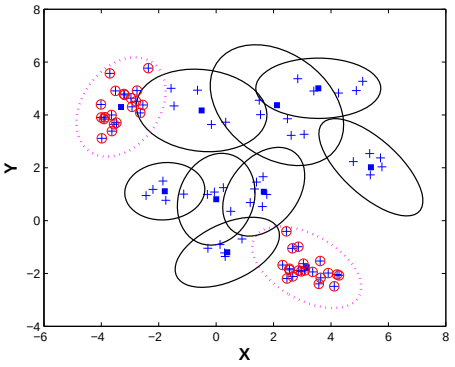
(c) 48/80 samples classified



(d) 80/80 classified - BayesNoDe



(e) 80/80 classified - ML-Wishart



(f) 80/80 classified - No novelty detection

Figure 2.2. Illustration of the algorithm with an artificial dataset. (a) Pink dashed lines indicate unknown classes with 20 samples each. Black solid lines indicate known classes with 5 samples each. (b)-(d) Red solid lines indicate newly discovered classes. Blue squares mark mean vectors for original classes. Red diamonds mark mean vectors for newly discovered classes. Blue + signs, indicate samples from known classes, red × signs indicate samples from unknown classes. (e) Blue solid lines indicate the classification boundaries for samples from unknown classes. (e),(f) Encircled + signs indicate undetected samples from unknown classes.

As explained in Sections 2.2 and 2.2.1, the general idea of ML classifiers is based on the ML decision function in (2.3) and works according to the formulation in (2.4). ML-Wishart and ML-Common are the special cases of ML. They differ in estimating the covariance matrices of the training classes. Corresponding mean vectors, μ_i, are all calculated by the sample mean. More specifically, ML-Common implements (2.5), where $\Sigma_i = \Sigma$ for all i, and Σ represents the common covariance matrix estimated by the average of the sample covariances. As described in [35] MLS extends ML-Common by simulating the space of all classes. This approach assumes a Gaussian prior for the mean vectors, and its parameters are estimated using the estimates of the mean vectors for each class. Lastly, for the proposed ML-Wishart and BayesNoDe, the covariance matrices are estimated for each class using the posterior mean defined in (2.6). The parameters m and Ψ are estimated as described in Section 2.2.2.

As for the SVDD algorithm, optimization involves two sets of parameters. These are C, the cost of leaving a training sample outside the support, and σ, the width of the Gaussian radial basis function (RBF) kernel. These parameters are estimated by 10-fold cross-validation at the class level. When optimizing parameters for a given class, the training samples of the given class are considered positive and the samples of remaining classes are considered negative. At each fold of the cross-validation algorithm, SVDD is trained using positive samples only but tested on both positive and negative samples. The parameter set (C_*, σ_*) that optimizes the area under the receiver operating characteristic (ROC) curve is chosen as the optimum set for the given class. This process is repeated for all classes.

Classifier Validation and Evaluation

Since the training dataset is nonexhaustive, the goal is to design a classifier that accurately detects samples of known classes as known and those of unknown classes as novelty. In this framework, classifiers can be more properly evaluated using receiver operating characteristic (ROC) curves. Here sensitivity is defined as the number

of samples from known classes classified as known divided by the total number of samples from known classes. Specificity is defined as the number of samples from unknown classes detected as novelty, divided by the total number of samples from unknown classes. Multiple sensitivity and specificity values are obtained for each classifier to plot the ROC curves. For the ML-based approaches, different operating points are obtained by varying the threshold γ in (2.5). For SVDD, the distances from the center of each class is normalized by the radius of the corresponding sphere. For a new sample, the minimum of the normalized class distances is computed and thresholded to obtain different operating points.

To evaluate the classifiers the 2054 samples are randomly split into two sets, as train and test, with 80% of the samples going into the training set and the remaining 20% into the test. Stratified sampling is used to make sure that each subclass is represented in both sets. This process is repeated ten times to obtain ten different pairs of train-test sets. Then, one subclass from each of the five bacteria species is randomly selected, so a total of five subclasses out of the twenty-eight available are identified. All samples of these five classes are removed from the training datasets making these classes unknown. The classifiers are trained with the resulting nonexhaustive training sets and tested on the corresponding test sets. For each data split, the area under the ROC curve, i.e., Az value is computed. The Az values averaged over the ten different train-test splits are recorded along with the standard deviation.

Results and Analysis

In order to account for the possible bias introduced by the set of removed classes the above process is repeated 20 times each time removing a randomly selected set of five classes from the training set. Each such repetition involves running the same experiment with a different nonexhaustive subset of the original data. Az values achieved for each classifier are included in Table II for all 20 experiments. As described earlier these values are the average of the ten runs each executed with a different

train-test split and the values in parentheses indicate standard deviations. The mean Az values across all 20 runs are listed in Table III. These results clearly favor the proposed *BayesNoDe* algorithm, which generated the best AUC in all 20 repetitions. Standard deviations indicate that the differences are statistically significant in most of the 20 experiments. The BayesNoDe algorithm is an extension of the ML-Wishart algorithm, both of which are proposed in this study. ML-Wishart ranked second, but the results indicate that creating new classes and augmenting the set of known classes with these new classes makes a considerable impact on the prediction accuracy of the classifier and gives the BayesNoDe algorithm a significant advantage over the ML-Wishart. SVDD ranked third along with ML-Common and MLS.

Next, we picked four sample cases out of the 20 using the overall Az values achieved by the classifiers as the selection criteria. Largest Az value among all 20 repetitions is recorded in repetition 10 (Figure 4a). Repetitions 13 and 16 represent two average cases (Figures 4b and 4c). Repetition 20 is included to show results for a relatively poor case (Figure 4d). The ROC curves corresponding to the proposed BayesNoDe algorithm dominate the other curves in all cases. We also analyzed the classification accuracy of the known samples and observed that the known samples are assigned to classes with over 95% accuracy across all operating points for all four cases considered here. These results indicate that the proposed approach not only performs well in identifying samples of the unknown classes as novelties but yields promising results in classifying samples of the known classes as well.

2.3.2 Letter Recognition

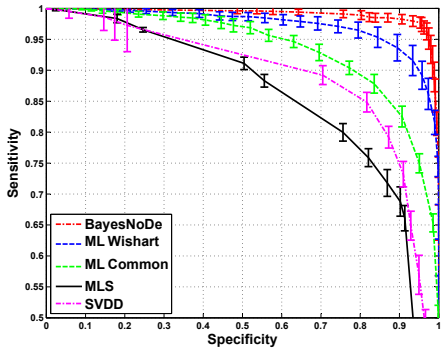
To show that improvements achieved by the proposed BayesNoDe algorithm is not specific to the Bacterial detection application that motivated this research, we used the benchmark letter recognition dataset from the UCI repository [36] for further validation of the proposed approach for novelty detection. This dataset is mainly selected for containing a large number of classes. The dataset contains 20,000 samples

Table 2.1
AUC (Area Under the Curve) values averaged over 10 iterations for all 20 experiments run with the bacteria dataset. A set of five subclasses is randomly selected and considered unknown during each of the 20 experiments. BayesNoDe results in the best AUC values for all 20 experiments. Values in parenthesis indicate standard deviations.

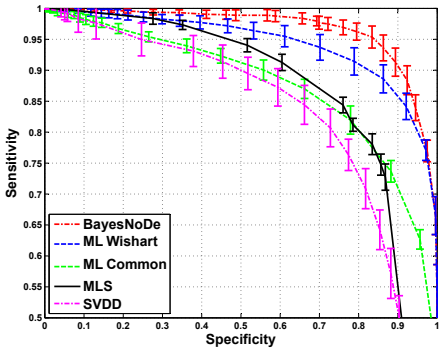
Rep. #	1	2	3	4	5	6	7	8	9	10
BNode	0.97 (0.01)	0.92 (0.01)	0.98 (0.01)	0.92 (0.01)	0.93 (0.01)	0.95 (0.01)	0.98 (0.01)	0.96 (0.01)	0.95 (0.01)	0.99 (0.01)
ML-C	0.88 (0.01)	0.71 (0.01)	0.90 (0.01)	0.82 (0.01)	0.79 (0.01)	0.83 (0.01)	0.83 (0.01)	0.87 (0.01)	0.89 (0.01)	0.94 (0.00)
ML-W	0.95 (0.01)	0.88 (0.01)	0.96 (0.00)	0.90 (0.01)	0.89 (0.01)	0.92 (0.01)	0.95 (0.01)	0.94 (0.01)	0.94 (0.01)	0.98 (0.01)
MLS	0.87 (0.01)	0.80 (0.01)	0.82 (0.01)	0.81 (0.01)	0.80 (0.01)	0.84 (0.01)	0.92 (0.01)	0.86 (0.01)	0.78 (0.01)	0.85 (0.01)
SVDD	0.87 (0.01)	0.77 (0.02)	0.90 (0.02)	0.81 (0.03)	0.76 (0.03)	0.81 (0.02)	0.86 (0.02)	0.84 (0.02)	0.86 (0.02)	0.89 (0.01)
Rep. #	11	12	13	14	15	16	17	18	19	20
BNode	0.91 (0.01)	0.98 (0.01)	0.97 (0.01)	0.93 (0.01)	0.89 (0.01)	0.95 (0.01)	0.95 (0.01)	0.82 (0.01)	0.92 (0.01)	0.88 (0.01)
ML-C	0.80 (0.01)	0.90 (0.01)	0.88 (0.01)	0.76 (0.01)	0.78 (0.03)	0.83 (0.01)	0.81 (0.01)	0.72 (0.01)	0.77 (0.01)	0.81 (0.02)
ML-W	0.87 (0.01)	0.96 (0.01)	0.94 (0.01)	0.88 (0.01)	0.85 (0.01)	0.92 (0.01)	0.91 (0.01)	0.79 (0.01)	0.87 (0.01)	0.85 (0.01)
MLS	0.78 (0.01)	0.82 (0.01)	0.87 (0.01)	0.81 (0.01)	0.86 (0.01)	0.85 (0.01)	0.84 (0.01)	0.80 (0.01)	0.74 (0.01)	0.84 (0.01)
SVDD	0.83 (0.01)	0.90 (0.01)	0.82 (0.06)	0.76 (0.03)	0.77 (0.03)	0.83 (0.01)	0.81 (0.01)	0.73 (0.03)	0.81 (0.02)	0.80 (0.02)

Table 2.2
Average AUCs over 20 repetitions.

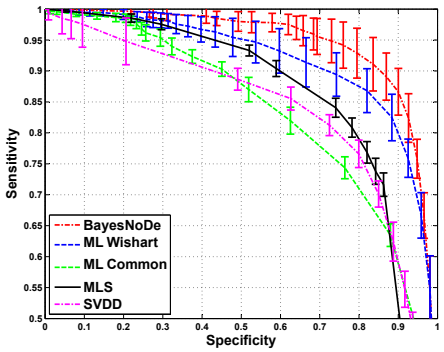
Methods	**Avg. AUC**
BayesNoDe	0.94 (0.05)
ML-Wishart	0.91 (0.06)
ML-Common	0.83 (0.04)
MLS	0.83 (0.04)
SVDD	0.82 (0.05)



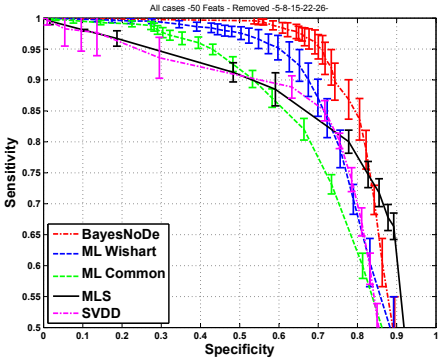
(a) Case 10. Removed classes: 6,12,15,18,27



(b) Case 13. Removed classes: 1,11,16,22,23



(c) Case 16. Removed classes: 2,8,16,21,28



(d) Case 20. Removed classes: 5,8,15,22,26

Figure 2.3. ROC curves for selected repetitions 10, 13, 16 and 20.

for 26 classes, one for each letter of the alphabet. Each sample is characterized using 16 features.

This dataset is different than the bacteria detection dataset in that, it is not susceptible to the curse of dimensionality as much. There is an average of 770 samples for each class as opposed to an average of 80 samples for each bacteria subclasses. The dimensionality of the data (d=16) is also much lower than the 50 features used in the bacteria detection dataset.

We followed an experiment design similar to the bacteria detection experiment. The 20,000 samples are randomly split into train and test sets, with 80% of the samples going into the training set and the remaining 20% in the test. Stratified sampling is used to make sure each class is represented in both the training and the test sets. This process is repeated five times to obtain five different pairs of train-test sets. Then, five classes are randomly selected and their samples are removed from the training datasets. The classifiers are trained with the resulting non-exhaustive training sets, and tested on the corresponding test sets. For each case, Az value is computed. The Az values averaged over the five different train-test splits are recorded along with the standard deviation.

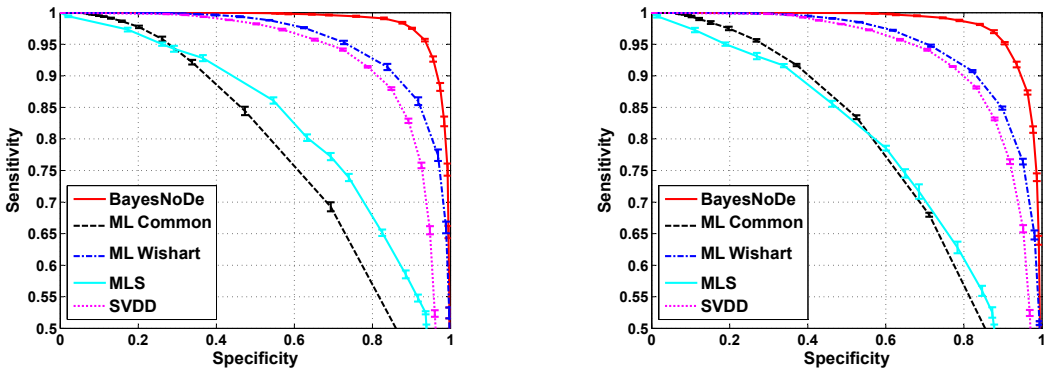
Classifier Design

The same set of classifiers considered in Experiment 2.3.1 are also considered here. SVDD and MLS are trained the same way as in Experiment 2.3.1. For the ML based classifiers, since classes contain a relatively larger number of samples, a single Gaussian might not fit class data well. In this case, as discussed in Section 2.2.3, the actual class distributions can be modeled more effectively using a mixture of Gaussians. We fit up to five components for each class distribution using standard expectation-maximization algorithm [27] with the optimum number of components selected using the Bayesian Information Criterion [37]. Once mixture models are obtained, each subclass is considered as an independent class and all maximum-

likelihood based classifiers are run with the new set of known classes. On the average for each class data mixture fitting returned three subclasses.

Results and Analysis

The experiment is repeated twice each time removing a randomly selected set of five classes from the training set. The ROC curves are plotted in Figures 5a and 5b. For this experiment SVDD seems to model the data well and becomes the sole competitor to BayesNoDe and ML-Wishart. ML-Wishart performs slightly better than SVDD. The detection accuracy of BayesNoDe is almost perfect and as the error bars indicate the improvements achieved over other methods are statistically significant.



(a) Removed classes: 7,9,12,14,24. (b) Removed classes: 2,9,11,12,22.

Figure 2.4. ROC curves for two different set of removed classes.

2.4 Discussion

This preliminary approach serves as a proof-of-concept, where discovery and modeling of additional classes yields a more comprehensive model for the data and in turn improves the overall classification accuracy for known and unknown classes. However, it is limited in several ways. First and foremost it is based on the class conditional

likelihood for creating new classes which does not allow for the incorporation of the prior belief about the odds of encountering new classes in the tested sequence of data. Second, it makes a rather restrictive assumption about the underlying data model, which may not hold in practice. Third, formation of new classes gradually as samples arrive relies on the condition that the classes should be well separated. In other words, without an overall update in an iterative manner it is difficult for the classes to converge to the true distributions in many situations. Fourth, the γ threshold is crucial for the number of components introduced to model new classes, yet fixing a value based on sole domain knowledge and/or training data makes it inflexible to adjust for variability that might occur in future data. Fifth, a systematic approach to class association in this framework is not possible because the prior model based on the covariance matrix alone does not allow for modeling class hierarchy.

To incorporate a prior model and make the decision to create a new class in a data driven manner, to replace the fixed γ value with a parameter that can be estimated dynamically by means of all available data, and to gain more flexibility in assuming a prior data model as well we will design nonparametric Bayesian approaches in the following chapters for different learning problems in partially-observed settings.

3 BAYESIAN NONPARAMETRIC MODELS FOR PARTIALLY-OBSERVED SETTINGS

Parametric models have a fixed set of finite number of parameters, Θ, regardless of the size of the data set. Given Θ, the predictions are independent of the data D, $p(x, \Theta|D) = p(x|\Theta)p(\Theta|D)$. In other words, the parameters are a finite summary of the data, where estimating the parameters is also referred to as model-based learning (e.g. mixture of k Gaussians). For example, in the previous chapter we used a parametric model for the data where we assumed Gaussian distributions for all classes. A user defined threshold value, γ, played the key role in deciding whether a sample comes from a novel class or belongs to an existing class. Therefore, the robustness of the approach depends on an accurate reasoning or tuning for the γ value. However, a more desirable practice is to probabilistically represent the prior belief about existence of new classes and to dynamically estimate such essential values.

Nonparametric models, on the other hand, are a means for getting much more flexible models that can automatically infer an adequate model size/complexity from the data, without needing to explicitly do Bayesian model comparison. Nonparametric models allow the number of parameters to grow with the data set size, and one way to derive them is to start with a finite parametric model and take the limit as number of parameters go to infinity. For practical purposes, though, we can think of the predictions to mainly depend on the data and possibly on a small number of parameters (e.g. α, $p(x|D, \alpha)$).

Depending on the machine learning problem a wide variety of nonparametric models are available including Gaussian Processes (GP), Dirichlet Processes (DP), Hierarchical Dirichlet Processes (HDP), Infinite Hidden Markov Models, Indian Buffet Processes (IBP), Polya Trees, Dirichlet Diffusion Trees, etc. For a detailed introduction to Bayesian Nonparametrics and existing models we refer the reader to [38]. In

this thesis we use DPs and HDPs in modeling the partially-observed data sets and the presented applications, and in this chapter we present how DP mixtures can be used for supervised learning in partially-observed settings.

3.1 Partially-Observed Dirichlet Process Mixture Models (PO-DPM)

The Dirichlet process mixture (DPM) model has been heavily studied in unsupervised learning for offline and online clustering applications over the past decade [18–21, 23, 24, 39]. Most of these approaches assume that all the components of the mixture model are unobserved and study inference techniques to learn these components without any label information. Although certain aspects of these studies have been inherently useful for our study, an unsupervised approach would be most desirable in settings where the patterns and structure within the data set are completely unobserved. However, when label information for some portion of the observed data exist it could be exploited to better estimate prior parameters of the model. Overall, we take a similar approach to [39] in the use of DPMs where their purpose is to do spike sorting—to cluster neuron firing signals to find out how many neurons might be involved in a measured brain activity experiment. Yet, we have a major conceptual difference due to availability of labeled samples, which in turn results in algorithmic differences in the modeling and the inference as well. In this section we describe a framework to initiate a supervised DPM model with the labeled samples and incorporate the unlabeled samples to classify them into existing classes or introduce new classes as the data demand. We begin with explaining the DP in some detail.

3.1.1 Dirichlet Process Prior (DPP)

Let x_i, $i = \{1, \ldots, n\}$ be the feature vector characterizing a sample in the d-dimensional vector space $\Re$ and y_i be its corresponding class indicator variable. If x_i is distributed according to an unknown distribution $p(.|\theta_i)$, then defining a DPP over

class distributions is equivalent to modeling the prior distribution of θ by a Dirichlet process. More formally,

$$\begin{aligned} x_i|\theta_i &\sim p(\cdot|\theta_i) \\ \theta_i &\sim G(\cdot) \\ G &\sim DP(\cdot|G_0, \alpha) \end{aligned} \tag{3.1}$$

where G is a random probability measure, which is distributed according to a Dirichlet process (DP) defined by a base distribution, G_0, and the precision parameter, α. Given that G is distributed according to a DP, the stick-breaking construction due to [40] suggests $G = \sum_{i=1}^{\infty} \beta_i \delta_{\phi_i}$ where $\beta_i = \beta_i' \prod_{l=1}^{i-1}(1 - \beta_l')$, $\beta_i' \sim Beta(1, \alpha)$, and $\phi_i \sim G_0$. The points ϕ_i are called the *atoms* of G. In short, the stick-breaking interpretation considers a unit-length stick that is broken according to a sample β_i' from a Beta distribution where β_i' indicates the portion of the remainder of the stick and β_i is the length of the piece of the stick assigned to the i^{th} value after the stick is broken $i - 1$ times (see fig. 3.1). The precision parameter, α, is the parameter that controls how much of the stick will be left for subsequent values. The smaller α is, the larger β_i' will be, and the less of the stick will be left for subsequent values on average. In other words, α is the parameter that controls the prior probability of assigning a new sample to a new component and thus, plays a critical role in the number of components generated. Note that unlike continuous distributions the probability of sampling the same ϕ_i twice is not zero and proportional to β_i. Thus, G is considered a discrete distribution.

Next we present a general framework for learning with a nonexhaustively defined training dataset using Dirichlet process priors, which allows for discovery and modeling of new classes. To differentiate newly discovered classes from those initially available in the training library we introduce the notion of *observed* vs. *unobserved* classes, which refer to verified and unverified classes, respectively.

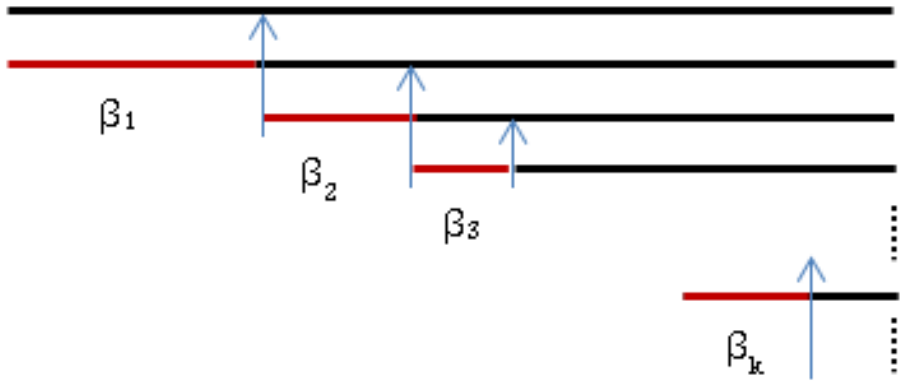


Figure 3.1. Generating the mixing proportions β_i using the stick-breaking procedure. Initially we have a stick of unit length at the top. The breaking points marked with vertical arrows are determined by the β_i' obtained from the beta distribution. The red lines correspond to the mixing proportions β_i. These pieces are removed for the next step and the breaking process is continued on the remaining piece shown with the black solid lines.

3.1.2 DPP in a Nonexhaustive Learning Framework (NEL-DPP)

The suitability of the DPP model for nonexhaustive learning can be better conceived with the help of the conditional prior of θ. Let's assume that at a certain time point the observed data contains a sequence of n samples. The conditional prior of θ_{n+1} conditioned on all past θ_i, $i = \{1, \ldots, n\}$ can be obtained by integrating out G in (3.1) which becomes:

$$\begin{aligned} \theta_1 &\sim G_0(\cdot) \\ \theta_2|\theta_1 &\sim \frac{\alpha}{\alpha+1} G_0(\cdot) + \frac{1}{\alpha+1}\delta_{\theta_1} \\ &\cdots \\ \theta_{n+1}|\theta_1, \ldots, \theta_n &\sim \frac{\alpha}{\alpha+n} G_0(\cdot) + \frac{1}{\alpha+n}\sum_{i=1}^{n}\delta_{\theta_i} \end{aligned} \tag{3.2}$$

This conditional prior can be interpreted as a mixture of two distributions. Any sample that originates from this prior comes from the base distribution $G_0(\cdot)$ with a probability of $\frac{\alpha}{\alpha+n}$ or uniformly generated from $\{\theta_1, \ldots, \theta_n\}$ with a probability of $\frac{n}{\alpha+n}$. In other words, the first sample in the sequence, θ_1, comes from $G_0(\cdot)$ with a probability of one, the second sample, θ_2, comes from $G_0(\cdot)$ with a probability of $\frac{\alpha}{\alpha+1}$ or is equivalent to θ_1 with a probability of $\frac{1}{\alpha+1}$, the third sample, θ_3, comes from $G_0(\cdot)$ with a probability of $\frac{\alpha}{\alpha+2}$ or is equivalent to one of θ_1 or θ_2 with a probability of $\frac{2}{\alpha+2}$ and so on. With a positive probability a sequence of n samples generated this way will not be all distinct. If we assume that there are $k \leq n$ distinct values of θ in a sequence of size n, then (3.2) can be rewritten as:

$$\theta_{n+1}|\theta_1^*, \ldots, \theta_k^* \sim \frac{\alpha}{\alpha+n} G_0(\cdot) + \frac{1}{\alpha+n}\sum_{j=1}^{k} n_j \delta_{\theta_j^*} \tag{3.3}$$

where θ_j^*, $j = \{1, \ldots, k\}$ are the distinct values of θ_i and n_j are the number of occurrences of each θ_j^* in the sequence. Each θ_j^* defines a unique class with an indicator variable y_j^*, whose samples are distributed according to the probability distribution $p(\cdot|\theta_j^*)$. Based on (3.3), after a sequence of n samples are generated, $y_{n+1} = y_j^*$ with probability equal to $\frac{n_j}{\alpha+n}$, and $y_{n+1} = y_{k+1}^*$, with probability equal to $\frac{\alpha}{\alpha+n}$, where y_{k+1}^* is the new class whose parameter is defined by θ_{k+1}^* and sampled from $G_0(\cdot)$.

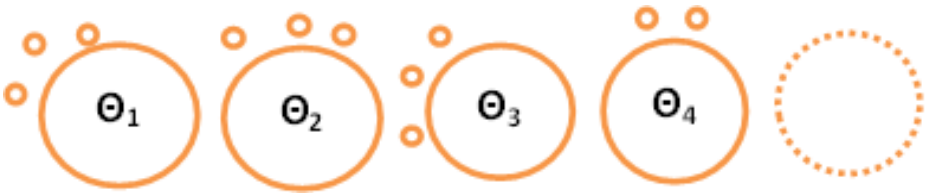


Figure 3.2. Illustration of the CRP model with tables and customers. Each circle corresponds to a table and thus to a unique cluster defined by θ_j. The black dots around the circles are the customers seated by the stochastic CRP prior model.

This prior model can also be illustrated as a *Chinese Restaurant process* (CRP) [41]. The CRP uses a metaphor of a Chinese restaurant with infinitely many tables where the $(n+1)^{th}$ customer sits at a previously occupied table j with a probability of $\frac{n_j}{\alpha+n}$ and at a new table $k+1$ with a probability of $\frac{\alpha}{\alpha+n}$. Here n_j is the number of customers sitting at table j and n is the total number of customers. Each table uniquely corresponds to one of the θ^*'s and therefore represents the grouping of samples in corresponding classes (fig. 3.2).

Our discussion so far has been limited to the prior model. Next, we will incorporate the data model and use the conditional posterior to determine whether, for instance, a new sample x_{n+1} should be assigned to one of the existing classes or to a new class sampled from G_0. More specifically, we are interested in the distribution $p(\theta_{n+1}|x_{n+1}, \theta_1^*, \ldots, \theta_k^*)$, which—using Bayes' rule—can be expressed as:

$$p(\theta_{n+1}|x_{n+1}, \theta_1^*, \ldots, \theta_k^*) \propto p(x_{n+1}|\theta_{n+1}) p(\theta_{n+1}|\theta_1^*, \ldots, \theta_k^*) \tag{3.4}$$

after substituting (3.3) into (3.4) and replacing G_0 with $P(\theta_{n+1})$ we obtain the following mixture distribution with two terms:

$$\begin{aligned} p(\theta_{n+1}|x_{n+1}, \theta_1, \ldots, \theta_n) &\propto \tfrac{\alpha}{\alpha+n} p(\theta_{n+1}) p(x_{n+1}|\theta_{n+1}) + \tfrac{1}{\alpha+n} \textstyle\sum_{j=1}^{k} n_j p(x_{n+1}|\theta_j^*) \delta_{\theta_j^*} \\ &= \tfrac{\alpha}{\alpha+n} p(x_{n+1}) p(\theta_{n+1}|x_{n+1}) + \tfrac{1}{\alpha+n} \textstyle\sum_{j=1}^{k} n_j p(x_{n+1}|\theta_j^*) \delta_{\theta_j^*} \end{aligned} \tag{3.5}$$

which indicates x_{n+1} either comes from a new class, y^*_{k+1}, which inherits θ^*_{k+1} sampled from $p(\theta_{n+1}|x_{n+1})$, with a probability proportional to $\frac{\alpha}{\alpha+n}p(x_{n+1})$ or belongs to the class indicated by y^*_j with a probability proportional to $\frac{n_j}{\alpha+n}p(x_{n+1}|\theta^*_j)$.

The probability that a given sample comes from a new class is a function of the number of samples, n, and the precision parameter, α. The parameter α incorporates our prior belief about the odds of encountering a new class. One viable approach to predicting α when training samples are obtained as a batch is to sample it from the distribution $p(\alpha|\tilde{k}, n)$ [42]. This approach is widely used in mixture density estimation involving batch data as part of a Gibbs sampler. Moreover, the base distribution G_0 is tightly coupled with the data model.

Since θ^*_j are not known and has to be estimated using samples in the represented classes, $p(x_{n+1}|\theta^*_j)$ can be replaced with the class conditional predictive distribution $p(x_{n+1}|D_j)$ where $D_j = \{x_i\}_{i\in C^j}$ denotes the subset of samples belonging to class y^*_j defined by the index set C^j. Thus, provided that class membership information for all samples processed before x_{n+1} are known, the decision function to assign x_{n+1} to a new class or one of the existing ones can be expressed as:

$$h\left(x_{n+1}\right) = \begin{cases} y_{n+1} = y^*_j & \text{if } \frac{n_{j^*}}{\alpha+n}p\left(x_{n+1}|D_{j^*}\right) \geq \frac{\alpha}{\alpha+n}p\left(x_{n+1}\right) \\ y_{n+1} = y^*_{k+1} & \text{if } \frac{n_{j^*}}{\alpha+n}p\left(x_{n+1}|D_{j^*}\right) < \frac{\alpha}{\alpha+n}p\left(x_{n+1}\right) \end{cases} \tag{3.6}$$

where $j^* = \text{argmax}_j \left\{\frac{n_j}{\alpha+n}p(x_{n+1}|D_j)\right\}_{j=1}^{k}$. However, in the nonexhaustive learning framework class membership information is only available for samples initially present in the training dataset. For all new samples processed before x^{th}_{n+1} sample the true class membership information is unknown and needs to be predicted.

3.2 Inference with a Nonexhaustive Set of Classes by Gibbs Sampling

As we move on to discussing how inference can be performed in this framework, we introduce new notation to distinguish between the two types of samples available during execution: samples initially available in the training dataset with known class membership information and those with no verified class membership information, i.e.

unlabeled samples. Let $X = \{x_1, \ldots, x_\ell\}$ be the set of all training samples initially available, $Y = \{y_1, \ldots, y_\ell\}$ be the corresponding set of known class indicator variables with $y_i \in \{1, \ldots, k\}$, k being the number of known classes, $\tilde{X}^n = \{\tilde{x}_1, \ldots, \tilde{x}_n\}$ be the set of n unlabeled samples and $\tilde{Y}^n = \{\tilde{y}_1, \ldots, \tilde{y}_n\}$ be the corresponding set of *unknown* class indicator variables with $\tilde{y}_i \in \left\{1, \ldots, \tilde{k} + k\right\}$, $\tilde{k}$ being the number of unrepresented classes associated with these n samples.

We are interested in predicting $\tilde{Y}_{n+1}$, i.e., the class labels for all $\tilde{X}_{n+1}$ at the time $\tilde{x}_{n+1}$ is observed, which can be done by finding the predictive distribution, $p(\tilde{Y}^{n+1}|\tilde{X}^{n+1}, X, Y)$. Although integrating out the parameters can be difficult, the closed form solution for the conditional distributions of the latent variables, $\tilde{y}_i$, can be obtained easily. So we can do Gibbs sampling with the sampler state consisting of variables $\tilde{y}_i$, $i = \{1, \ldots, n+1\}$, to approximate $p(\tilde{Y}^{n+1}|\tilde{X}^{n+1}, X, Y)$. One sweep of the Gibbs sampler evaluates the following conditional distribution $\forall i$:

$$p(\tilde{y}_i|\tilde{Y}^{(n+1)/i}, \tilde{X}^{n+1}, X, Y) \propto \frac{\alpha}{\alpha+n+\ell} p(\tilde{x}_i)\delta_{\tilde{k}+k+1} + \frac{1}{\alpha+n+\ell} \sum_{j=1}^{k+\tilde{k}} n_j p(\tilde{x}_i|D_j)\delta_j \qquad (3.7)$$

where $\tilde{Y}^{(n+1)/i}$ denotes $\tilde{Y}^{(n+1)}$ without $\tilde{y}_i$. Based on this distribution $\tilde{y}_i$ is assigned class id $\tilde{k} + k + 1$, i.e., $\tilde{x}_i$ is assigned to a new class, by a probability $\frac{\alpha}{\alpha+n+\ell} p(\tilde{x}_i)$ or class id j, i.e., $\tilde{x}_i$ is assigned to class Ω_j, by a probability $\frac{n_j}{\alpha+n+\ell} p(\tilde{x}_i|D_j)$, where Ω_j can be an observed as well as an unobserved class.

Unlike standard DPM models for unsupervised learning, in the proposed partially-observed DPM (PO-DPM) framework we already know that labeled samples originate from observed classes, whereas unlabeled samples can originate from observed as well as unobserved classes. So the task is to infer the class membership of unlabeled samples only, i.e. $\tilde{Y}^n$, while utilizing the labeled samples in estimating the prior parameters for the observed classes. Referring to the CRP analogy, each class corresponds to a table and we can imagine labeled samples in each class as *a group of customers* arriving at the restaurant simultaneously and sitting together at a table. On the contrary, unlabeled samples are more like *undecided* customers as yet to choose an existing table or a new one in the restaurant. As a result, we do inference only for unlabeled samples, $\tilde{x} \in \tilde{X}^n$, by evaluating the Gibbs update in (3.7)

iteratively given the state of all other variables. The described method that uses labeled and unlabeled samples together in the learning process could be discussed as a semi-supervised learning approach, yet we defer a more elaborate discussion on semi-supervised learning in partially-observed settings to Chapter 4.

BayesNoDe Algorithm Revisited via DPM

As an aside, we also considered the described approach in Chapter 2, which processed data sequentially, in the DP mixture framework with the same *Gaussian X Inverse-Wishart* assumption over the parameters. The sequential processing of the incoming data in the former approach corresponds to a *one-pass Gibbs sampling* on the class indicator variables and effectively results in the same outcome. In other words, we inferred the known and unknown classes equally well without needing a specific γ threshold value. On the other hand, Gibbs sampling usually needs many iterations before converging to the equilibrium state. Once the Gibbs sampler runs for a predefined number of sweeps, samples from the first several sweeps are ignored to account for the burn-in rate of the sampler. The state with the maximum a posteriori probability (MAP) is chosen as the optimum state and $\tilde{y}_{n+1}$ is predicted by the corresponding label assigned to $\tilde{x}^{n+1}$ at this state. In addition to predicting $\tilde{y}_{n+1}$, the optimum state also simultaneously defines the class assignments of all samples observed before $\tilde{x}^{n+1}$. This way unobserved classes with rapidly accumulating samples, which are potential candidates for emerging classes, can be identified earlier in the process. Yet, this regular Gibbs sampling scheme is extremely inefficient when data arrive sequentially, because of the repeated sampling of previous data for each incoming one.

In fact, for a fully online treatment of sequential data, we can use particle filters for inference as described in [43]. However, the focus of this thesis is on inference with data available as a batch and thus we continue dealing with Gibbs sampling

algorithms. In the following section we discuss the implementation of the Gibbs update formula with the specific data model.

3.3 A Normally Distributed Data Model

The Gibbs sampler requires evaluating the predictive $p(\tilde{x}_i|D_j)$ and the marginal $p(\tilde{x}_i)$ distributions. The predictive distribution for both observed and unobserved classes can be obtained by integrating out θ. The marginal distribution can be obtained from $p(\tilde{x}_i|D_j)$ by setting D_j an empty set. In general the exact solutions for the predictive and marginal distributions do not exist and approximations are needed. However, as presented next, a closed-form solution does exist for a Normally distributed data model and a properly chosen base distribution.

For each ω_j we consider a Gaussian distribution with mean μ_j and covariance Σ_j, i.e., $\omega_j \sim \mathcal{N}(\mu_j, \Sigma_j)$. For the mean vector and covariance matrix, we use a joint conjugate prior G_0:

$$G_0 = p(\mu, \Sigma) = \underbrace{\mathcal{N}\left(\mu|\mu_0, \frac{\Sigma}{\kappa}\right)}_{p(\mu|\Sigma)} \times \underbrace{W^{-1}(\Sigma|\Sigma_0, m)}_{p(\Sigma)} \tag{3.8}$$

where μ_0 is the prior mean and κ is a scaling constant that controls the deviation of the class conditional mean vectors from the prior mean. The smaller the κ is, the larger the between class scattering will be. The parameter Σ_0 is a positive definite matrix that encodes our prior belief about the expected Σ. The parameter m is a scalar that is negatively correlated with the degrees of freedom. In other words the larger the m is the less Σ will deviate from Σ_0 and vice versa.

To evaluate the update formula in (3.7) we need $p(x_{n+1}|D_j)$. To obtain $p(x_{n+1}|D_j)$ we need to integrate out $\theta = \{\mu, \Sigma\}$. Since the sample mean $\bar{x}$ and the sample covariance matrix S are sufficient statistics for the multivariate Normally distributed data, we can write $p(\mu, \Sigma|D_j) = p(\mu, \Sigma|\bar{x}_j, S_j)$. The formula for this posterior and sketch of its derivation is available in books on multivariate statistics [33]. Once we

integrate out $p(x_{n+1}, \mu, \Sigma|\bar{x}_j, S_j)$ first with respect to μ and then with respect to Σ we obtain the predictive distribution in the form of a multivariate Student-t distribution:

$$p\left(x_{n+1}|D_j\right) = t\left(\frac{n_j\bar{x}_j + \kappa\mu_0}{n_j+\kappa}, \frac{\Sigma_0 + n_jS_j + \frac{n_j\kappa}{n_j+\kappa}(\bar{x}_j-\mu_0)(\bar{x}_j-\mu_0)^T}{\frac{(\kappa+n_j)(m+n_j-d+1)}{(\kappa+n_j+1)}}, m+n_j-d+1\right) \tag{3.9}$$

where the three parameters in (3.9) are the location vector, the positive definite scale matrix, and the degrees of freedom, respectively. In addition to $p(x_{n+1}|D_j)$ we need $p(x_{n+1})$ when evaluating the decision function in (3.6), which is also a multivariate Student-t distribution with D_j being an empty set. Thus, we can obtain $p(x_{n+1})$ from (3.9) by setting n_j equal to zero and eliminating all terms involving x_j, S_j.

3.3.1 Estimating the Parameters of the Prior Model

The parameters $(\Sigma_0, m, \mu_0, \kappa)$ of the prior model can be estimated beforehand using samples from the well-defined classes. The same argument in section 2.2.2 for Σ_0 and m applies here as well; thus, we estimate Σ_0 by S_p, i.e., the pooled covariance, and m by maximizing the marginal likelihood of $(n_j-1)S_j$ for $m > d+1$ numerically. Once again, S_p is defined by:

$$S_p = \frac{(m-d-1)\sum_{j=1}^{k}(n_j-1)S_j}{n-k} \tag{3.10}$$

where n is the total number of samples in the training set, i.e., $n = \sum_{j=1}^{k} n_j$. The marginal distribution of $(n_j-1)S_j$ can be obtained by integrating out the joint distribution $p((n_j-1)S_j, \Sigma_j) = p((n_j-1)S_j|\Sigma_j)p(\Sigma_j)$ with respect to Σ_j. For a Normal data model $p((n_j-1)S_j|\Sigma_j)$ is a Wishart distribution with a scale matrix Σ_j and degrees of freedom n_j-1, i.e., $(n_j-1)S_j|\Sigma_j \sim W(\Sigma_j, n_j-1)$ and $p(\Sigma_j)$ is an inverted-Wishart distribution as defined in (3.8). The parameters κ and μ_0 can be estimated by maximizing the joint likelihood of $\bar{x}$ and S, $p(\bar{x}, S)$, with respect to κ and μ_0, respectively, which results in $\hat{\mu}_0 = \sum_{j=1}^{k}\bar{x}_j/k$ for μ_0 and $\hat{\kappa} = (kd)\left(\sum_{j=1}^{k}\left(n_j(\bar{x}_j-\hat{\mu}_0)^T(\hat{\Sigma}_0+n_jS_j)^{-1}(\bar{x}_j-\hat{\mu}_0)(n_j+m-d)\right)-kd\right)^{-1}$ for κ.

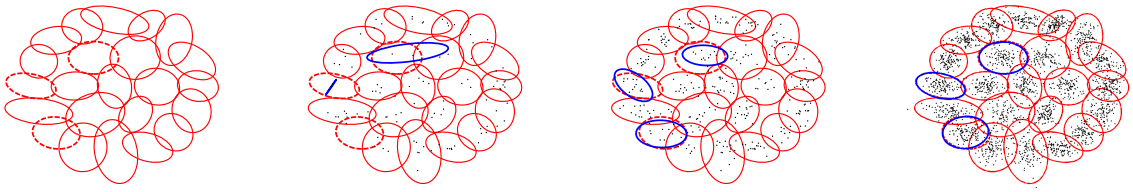
3.4 Experiments

3.4.1 An Illustrative Example

We present an illustrative example for the NEL-DPP algorithm discovering and modeling classes with a 2-D simulated dataset. We generate twenty three classes where the class covariance matrix of each class is obtained from an inverted-Wishart distribution with parameters $\Psi = 10I$ and $m = 20$ and mean vectors are equidistantly placed alongside the peripheries of two circles with radius 4 and 8 creating a flower-shaped dataset. Here, I denotes the 2-D identity matrix. Three of the twenty three classes are randomly chosen as unrepresented. The nonexhaustive training data contains twenty classes with each class represented by 100 samples (a total of 2000 samples) whereas the exhaustive testing data contains twenty three classes with 100 samples from each (a total of 2300 samples). The objective here is to discover and model the three unrepresented classes while making sure samples of represented classes are classified as accurately as possible. Figure 3.3(a) shows true class distributions for all twenty three classes. The represented classes are shown by solid lines and unrepresented ones by dashed lines. The ellipses correspond to the distributions of the classes that are at most three standard deviations away from the mean. The testing samples are processed and incidentally classified by the NEL-DPP algorithm. We chose the precision parameter α as 2. Figures 3.3(b), 3.3(c), and 3.3(d) demonstrate the discovery and modeling of new classes when 50, 500, and all 2300 test samples are classified, respectively. The discovered classes are marked by solid blue lines. All three classes are discovered and their underlying distributions are successfully recovered by generating one cluster for each class. Of the 300 samples belonging to these three classes 281 of them are correctly identified as novelties (Sensitivity=93.7%) and of the 2000 samples belonging to the represented classes 1996 of them are correctly identified as known (Specificity=99.8%).

When a class is discovered for the first time there is only one sample associated with it. Hence, initial class contours do not approximate the true distributions well.

However, as more samples are assigned to these classes the contours gradually improve to more accurately approximate the true distributions.



(a) True class distributions (b) 100/2300 classified (c) 300/2300 classified (d) 2300/2300 classified

Figure 3.3. Illustration of the proposed algorithm with an artificial dataset. (a) Red dashed lines indicate unrepresented classes. Red solid lines indicate represented classes. (b)-(d) Blue solid lines indicate newly discovered classes. Black '·' marks indicate testing samples.

3.4.2 Bacteria Detection

A total of 2054 samples from 28 classes each representing a different bacteria serovar were considered in this study. These are the type of serovars most commonly found in food samples. Each serovar is represented by between 40 to 100 samples where samples are the *forward-scatter patterns* characterizing the phenotype of a bacterial colony obtained by illuminating the colony surface by a laser light. Each scatter pattern is a gray level image characterized by a set of 50 features. More information about this dataset is available in [44]. Samples are randomly split into two as train and test, with 80% of the samples going into the training set and the remaining 20% in the test. Stratified sampling is used to make sure each class is proportionately represented in both the training and the test sets. Two different pairs of train/test sets are generated, each by removing a different set of unrepresented classes from the original training data. For the first pair the most separated four classes are identified. These are classes that would be classified with a close to perfect accuracy if they were represented in the training set. For the second pair the least separated four classes are identified. These are classes that would be classified with a relatively poor accuracy

even when they were represented in the training set. Then, for each pair, all samples of the four unrepresented classes are moved from the training set to the test set. In each pair the nonexhaustive training set contains 24 classes whereas the exhaustive set contains 28 classes. These two pairs of training/test sets are labeled the most-separated and the least-separated pairs.

The performance of the NEL-DPP algorithm is evaluated on two fronts: novelty detection and class discovery. Novelty detection is evaluated by the area under the Receiver Operating Characteristic (ROC) curve (AUC) obtained on the test data. Here, sensitivity is defined as the percent of samples from unrepresented classes identified as novelty and specificity is defined as the percent of samples from represented classes classified into one of the represented classes. The performance of the NEL-DPP algorithm is compared against three other techniques: support vector domain description (SVDD) [8], resampling approach [45], and a version of the proposed algorithm implemented with a static training set (NoDe-DPP). NoDe-DPP detect novelties without modeling them and is considered in this experiment to see the direct impact of novelty modeling on the overall results. To account for the effect of the order of the samples for the sampling process we repeated the experiments with 20 random permutations of the samples. We obtained different operating points on the ROC curve for the NEL-DPP and NoDe-DPP algorithms by varying α from 0 to ∞. Results in Table 3.4.2 show the AUC achieved by the four techniques using the *most-separated* and *least-separated* training/test pairs.

The AUC values show that the proposed approach significantly outperforms the other three techniques irrespective of whether the most-separated or least separated pair is used. When the unrepresented classes are well-separated the proposed approach almost yields the ideal AUC value. These results also illustrate the positive impact of online novelty modeling on novelty detection as demonstrated by the difference in the performances of NEL-DPP and NoDe-DPP. The low standard deviation across the 20 runs indicate the results are robust to changes in the execution order of test samples.

Table 3.1
Comparing results of novelty detection in terms of AUC values achieved by the four techniques using the two different training/test set pairs. Numbers in parenthesis are standard deviations across multiple runs.

	NEL-DPP	NoDe-DPP	SVDD	Resampling
most-separated	0.99 (0.00)	0.97	0.92	0.96
least-separated	0.96 (0.00)	0.93	0.72	0.90

Table 3.2
Performance of the NEL-DPP algorithm on class discovery using the two different training/test set pairs. Numbers in parenthesis indicate standard deviations across multiple runs.

	most-separated				least-separated			
unrepresented classes	1	2	3	4	1	2	3	4
avg. # of clusters	1.2	1.7	2.0	2.1	2.3	1.7	2.6	3.5
% recovery rate (%)	98.0	100.0	100.0	99.0	67.0	52.0	40.0	57.0
	(1.2)	(0.0)	(1.7)	(1.6)	(8.0)	(13.0)	(15.0)	(15.0)

When evaluating NEL-DPP for class discovery we considered two criteria: number of newly discovered classes and percent recovery rate for each unrepresented class. To compute these two values we assigned each discovered class to the unrepresented class having the majority of the samples in that class. Percent recovery rate is computed by the ratio of the number of samples recovered from an unrepresented class to the total number of samples in that class. We determined the precision parameter α by encoding our prior belief about the current size and nature of bacterial serovars using the idea that we adopted from [18] and modifying it into a suitable heuristic for our needs. Again, to account for the effect of the execution order of test samples we repeated the experiment twenty times with random permutations in each. The results showing the average number of classes discovered and percent recovery rate for each unrepresented class are shown in table 3.4.2.

3.5 Discussion

The data model used in this chapter was limited with the Normal model. We can extend it to problems involving more flexible class distributions by choosing a mixture model for each class data and a hierarchical DP model over class distributions. The learning problem then can be better formulated as a semi-supervised learning (SSL) problem as labeled and unlabeled samples are actively involved in the learning and inference process, yet again the labeled data being partially-observed. Thus, in the following chapter we will present the resulting approach as a semi-supervised learning framework.

4 SELF-ADJUSTING MODELS FOR SEMI-SUPERVISED LEARNING IN PARTIALLY-OBSERVED SETTINGS

In this chapter we present a novel semi-supervised approach to learning with a non-exhaustive training data set. The main motivation for this approach is to extend the PO-DPM algorithm in Chapter 3 to handle non-Normal class distributions. To achieve this, we model each class (both observed and unobserved) as a mixture of Normal distributions and replace the prior DP model over the class distributions with a hierarchical Dirichlet Process (HDP) model. Unlike the previous approach, while processing unlabeled samples, here we allow the self-adjusting model to add new mixture components into observed classes in addition to novel components for prospective unobserved classes. Therefore, during the inference both labeled and unlabeled samples are actively involved in the MCMC sampling process. As a result, it turns out to be a semi-supervised learning problem in a partially-observed setting.

4.1 Semi-Supervised Learning from Nonexhaustive Data

Despite close to two decades of active research in semi-supervised learning (SSL) there is still no consensus among researchers whether unlabeled data helps with learning. Numerous results reported over the years, with some studies showing significant improvements in classifier performance when unlabeled data is used along with labeled data, yet others presenting results [46–48] suggesting that semi-supervised learning is nothing but a hype, clearly indicate that the controversy surrounding semi-supervised learning will not come to an end anytime soon.

So far it has been theoretically proved that: 1. in the context of finite mixture models when the 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, under the

additional assumption of statistical identifiability, unlabeled data alone is sufficient to identify mixture components; 2. under various assumptions, classification error decreases exponentially with the number of labeled samples, and linearly with the number of unlabeled samples [49]; 3. under a zero-bias assumption, unlabeled data reduces the variance of the estimator and helps classification [50]. Although these results are strong and present the ideal conditions under which unlabeled data would be useful, the assumptions on which they are based are far from realistic for real-world data. It is now an established fact in semi-supervised learning that when there is a mismatch between approximating and true distributions, unlabeled data may actually degrade the accuracy of the classifier. Thus, it is somewhat of a dichotomy, to expect a distribution learned with limited labeled data to be flexible enough to accommodate a large amount of unlabeled data.

In most semi-supervised settings the limited labeled data is not only scarce but also collected without full knowledge of the underlying components of the data-generating mechanism. The main challenge that arises in the mining of real-world data sets but is often overlooked in semi-supervised learning is that the data model is not only unknown at the time of training but may also have an evolving nature that makes learning with a fixed model impractical. Under such circumstances it would be impractical to assume that labeled and unlabeled data sets come from the same distribution because certain aspects of the data-generating mechanism evident at the time the unlabeled data set was observed may not have been evident at the time the labeled data set was collected. In other words, it is natural to have a labeled dataset where the sets of classes and components are not exhaustively defined. The *Pathogen Detection* and *Hyperspectral Data Analysis* problems introduced in Chapter 1 perfectly fit to the described scenario and we will be experimenting our approach on these applications.

4.1.1 Our Approach and Contributions

Non-exhaustiveness of the labeled data set is a very realistic yet ill-defined scenario where traditional approaches to semi-supervised learning with a fixed model assumption would fail, as there is a clear mismatch between the model defined by the labeled data set and the model that generated the unlabeled data set. In this study we present a new framework for semi-supervised learning by replacing the traditional brute-force approach of fitting a fixed model onto the unlabeled data set with a new approach that can enable "data to speak for itself". We believe that our approach differs significantly from earlier work in that we relax the fixed model assumption defined by the labeled data in order to have a self-adjusting model that can evolve by dynamically adding new components or classes to better accommodate unlabeled data.

We model each class by a Gaussian mixture model (GMM) with an unknown number of components. We define a hierarchical Dirichlet process (HDP) over class distributions to dynamically model the number of components as well as classes. HDP also offers a natural framework for parameter sharing across inter- and intra-class components, practically addressing the ill-defined covariance estimation problem even for components observed with only few samples. We use a collapsed Gibbs sampler to perform inference and to estimate the posterior distribution of the component indicator variables for all samples in the labeled and unlabeled data sets. Our specific contributions in this study can be summarized as follows:

1. We propose a new framework for semi-supervised learning where unlabeled data can potentially improve learning even when the models that generated the unlabeled and labeled data sets are different.

2. We extend the concept of HDP, which allows joint learning of components across a fixed number of observed classes, to learning components of potentially infinite number of unobserved classes in addition to those of observed ones.

3. We provide a strategy for sharing the covariance matrices across different components while leaving the mean vectors free. Sharing both the covariance matrix and the mean vectors across different components implies using the same Gaussian distribution across different classes, which raises the issue of statistical identifiability.

4. New class discovery and discovery of new components from existing classes comes as a by-product of our approach.

4.1.2 Previous Work in Semi-Supervised Learning

Self-training is a widely used approach in semi-supervised learning. A single classifier is first trained using the small amount of labeled data and then applied to the unlabeled data to predict their labels. The most confident unlabeled samples along with their predicted labels are then moved into the training data and the classifier is retrained with the updated training set. The algorithm iterates in this fashion by learning from its own predictions until a convergence criterion is met.

Co-training, introduced by Blum & Mitchell [51], is a multi-view approach to SSL. The assumption is that features can be divided into two sets (views), which (ideally) are conditionally independent given the class information, and each subset alone can be used to obtain a good classifier. The approach involves two classifiers, each trained initially on one of the two feature subsets of the labeled data. Each classifier evaluates the unlabeled data and determines a few samples with highest confidence in their predicted labels. Those samples and their predicted labels are then incorporated into the training set of the other classifier so that each one is retrained with the respective updated data. The algorithm repeats this procedure until a termination criterion is met. Many other studies have followed, some analyzing the co-training paradigm in detail [52, 53], some relaxing the assumptions on the feature subsets [54, 55], others building new concepts upon it such as statistical co-Learning [56] and its improved version democratic co-Learning [57] or tri-training [58].

Another major line of work consists of graph-based methods where nodes represent the labeled and unlabeled samples in the dataset and edges indicate the similarity of nodes, which can be weighted by the pairwise distance between samples. A common assumption is that data samples with similar features tend to lie in the same class, i.e. smoothness of labels. Unlike generative models, graph-based methods are transductive, meaning that no general decision rule covering the entire data space is obtained, but only the labels of the test data are targeted. Many graph-based methods estimate a function on the graph in a graph-cut [59], a label propagation [60, 61] or a regularization [62, 63] framework, details of which are beyond the scope of this paper.

Yet another group of methods exploits the low-density separation assumption, that the classes in the unlabeled data form clusters and that the decision boundary between classes pass through a low-density region. Some of the approaches are transductive support vector machines (TSVM) [64–66], Gaussian process based approaches [67,68], information regularization [69], and entropy minimization, to name a few.

Generative mixture models have been a classical approach to SSL. As long as the conditional distribution is an identifiable mixture model, such as a Gaussian mixture, the large amount of unlabeled samples can be used to infer the parameters of the mixture components, commonly through expectation-maximization (EM) [27]. Then labeled samples can be used to determine the component labels assuming that the classes exhibit a well clustering property. Applications of this technique include [70–72]. All these earlier studies in generative mixture models assume a fixed structure, i.e., all samples in the unlabeled data are assumed to originate from one of the classes represented in the labeled data. As discussed earlier this is not a very realistic assumption for most real-world datasets. One of the earliest approaches to tackle this problem was introduced within the context of supervised learning [73]. In this study, known and unknown classes were modeled by a mixture of expert model with learning performed by expectation-maximization. The optimal number of mix-

ture components was determined by using minimum description length coupled with some heuristics.

Most of what we did in this study is more closely related to the partially observed Dirichlet process mixture models, PO-DPM, presented in Chapter 3. Another recent study that involves the application of DPM for partially observed settings was introduced in [74]. This study models training data by a HDP and introduces a DP model to handle incoming data. Incoming data contains samples from observed as well as unobserved classes. HDP and DP models were then coupled with the goal of identifying news articles with new topics while classifying those with older topics into one of the classes represented in the training data.

The proposed approach is similar to the last two studies mentioned above in terms of using a DP/HDP model in a partially observed setting. However, in addition to distinct algorithmic aspects discussed throughout this chapter, we believe that our study pioneers the approach to learning with a non-exhaustively defined labeled data set and presents a unique framework to tackle unlabeled data in semi-supervised settings.

The rest of this chapter is organized as follows. In Section 4.2.1 we briefly review the hierarchical Dirichlet process (HDP). In Section 4.2.2 we discuss how HDP can be extended to partially observed settings. In Section 4.2.3 we incorporate the data model and discuss a strategy for sharing the covariance matrices while leaving the mean vectors free. In Section 4.2.4 we demonstrate the feasibility of the proposed approach on an artificial dataset. In Section 4.3 we present results of our experiments comparing the proposed approach against several state-of-the-art supervised and semi-supervised learning techniques for the bacteria classification and hyperspectral image analysis problems.

4.2 Bayesian Nonparametric Approach to Semi-Supervised Learning

We start this section with a brief review of the Hierarchical Dirichlet Processes (HDP) [75] widely used in the machine learning literature for co-clustering multiple groups of data by enabling sharing of parameters across components. Throughout this section we use the terms *group* and *class* interchangeably. We also assume that each group data comes from a mixture model with an unknown number of components.

4.2.1 Hierarchical Dirichlet Processes (HDP)

HDP extends Dirichlet Processes (DP) [28], which is mainly used in clustering and density estimation problems as a nonparametric prior defined over the number of mixture components. HDP models each group of data in the form of a DPM model, where DPM models across different groups are connected together through a higher level DP. We use the notation $x_{ji} \in \Re^d$, $i = \{1, ..., n_j\}$, $j = \{1, ..., J\}$ to identify sample i in the group j where n_j denotes the number of samples in group j, J is the total number of groups, and θ_{ji} defines the parameters of the mixture component associated with x_{ji}. Each x_{ji} is associated with a mixture component defined by the parameter θ_{ji}, which is generated i.i.d. from a Dirichlet Process as follows:

$$\begin{array}{rcll} x_{ji}|\theta_{ji} & \sim & p(\cdot|\theta_{ji}) & \text{for each j, i} \\ \theta_{ji}|G_j & \sim & G_j & \text{for each j, i} \end{array} \tag{4.1}$$

where G_j's are random probability measures distributed i.i.d. according to a DP with base distribution G_0 and precision parameter α.

To reiterate, the stick-breaking construction due to [40] suggests $G_j = \sum_{i=1}^{\infty} \beta_{ji} \delta_{\theta_{ji}}$ where $\beta_{ji} = \beta'_{ji} \prod_{l=1}^{i-1}(1 - \beta'_{jl})$, $\beta'_{ji} \sim Beta(1, \alpha)$, and $\theta_{ji} \sim G_0$. The points θ_{ji} are called the *atoms* of G_j. Note that unlike continuous distributions the probability of sampling the same θ_{ji} twice is not zero and proportional to β_{ji}. Thus, G_j is considered a discrete distribution. The precision parameter, α, is the parameter that controls how much of the stick will be left for subsequent values. The smaller the α is, the larger the β'_{ji} will be, and the less of the stick will be left for subsequent values on

average. Thus, α is the parameter that controls the prior probability of assigning a new sample to a new component and thus, plays a critical role in the number of components generated.

In the HDP model the base distribution G_0 is distributed according to a higher level DP with a base distribution H and parameter γ. This hierarchical model couples G_j's and allow for sharing of mixture components within and between groups. HDP model is completed as follows:

$$\begin{aligned} G_j|G_0,\alpha &\sim DP(G_0,\alpha) \qquad \text{for each j,} \\ G_0|H,\gamma &\sim DP(H,\gamma) \end{aligned} \tag{4.2}$$

The generative process defined by an HDP model can be better explained by an analogy to the Chinese Restaurant Franchise (CRF) [75]. We have a restaurant franchise with a global menu of dishes shared across all restaurants. In each restaurant a certain dish is served at each occupied table, which is shared by all customers sitting in that table. The same dish can be served in other tables across multiple restaurants. The popularity of a particular dish is proportional to the number of tables serving that dish. In an arbitrary restaurant j, customer i is associated with θ_{ji} and is seated at table t_{ji}, and table t is associated with one of the K random draws from H, i.e., $\psi_{jt} \in \{\phi_1,\ldots,\phi_K\}$, which represents the global menu of dishes. A dish from the global menu served at table t in restaurant j is denoted by the indicator variable k_{jt}. In the HDP model the parameter γ controls the prior probability of serving a new dish at a new table.

In this model, restaurants correspond to classes, each table in a restaurant corresponds to a mixture component in the mixture model, and each dish in the menu corresponds to a unique set of parameters shared by one or more components.

The conditional distributions for t_{ji} and k_{jt} are obtained by integrating out G_j and G_0, respectively:

$$t_{ji}|t_{j1},\ldots,t_{j,i-1},\alpha \sim \tfrac{\alpha}{n_j+\alpha}\delta_{t^{new}} + \textstyle\sum_{t=1}^{m_j.} \tfrac{n_{jt}}{n_j+\alpha}\delta_t \tag{4.3}$$

where $m_j.$ is the number of tables in restaurant j and n_{jt} is the number of customers at table t in restaurant j. According to this conditional distribution θ_{ji} inherits one

of the existing ψ_{jt} with probability $\frac{n_{jt}}{n_j+\alpha}$ or $\psi_{j,m_j.+1}$, i.e., a new table, with probability $\frac{\alpha}{n_j+\alpha}$. Similarly,

$$k_{jt}|k_{j1},\ldots,k_{j,t-1},\gamma \quad \sim \quad \frac{\gamma}{m_{..}+\gamma}\delta_{k^{new}} + \sum_{k=1}^{K}\frac{m_{.k}}{m_{..}+\gamma}\delta_k \tag{4.4}$$

where $m_{.k}$ is the number of tables across all restaurants serving dish ϕ_k and $m_{..}$ is the total number of tables across all restaurants. According to this conditional distribution ψ_{jt} is equal to one of the ϕ_k with a probability $\frac{m_{.k}}{m_{..}+\gamma}$ or ϕ_{K+1}, i.e., a new dish, with probability $\frac{\gamma}{m_{..}+\gamma}$.

Inference in the described CRF setting can be performed using a Gibbs sampler by iteratively sampling the variables $\boldsymbol{t} = \left\{\{t_{ji}\}_{i=1}^{n_j}\right\}_{j=1}^{J}$, $\boldsymbol{k} = \left\{\{k_{jt}\}_{t=1}^{m_{j.}}\right\}_{j=1}^{J}$, and $\boldsymbol{\phi} = \{\phi_k\}_{k=1}^{K}$ given the state of all other variables.

The conditional distributions for t_{ji} is:

$$p(t_{ji}=t|\boldsymbol{t}\backslash t_{ji},\boldsymbol{k},\boldsymbol{\phi},\boldsymbol{x}) \propto \begin{cases} \alpha p(x_{ji}) & \text{for } t = m_{j.}+1 \\ n_{jt}^{-i}p(x_{ji}|\phi_{k_{jt}}) & \text{for } t \in \{1,\ldots,m_{j.}\} \end{cases} \tag{4.5}$$

The conditional distributions for k_{jt} is:

$$p(k_{jt}=k|\boldsymbol{t},\boldsymbol{k}\backslash k_{jt},\boldsymbol{\phi},\boldsymbol{x}) \propto \begin{cases} \gamma\prod_{i:t_{ji}=t}p(x_{ji}) & \text{for } k = K+1 \\ m_{.k}^{-jt}\prod_{i:t_{ji}=t}p(x_{ji}|\phi_k) & \text{for } k \in \{1,\ldots,K\} \end{cases} \tag{4.6}$$

In the above conditional distributions n_{jt}^{-i} is the number of customers sitting at table t in restaurant j not including the customer i, $m_{.k}^{-jt}$ is the number of tables sharing the same dish ϕ_k not including the table t in the restaurant j. The conditional distribution for $\boldsymbol{\phi}$ is omitted as we choose a conjugate pair of H and $p(\cdot|\phi)$ in this study, which allows us to integrate out ϕ analytically to obtain a collapsed version of the Gibbs sampler.

4.2.2 Partially-Observed HDP Model (PO-HDP)

In this section we extend the HDP model to semi-supervised learning in partially-observed settings. We model each class by a Gaussian mixture model (GMM) with an

unknown number of components. We introduce the notion of observed and unobserved classes/subclasses to distinguish classes/subclasses represented in the labeled data set from those not represented. Each subclass is represented by a single component in the corresponding GMM model. Thus, we use subclasses and components interchangeably in the rest of the paper.

The labeled data set is non-exhaustively defined because the set of classes and the set of components for some or all of the classes are not complete, i.e., partially observed. The class labels for samples in the labeled data set are known, whereas component labels are not. The unlabeled data set may contain samples from classes and subclasses not represented in the labeled data set. However, neither the class labels nor the component labels are known for samples in the unlabeled data set. The number of components in each class and the total number of classes are also not known.

In the partially-observed setting the learning problem includes the following two tasks: (*i*) inferring the component membership of labeled samples and (*ii*) inferring both the group and component membership of unlabeled samples. Unlike labeled samples which are known to originate from observed classes, unlabeled samples can originate from observed as well as unobserved classes. Notice that each class in the proposed SSL framework corresponds to a separate restaurant in the CRF concept. To relate this partially observed setting to the CRF analogy each unlabeled sample can be considered as an *undecided* customer who has not yet decided which restaurant to go. These customers can go to one of the restaurants in the franchise but may as well choose an out-of-franchise restaurant, which is treated as a new restaurant with a single table in the proposed framework. Labeled samples represent customers who already arrived at one of the franchise restaurants and waiting to be seated. These customers can be seated using the same Gibbs sampler scheme presented in the previous section after accounting for the presence of undecided customers who eventually choose to go to the same restaurant. In short, *decided* customers sit at existing or new tables in existing restaurants only, whereas *undecided* customers can

seat at new tables in new restaurants in addition to existing or new tables in existing restaurants. Before we move on to describing the details of our approach for extending the HDP framework for semi-supervised learning in a partially-observed setting we introduce new notation to distinguish between labeled and unlabeled samples.

We use $\boldsymbol{x} = \left\{\{x_{ji}\}_{i=1}^{n_j}\right\}_{j=1}^{J}$ and $\boldsymbol{t} = \left\{\{t_{ji}\}_{i=1}^{n_j}\right\}_{j=1}^{J}$ to denote samples and component indicator variables, respectively, for the labeled data. For the same variables in the unlabeled data, we use $\tilde{\boldsymbol{x}} = \{\tilde{x}_i\}_{i=1}^{n_u}$ and $\tilde{\boldsymbol{t}} = \left\{\tilde{t}_i\right\}_{i=1}^{n_u}$. For the unlabeled data we also introduce $\tilde{\boldsymbol{y}} = \{\tilde{y}_i\}_{i=1}^{n_u}$ to denote the unknown class indicator variables, where $\tilde{y}_i \in \{1, \ldots, J + \bar{J}\}$, $\bar{J}$ is the number of newly created classes after observing the unlabeled data. Finally we use $\boldsymbol{k} = \left\{\{k_{jt}\}_{t=1}^{m_{j\cdot}}\right\}_{j=1}^{J}$ and $\tilde{\boldsymbol{k}} = \left\{\tilde{k}_j\right\}_{j=1}^{\bar{J}}$ to define indicator variables for the unique parameter sets shared across observed and newly created classes, respectively.

The part of Gibbs sampler for inferring the component membership of labeled samples involve evaluating the following conditional distributions iteratively given the state of all other variables.

The conditional distribution for t_{ji} for a labeled sample is:

$$p(t_{ji} = t|\boldsymbol{t}\backslash t_{ji}, \boldsymbol{k}, \boldsymbol{\phi}, \boldsymbol{x}, \tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}, \tilde{\boldsymbol{t}}) \propto \begin{cases} \alpha\, p(x_{ji}) & \text{for } t = m_{j\cdot} + 1 \\ (n_{jt}^{-i} + \tilde{n}_{jt}) p(x_{ji}|\phi_{k_{jt}}) & \text{for } t \in \{1, \ldots, m_{j\cdot}\} \end{cases} \tag{4.7}$$

where $\tilde{n}_{jt}$ is the number of unlabeled samples assigned to component t in class j. Unlike a labeled sample, which is either assigned to one of the existing components associated with its class of origin or to a new component generated for that class, an unlabeled sample can be assigned to any of the existing components across all classes or to a new component generated for a new class. In this framework each new class will inherently have one component. The fact that true labels of unlabeled samples are not known makes it impossible to readily associate new components with existing ones.

The conditional distribution for $\tilde{t}_i$ for an unlabeled sample is:

$$p(\tilde{t}_i = t|\boldsymbol{t}, \boldsymbol{k}, \boldsymbol{\phi}, \boldsymbol{x}, \tilde{\boldsymbol{x}}, \tilde{\boldsymbol{t}} \backslash \tilde{t}_i, \tilde{\boldsymbol{y}}, \tilde{\boldsymbol{k}}) \propto$$
$$\begin{cases} \alpha\, p(\tilde{x}_i) & \text{for } t = 1 \qquad j = J + \bar{J} + 1 \\ (n_{jt} + \tilde{n}_{jt}^{-i}) p(\tilde{x}_i|\phi_{k_{jt}}) & \text{for } t \in \{1, \ldots, m_{j\cdot}\} \quad j \in \{1, \ldots, J + \bar{J}\} \end{cases} \tag{4.8}$$

Next, we discuss the part of the Gibbs sampler for inferring the indicator variables of unique parameters for components of existing and new classes.

A component in an existing class may contain both labeled and unlabeled samples. Thus, the conditional distribution for k_{jt} is:

$$p(k_{jt} = k|\boldsymbol{t}, \boldsymbol{k} \backslash k_{jt}, \boldsymbol{\phi}, \boldsymbol{x}, \tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}, \tilde{\boldsymbol{t}}, \tilde{\boldsymbol{k}}) \propto$$
$$\begin{cases} \gamma \prod_{i:t_{ji}=t} p(x_{ji}) \prod_{i:\tilde{t}_i=t \wedge \tilde{y}_i=j} p(\tilde{x}_i) & \text{for } k = K + 1 \\ m_{\cdot k}^{-jt} \prod_{i:t_{ji}=t} p(x_{ji}|\phi_k) \prod_{i:\tilde{t}_i=t \wedge \tilde{y}_i=j} p(\tilde{x}_i|\phi_k) & \text{for } k \in \{1, \ldots, K\} \end{cases} \tag{4.9}$$

On the other hand a component in a new class contains only unlabeled samples. Thus, the conditional distribution for $\tilde{k}_j$ is:

$$p(\tilde{k}_j = k|\boldsymbol{t}, \boldsymbol{k}, \boldsymbol{\phi}, \boldsymbol{x}, \tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}, \tilde{\boldsymbol{t}}, \tilde{\boldsymbol{k}} \backslash \tilde{k}_j) \propto$$
$$\begin{cases} \gamma \prod_{i:\tilde{y}_i=j} p(\tilde{x}_i) & \text{for } k = K + 1 \\ m_{\cdot k}^{-j} \prod_{i:\tilde{y}_i=j} p(\tilde{x}_i|\phi_k) & \text{for } k \in \{1, \ldots, K\} \end{cases} \tag{4.10}$$

Finally, the class indicator variables $\tilde{\boldsymbol{y}}$ for unlabeled samples can be obtained from $\tilde{\boldsymbol{t}}$. If an unlabeled sample is assigned to a new component this will indicate a new class and thus $\tilde{y}_i = J + \bar{J} + 1$. If an unlabeled sample is assigned to one of the existing components associated with class j then $\tilde{y}_i = j$. Note that class j can be one of the classes represented in the labeled data set as well as one of the classes previously associated with unlabeled samples, i.e., $j \in \{1, \ldots, J + \bar{J}\}$. Each sweep of the Gibbs sampler also involves sampling γ and α values using the technique described in [42].

This completes our discussion for learning with labeled and unlabeled data sets with an HDP model in a partially-observed setting. Next, we will present the data model used in this study and discuss a strategy for sharing the covariance matrices of mixture components while leaving their mean vectors free.

4.2.3 Parameter Sharing in a Gaussian Mixture Model

We model each class by a mixture model with each component data distributed according to a Gaussian distribution with mean vector μ_{jt} and a covariance matrix Σ_{jt}, i.e., $\psi_{jt} = \{\mu_{jt}, \Sigma_{jt}\}$. For the base distribution H, from which the component parameters ϕ_k's are sampled, we define a conjugate prior:

$$H = p(\mu, \Sigma) = \underbrace{\mathcal{N}\left(\mu|\mu_0, \frac{\Sigma}{\kappa}\right)}_{p(\mu|\Sigma)} \times \underbrace{W^{-1}(\Sigma|\Sigma_0, m)}_{p(\Sigma)} \tag{4.11}$$

where μ_0 is the prior mean and κ is a scaling constant that controls the deviation of the mean vectors of mixture components from the prior mean. The smaller the κ is, the larger the scattering between the components will be. The parameter Σ_0 is a positive definite matrix that encodes our prior belief about the expected Σ. The parameter m is a scalar that is negatively correlated with the degrees of freedom. In other words the larger the m is the less Σ will deviate from Σ_0 and vice versa. The parameters $(\Sigma_0, m, \mu_0, \kappa)$ are estimated using labeled samples in the same way described in section 3.3.1.

To evaluate the Gibbs sampler introduced in the previous section we need the conditional distribution $p(x|\phi_{k_j})$ and the marginal distribution $p(x)$. Since ϕ_{k_j} are not known they can be replaced with the class conditional predictive distributions $p(x|D_{jt})$, where D_{jt} denotes the subset of samples belonging to component t in class j. This collapsed version of the Gibbs sampler reduces the state space of the sampler and leads to faster convergence to the equilibrium distribution [39]. The marginal distribution can be obtained from $p(x|D_{jt})$ by setting D_{jt} an empty set. For the multivariate Gaussian data the sample mean $\bar{x}$ and the sample covariance matrix S are sufficient statistics and therefore we can write $p(x|D_{jt}) = p(x|\bar{x}_{jt}, S_{jt})$.

To evaluate $p(x|D_{jt})$ for a given x requires evaluating the following integral with respect to $\psi_{jt} = \{\mu_{jt}, \Sigma_{jt}\}$.

$$p(x|D_{jt}) = \int p(x|\psi_{jt}) p(\psi_{jt}|D_{jt}) \partial\psi_{jt} \tag{4.12}$$

If parameter sharing across different components were not allowed, evaluating the above integral analytically would yield a multivariate Student-t distribution with the following parameters.

Location vector:

$$\hat{\mu} = \frac{n_{jt}\bar{x}_{jt} + \kappa\mu_0}{n_{jt} + \kappa}$$

Scale matrix:

$$\hat{\Sigma} = \frac{\Sigma_0 + (n_{jt} - 1)S_{jt} + \frac{n_{jt}\kappa}{n_{jt}+\kappa}(\bar{x}_{jt} - \mu_0)(\bar{x}_{jt} - \mu_0)^T}{\frac{(\kappa+n_{jt})\,v}{(\kappa+n_{jt}+1)}} \tag{4.13}$$

Degrees of freedom:

$$v = m + n_{jt} - d + 1$$

However, in the proposed framework the clustering property of the HDP model allows multiple components to inherit one of the distinct parameters in $\boldsymbol{\phi}$. Thus, instead of integrating out ψ_{jt} as in (4.12), sharing property of the HDP model requires that we integrate out ϕ_k in the predictive distribution. Let $D_{.k}$ be the samples of all components sharing parameter ϕ_k then the predictive distribution $p(x|D_{.k})$ can be obtained by evaluating the following integral:

$$p(x|D_{.k}) = \int p(x|\phi_k)p(\phi_k|D_{.k})\partial\phi_k \tag{4.14}$$

If ϕ_k contains both the mean vector and the covariance matrix then this would imply sharing the same mean vector and the covariance matrix across multiple components. This would mean fitting each component by the same Gaussian distribution, which would not make sense as components sharing the same parameters would no longer be identifiable. To tackle this problem we adopt a strategy, where sharing is limited with the covariance matrices only. Thus, if we set $\phi_k = \{\Sigma_k\}$ and evaluate the integral in (4.14) analytically we obtain the predictive distribution as a multivariate Student-t distribution with the same location vector as previously but with the scale matrix and degrees of freedom updated as follows.

Scale matrix:

$$\hat{\Sigma} : \frac{\Sigma_0 + \sum_{jt:k_{jt}=k}(n_{jt} - 1)S_{jt} + \frac{n_{jt}\kappa}{n_{jt}+\kappa}(\bar{x}_{jt} - \mu_0)(\bar{x}_{jt} - \mu_0)^T}{\frac{(\kappa+n_{jt})\,v}{(\kappa+n_{jt}+1)}} \tag{4.15}$$

Degrees of freedom:

$$v : m + \sum_{jt:k_{jt}=k} (n_{jt} - 1) - d + 2$$

where the summation terms are over all components sharing the same covariance matrix.

In (4.13) v in the denominator has an averaging affect on the accumulating scatter matrices in the numerator. Practically speaking, as the components grow during the iteration of samples, they tend to resemble other components with similar scattering of samples, which effectively improves the convergence of the sampling algorithm and at the same time addresses the ill-defined covariance estimation problem for components with very few samples.

Next, we demonstrate the PO-HDP approach discovering and recovering new classes and new components of observed classes on a synthetic 2D data set.

4.2.4 Illustration of the PO-HDP Approach

We generated ten classes, each as a mixture of three Gaussian components. The covariance matrices for individual components are randomly drawn from a template set of five covariance matrices, each with different shape and orientation (Figure 4.1). The mean vectors of the classes are equidistantly placed alongside the peripheries of two concentric circles with radii 15 and 7 and whose centers are located at the origin. The component means are arbitrarily chosen alongside a circle centered at the class mean.

We generated 50 samples from each component making 150 samples for each class. We sequestered 30% of these samples as test data by stratified sampling and use the remaining 70% for training. Out of the training samples, 30% are considered as labeled data and 70% as unlabeled. In order to produce a nonexhaustively-defined labeled data set both in terms of the number of classes and the number of components for an observed class, we considered all of the components of two of the classes and

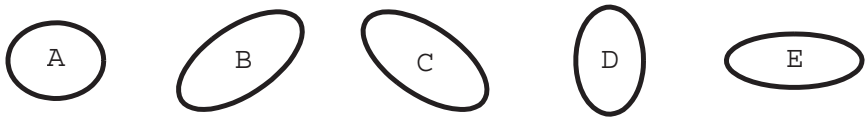


Figure 4.1. The template of covariance matrices used for the illustrative example

one of the components of a third class as unobserved and move all of their samples from the labeled set into the unlabeled set.

The objective here is to discover and recover the two unobserved classes and the unobserved component of the observed class while making sure the samples from all other observed classes are classified as accurately as possible. In Figure 4.2 the true distributions of the observed classes are shown by solid lines and the unobserved ones by dashed lines. The ellipses correspond to the distribution of the individual components that are at most three standard deviations away from its mean.

The inferred component distributions for unobserved components are overlayed with the true component distributions in Figure 4.2. The observed classes are marked by solid blue lines, the components discovered by the proposed approach for the unobserved classes and the unobserved component of an observed class are marked by solid red and cyan lines, respectively.

4.2.5 Implementation Details for PO-HDP

We end this section by briefly discussing some of the implementation details involving the Gibbs sampler presented in Section 4.2.2. We initialize the HDP model by generating a component for each observed class and assigning a random sample from that class to this component. During each sweep of the Gibbs sampler, all data samples are assigned to one of the existing components or to a new component using equations (4.7) and (4.8) for labeled and unlabeled samples, respectively. This

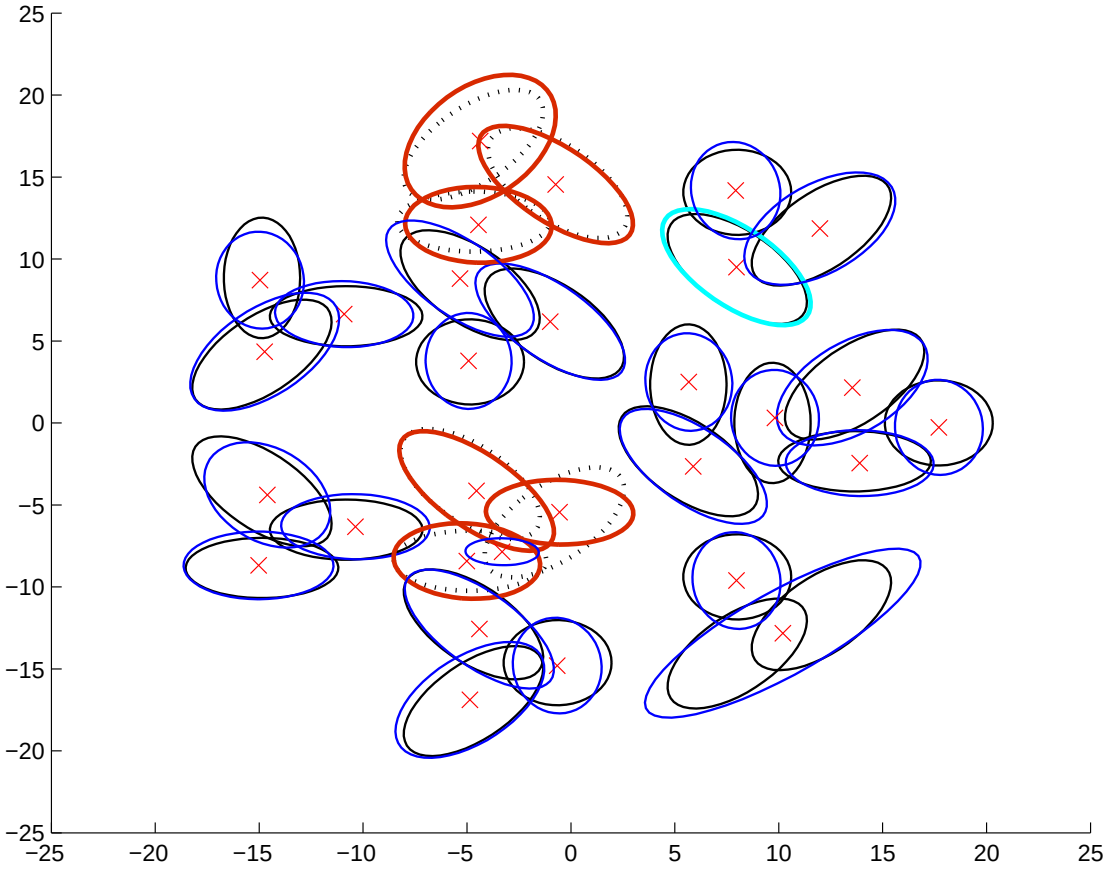


Figure 4.2. True class distributions of the observed classes are displayed with solid black curves and those of the unobserved classes with dashed black curves. The single unobserved component from an observed class is shown with solid cyan curve. The results of the SA-SSL approach for observed and unobserved classes are displayed with solid blue and red curves, respectively.

is followed by sampling the parameters of the components based on the most current assignment of the samples. For components associated with observed classes the equation (4.9) is used, for those associated with unobserved classes the equation (4.10) is used. Both labeled and unlabeled samples can generate new components but unlike a component generated for an unlabeled sample, which is assigned to a new class, a new component generated for a labeled sample is readily assigned to the observed class the labeled sample belongs to.

When a sample is assigned to an existing component the mean vector of the corresponding component and the covariance matrices of all components associated with the same ϕ are updated. If an unlabeled sample ends up at a new component then we introduce a new class and tentatively label the sample with that class until the next iteration and process remaining unlabeled samples by taking the new component into account as well. Unlike an observed class, which is fixed by definition, a new class can be removed when no samples are left in the component associated with that class. A component associated with an observed class may contain both labeled and unlabeled samples at a given sweep but during later sweeps the labeled samples may move to other components leaving only unlabeled ones in that component. In this case we reassign that component to a new class.

Finally, as mentioned before, we used the formulation in [42] to sample the precision parameters α and γ of the HDP model for the Gibbs sampler. This formulation requires defining Gamma priors with shape parameters $(a0, b0)$ and $(a1, b1)$ over α and γ, respectively. The posterior distribution for α is conditioned on the total number of samples N and the total number of existing components $m_{..}$ in the current iteration. Similarly, the posterior for γ is conditioned on $m_{..}$ and the number of existing unique parameters in the current iteration K. While experimenting with the shape parameters of the Gamma priors, we observed that as $m_{..}$ increases it suppresses the effect of $a0$ in the posterior and the expected value of the Gamma posterior tends to increase regardless of $a0$. Regarding the second parameter of the posterior, as N is fixed, a large value for $b0$ is necessary to balance the effect of the first parameter on

the expected value. A similar argument can be made for γ based on the values of K and T. As a result we set $a0$ and $a1$ to one and coarsely tuned $b0$ and $b1$ values. We used the same $b0 = 100$ and $b1 = 50$ values for the experiments presented in Sections 4.3.2 and 4.3.3.

4.3 Experiments

4.3.1 A Comparative Illustration

For this illustration we generated three classes, each as a mixture of three Gaussian components. The covariance matrices for individual components are randomly drawn from a set of five different templates of covariance matrices, each with a different shape and orientation (Figure 4.1). The mean vectors of the classes are equidistantly placed along the periphery of a circle centered at the origin with radius 7. Similarly, the component means are arbitrarily chosen along a circle with radius 1, centered at the corresponding class means.

We generated 110 samples from each component for a total of 330 samples for each class. We randomly selected 10 samples from each component as labeled data and used the remaining 100 samples from that component as unlabeled data. In order to produce a partially-observed labeled data set in terms of both the number of classes and the number of components for an observed class, we considered all components of a class and one component of a second class as unobserved and discarded all their labeled samples, leaving only unlabeled samples from these components.

The purpose of this illustration is three fold. First, we show that the proposed HDP model, which uses unlabeled and labeled data together, can more accurately recover the underlying distributions of the observed classes compared to the version that uses only labeled data. Second, we demonstrate that the proposed self-adjusting model can successfully discover and recover the underlying distributions of classes/subclasses that exist in the unlabeled data but are unobserved in the labeled data, whereas classical approaches that deal with samples of unrepresented

classes/subclasses by assigning reduced weight to them can neither discover unobserved classes nor accurately model observed classes. Third, we illustrate the sharing aspect of the proposed approach by first identifying the types of the covariance matrices of the recovered distributions and then comparing them against the true types of the covariance matrices used to generate data from each subclass. We show that with the proposed approach the labels of the covariance matrices shared among recovered subclass distributions perfectly match the labels of those shared among true subclass distributions.

Figure 4.3(a) shows true subclass distributions for all nine subclasses. The observed subclasses, i.e., those that are represented in the labeled data set, are shown by solid lines and unobserved ones by dashed lines. The ellipses correspond to the distributions of the subclasses that are at most three standard deviations away from the mean.

Figure 4.3(b) shows the distributions of the five observed subclasses recovered by the version of the HDP model that uses only labeled data. Note that the recovered distributions deviate from the true subclass distributions. Additionally three of the five recovered subclass distributions share different types of covariance matrices than those used in the true subclass distributions.

Figure 4.3(c) shows the impact of unlabeled data over the recovered subclass distributions when unlabeled data contain samples from classes/subclasses unobserved in the labeled data and a fixed model is used to accommodate unlabeled data. These results are obtained using the technique introduced in [72], which assigns reduced weight to unlabeled samples as determined by their posterior probabilities. Note that, since unlabeled data from unobserved subclasses dominate labeled data from observed classes, the recovered distributions for observed classes significantly deviate from true subclass distributions.

Figure 4.3(d) shows the results of the proposed approach. Both observed and unobserved subclass distributions are almost perfectly recovered. The sharing of the covariance matrices among recovered subclass distributions matches the sharing of

covariance matrices among true subclass distributions. Labels for the covariance matrices of recovered distributions are also correctly identified.

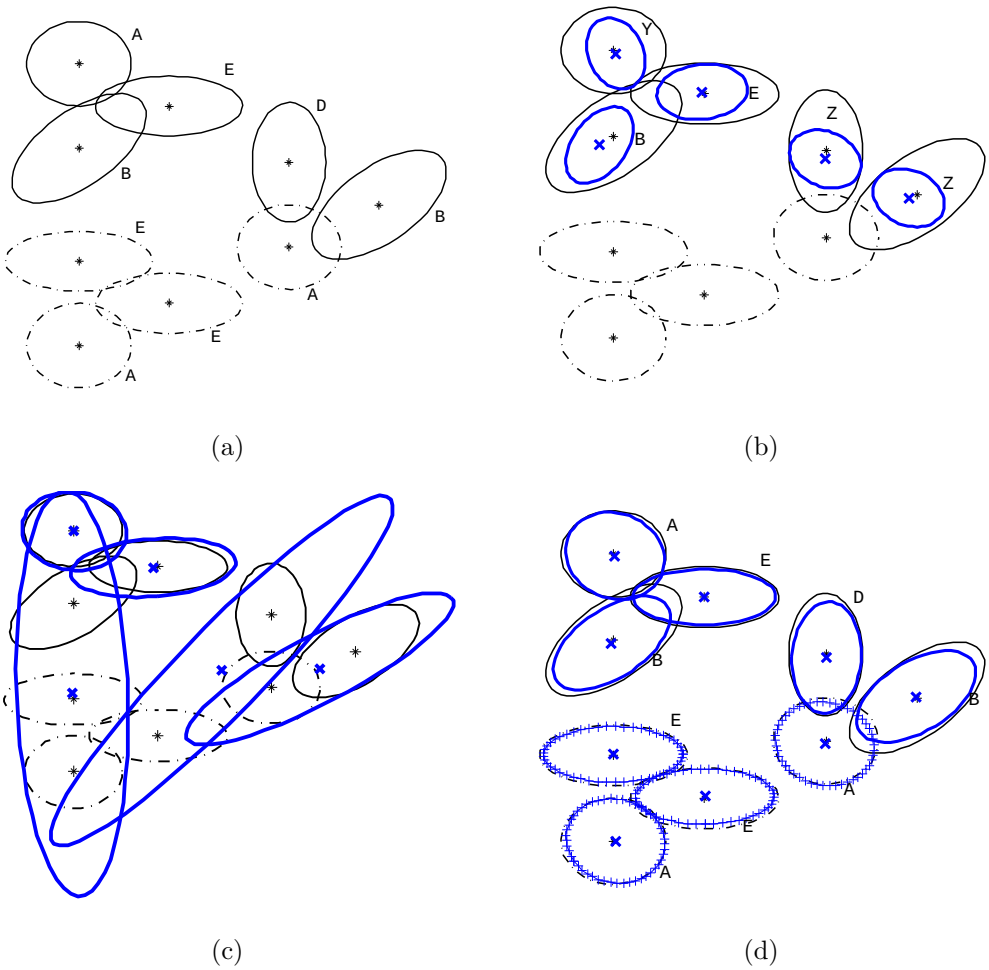


Figure 4.3. Illustration of the proposed algorithm with an artificial dataset. Solid and dashed black contours indicate observed and unobserved subclasses, respectively. Solid blue contours indicate recovered versions of observed subclasses whereas blue contours plotted with the plus sign indicate the recovered versions of unobserved subclasses. Letters denote the labels of the covariance matrices. The star and cross signs show the location of the true and predicted mean vectors of subclasses, respectively. (a) True subclass distributions. (b) Distributions recovered by the standard HDP model using only labeled data set. (c) Distributions recovered by a fixed model that assigns full weight to labeled samples and reduced weight to unlabeled samples, using both labeled and unlabeled data sets. (d) Distributions recovered by the proposed self-adjusting model using both labeled and unlabeled data sets.

4.3.2 Experiments on Bacteria Detection and Remote Sensing

In this section we aim to observe the classification accuracy of our approach compared with a number of approaches from literature using the available labeled data in both experiments.

Evaluated Classifiers

For these experiments we considered three supervised learning methods as baseline techniques, where only the labeled training samples is used for learning the classifiers. The first one is a Naive Bayes classifier (SL-NB). The second one is a maximum-likelihood classifier with each class modeled by a single Gaussian (SL-ML). The third one is a maximum-likelihood classifier with each class modeled by a mixture of Gaussian components (SL-EM). This method fits a mixture model onto each class data by Expectation-maximization.

In addition to these supervised learning methods we implemented a number of benchmark semi-supervised learning algorithms. The first one is the semi-supervised EM algorithm introduced in [50] (SSL-EM). Briefly, this algorithm first, fits a Gaussian distribution onto each class data in the labeled data set, then, it evaluates the posterior probabilities of the unlabeled samples for each class using the learned distributions, finally, it incorporates unlabeled samples into the parameter estimation process for each class by weighting them by their posterior probabilities. This process repeats until convergence and the resulting classifier is applied on the test data.

We also implemented two versions of the self-training method (SELF) with base learners ML and NB, respectively. Another algorithm we have included is the Co-training algorithm (CO-TR) implemented in two versions with base learners ML and NB respectively. For SELF and CO-TR we only include the better performing version in the experimental results.

One last approach we considered is the semi-supervised adaptation of the algorithm introduced in [73] (SSL-MOD). In this technique, similar to SSL-EM, we esti-

mate an initial Gaussian model for each class using the labeled samples and classify the unlabeled samples. The maximum of the class likelihood values are obtained for each sample. A two component Gaussian mixture model is fit onto this likelihood data in order to identify unlabeled samples with higher and lower likelihood values. We expect that unlabeled samples belonging to the observed classes will yield higher likelihood values whereas those from unobserved classes will yield low likelihood values. Then we merge the unlabeled samples in the higher-likelihood group with the labeled samples to re-estimate the parameters of the classes. This process repeats until convergence and in the end another EM is performed on the samples remaining in the low-likelihood group to identify unobserved components. This technique is the only SSL technique, other than the proposed approach, that attempts to model unobserved classes. The proposed self-adjusting SSL approach is identified by SA-SSL in this section.

Classifier Design and Evaluation

The labeled, unlabeled, and test data sets are generated as follows. We first divide the available labeled data into two and reserve one portion as test data. Then we further split the remaining portion into two as the labeled and unlabeled training data sets. During each split stratified sampling is used to make sure each class is proportionately represented in each subset. Some of the classes are considered *unobserved* and moved from the labeled set to the unlabeled set generating a non-exhaustive labeled data set. Both the unlabeled and test sets are exhaustive. The exact numbers for the number of unobserved classes and the proportions for the test, train and unlabeled sets are specified for each experiment below. We evaluate the performance of the classifiers using the overall classification accuracy and the average classification accuracies evaluated separately for observed and unobserved classes on the test set. We repeated this process ten times by generating ten random

test/train/unlabeled splits and report the average accuracies along with the standard deviations.

The performance of the proposed SA-SSL algorithm is evaluated on three fronts: overall classification accuracy, classification accuracy for observed classes, classification accuracy for unobserved classes. To compute the classifier accuracy for unobserved classes each newly created component is assigned to the unobserved class having the majority of the samples in that component. Classification accuracy for each unobserved class is computed by the ratio of the total number of samples recovered by the corresponding components to the total number of samples in that class.

Experiment 1: Pathogen Detection

In this experiment a total of 2054 samples from 28 classes each representing a different bacteria serovar were considered. These are the type of serovars most commonly found in food samples. Each serovar is represented by between 40 to 100 samples where samples are the *forward-scatter patterns* characterizing the phenotype of a bacterial colony obtained by illuminating the colony surface by a laser light. Each scatter pattern is a gray level image characterized by a set of 22 features. More information about this dataset is available in [44]. We removed 30% of the samples as test data, and half of the remaining 70% is treated as the labeled data set and the other half as unlabeled. Four of the classes are considered unobserved and all of their samples are moved from the labeled set to the unlabeled set. So the non-exhaustive labeled set contains 24 classes and the exhaustive unlabeled and test data contains all of the 28 classes.

As the results in Table 4.3.2 suggest the proposed SA-SSL algorithm significantly outperforms all other techniques in terms of overall classifier accuracy as well as classifier accuracies for observed and unobserved classes. In addition to classifying samples from unobserved components with a reasonable accuracy, the proposed approach also

Table 4.1
Average of 10 iterations, each run with different test/train/unlabeled splits of the Bacteria dataset. The first column shows the overall accuracy on the test samples, second and third columns show the accuracies for the observed and unobserved classes, respectively.

Method	Acc	Acc-O	Acc-U
SA-SSL	0.81 (0.03)	0.84 (0.01)	0.68 (0.2)
SSL-EM	0.50 (0.02)	0.59 (0.02)	0
SSL-MOD	0.44 (0.03)	0.48 (0.03)	0.23 (0.09)
SELF	0.62 (0.01)	0.73 (0.02)	0
CO-TR	0.62 (0.01)	0.74 (0.01)	0
SL-ML	0.64 (0.02)	0.76 (0.02)	0
SL-NB	0.52 (0.02)	0.62 (0.02)	0
SL-EM	0.30 (0.05)	0.35 (0.06)	0

performs favorably compared to other techniques for classifying samples of observed components.

Experiment 2: Multi-spectral Image Data Set

We used the Flightline C1 multispectral image data set for this experiment. This is a 12-band multispectral image taken over Tippecanoe County, Indiana by the M7 scanner in June, 1966. There are eight classes, each class representing a different crop type. The data set consists of 949 scan lines with 220 pixels per line for a total of 208,780 pixels, 69,413 of which are available as labeled pixels. More information about this multispectral imagery is available in [76]. This data set has been previously studied in the remote-sensing literature within the context of both supervised and semi-supervised learning problems [72, 77, 78]. However, these earlier studies picked training samples from each and every class across the image, making sure that the list of classes in the training data set is complete. Considering the spatially evolving nature of remote-sensing imagery in general and this image data set in particular

Table 4.2
Average of 10 iterations each run with different test/train/unlabeled splits of the multi-spectral image dataset. The first column shows the overall accuracy on the test samples, second and third columns show the accuracies for the observed and unobserved classes, respectively.

Method	Acc	Acc-O	Acc-U
SA-SSL	0.92 (0.01)	0.91 (0.01)	0.98 (0.01)
SSL-EM	0.75 (0.10)	0.78 (0.10)	0
SSL-MOD	0.77 (0.06)	0.79 (0.07)	0.0
SELF	0.82 (0.01)	0.85 (0.01)	0
CO-TR	0.81 (0.02)	0.84 (0.02)	0
SL-ML	0.84 (0.01)	0.87 (0.01)	0
SL-NB	0.77 (0.02)	0.80 (0.02)	0
SL-EM	0.77 (0.01)	0.80 (0.01)	0

we do not believe that earlier studies have realistically analyzed this image data set. Besides, if the classifier had access to labeled samples from each and every field on the scene, we would not need a semi-supervised learning algorithm in the first place because one can easily augment the labeled data set by assigning the same label to all samples within a crop field.

In this data set we used 0.2% of all samples as the labeled data set, 5% as the unlabeled data set and the remaining samples are left for testing. One class is considered unobserved and moved from the labeled data set to the unlabeled data set, leaving a total of 121 labeled samples from seven classes for the non-exhaustive labeled set and around 3000 samples from all classes in the unlabeled set. The proposed SA-SSL significantly outperforms all other techniques compared and recovers the one missing class with an almost perfect accuracy while achieving a fairly good accuracy for observed classes.

4.3.3 Experiments on Entire Remote Sensing Images

In this section we further observe the scalability of the proposed approach on two remote sensing data sets, first being the entire FlightLine C1 image with over 200K samples and the second being a hyperspectral image with 126 dimensions. In addition to the accuracy on labeled samples we will obtain entire classification map on each data set.

Classifier Models

We included the same algorithms from section 4.3.2 and an additional one from the remote sensing literature for comparison with our approach. The additional one is the transductive SVM algorithm introduced in [79] (SSL-SVM). The original TSVM algorithm [64] solves an optimization problem to maximize the margin between two classes using the combined set of labeled and unlabeled samples, whereas SSL-SVM incorporates a subset of the unlabeled samples into the learning process in an iterative manner. The method starts with a regular SVM trained using only labeled samples and iterates each time by extending the training data set with the unlabeled samples closest to the positive and negative margins of the separating hyperplane.

Classifier Design and Evaluation

The proposed framework is evaluated not only based on how accurately it classifies samples of known classes, i.e., classes that exists on the top half of the image, but also how well it discovers and recovers the missing classes and subclasses. We will use the ground truth available for the bottom half of the image to compute the overall classification accuracy as well as classification accuracies for known and unknown classes separately. To compute the classifier accuracy for each class each component is assigned to the class having the majority of the samples in that component. Classification accuracy for each class is computed by the ratio of the total number of

samples recovered by the corresponding components to the total number of samples in that class.

Experiment 1: Multispectral Image Data Set

In this experiment we used the Flightline C1 multispectral image data set described in section 4.3.2.

The fundamental problem in remote sensing image classification is whether or not a classifier trained using information from *known* scenes generalizes over to *unknown* scenes. Thus, to evaluate the proposed approach under more realistic settings we horizontally divide the image data set into two halves and consider the top half of the image as the known scene and the bottom half as the unknown one. We can see that the lists of classes from the two parts of the image do not fully coincide. The bottom half of the image contains two more classes (rye and alf alfa) and one additional subclass (wheat) than the top one. These fields are outlined by dashed rectangles in Figure 4.4. During our experiments labeled samples are selected only from the top part of the image whereas unlabeled samples are selected from the entire image. The labeled data set contains 1,500 samples randomly selected from the top portion of the image. The remaining samples from the labeled fields are used as unlabeled samples. The training data set contains 94,412 samples of which 1,500 are labeled and the remaining 92,912 are unlabeled. The classifiers are evaluated using only the bottom part of the image. The 3-color image of the Flightline C1 data set and the corresponding labeled field map are shown in Figure 4.4(a) and Figure 4.4(b), respectively.

Results and Analysis

Classifier accuracies for all techniques are listed in Table 4.3.3. The first column includes classifier identifiers, the second, third, and fourth columns show overall, known, and unknown class accuracies, respectively. The color and BW version of the classification maps generated, and the number of components used to model each

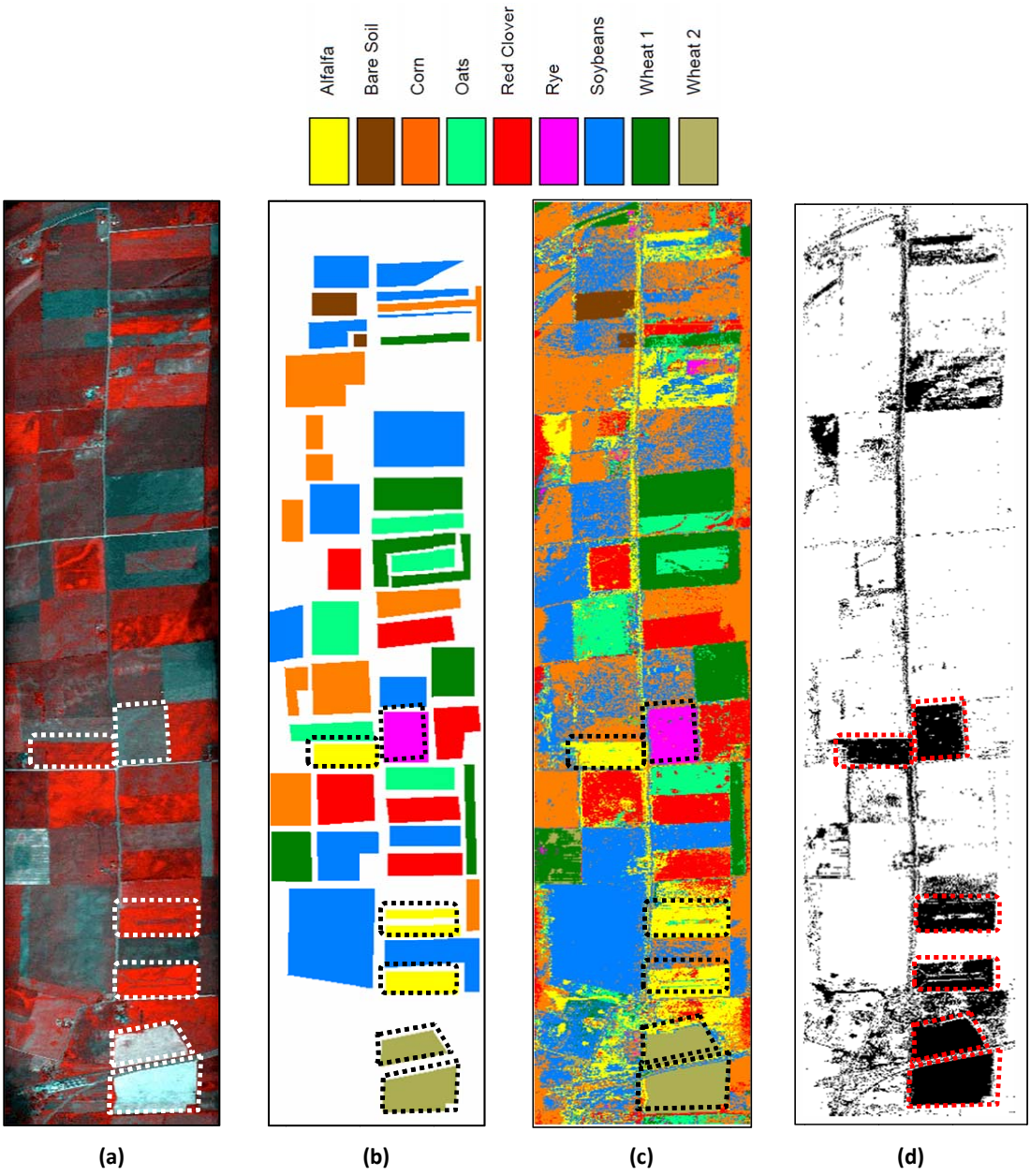


Figure 4.4. (a) 3-color (R:11, G:9, B:7) image of the Flightline C1. (b) Labeled field map. (c) Classification map obtained by the proposed SA-SSL approach. (d) Black and white version of the classification map with black regions indicating new classes and white regions existing ones.

Table 4.3
The classifier accuracies for the FlightLine C1 data set. The first column shows the overall accuracy on the test samples, second and third columns show the accuracies for the observed and unobserved classes, respectively.

Method	Acc	Acc-O	Acc-U
SA-SSL	**0.83**	**0.83**	**0.81**
SSL-SVM	0.62	0.81	0
SSL-EM	0.63	0.81	0
SSL-MOD	0.60	0.78	0.01
SELF	0.61	0.79	0
CO-TR	0.63	0.82	0
SL-ML	0.63	0.82	0
SL-NB	0.56	0.73	0
SL-EM	0.63	0.82	0

class by the proposed SA-SSL algorithm are shown in Figure 4.4(c), Figure 4.4(d), and Table 4.3.3, respectively. Dashed rectangles in Figure 4.4(c) and Figure 4.4(d) indicate newly discovered fields.

These results show that the proposed approach successfully discovers and recovers the two missing classes and one missing subclass with a fairly good accuracy while achieving a classifier accuracy that is comparable to other techniques for observed classes. When we combine results from both observed and unobserved classes, we see that the proposed approach has a significantly higher overall classifier accuracy than other techniques. A total of 40 components and 33 unique covariance matrices were generated across eight classes for this data set indicating that some of the components shared covariance matrices with other components.

Table 4.4
The number of components identified for each class in the FLC1 data set.

Classes	Number of Components
Alfalfa	3
Bare Soil	1
Corn	9
Oats	3
Red Clover	2
Rye	2
Soybeans	12
Wheat	6
Wheat 2	2
Total	40

Experiment 2: Hyperspectral Image Data Set

This data set is a flightline image of the Purdue University West Lafayette campus. The hyperspectral data was collected on September 30, 1999 with the airborne HYMAP system [76], providing image data in 126 spectral bands in the visible and infrared regions (0.4 2.4 μm). The system was flown at an altitude such that the pixel size is about 5 meters. The data set consists of 358 scan lines with 390 pixels per line for a total of 139,620 pixels. A 3-color image of the scene and the corresponding labeled field map based on the available ground truth are shown in Figures 4.5(a) and 4.5(b), respectively. For this data set top two thirds of the image is considered as *known* and the bottom third as *unknown*. The bottom part of the scene contains one additional class (greenhouses) than the top part. The fields belonging to greenhouses are outlined by dashed rectangles in the figures. The same classifier models described in Section 4.3.3 are also considered for this data set. The labeled data set contains 5,036 samples randomly selected from the top two thirds of the image. The remaining samples from the labeled fields are used as unlabeled samples. The training data set contains 20,973 samples of which 5,036 are labeled and the remaining 15,937 are unlabeled. The classifiers are evaluated using only the bottom part of the image. The classifiers are evaluated according to Section 4.3.3.

Results and Analysis

Classifier accuracies for all techniques are listed in Table 4.3.3. The first column includes classifier identifiers, the second, third, and fourth columns show overall, known, and unknown class accuracies, respectively. The color and BW version of the classification maps generated, and the number of components used to model each class by the proposed SA-SSL algorithm are shown in Figure 4.5(c), Figure 4.5(d), and Table 4.3.3, respectively. Dashed rectangles in Figure 4.5 indicate newly discovered fields.

The proposed SA-SSL significantly outperforms all other techniques compared both in terms of observed and unobserved class accuracies. The greenhouse fields are

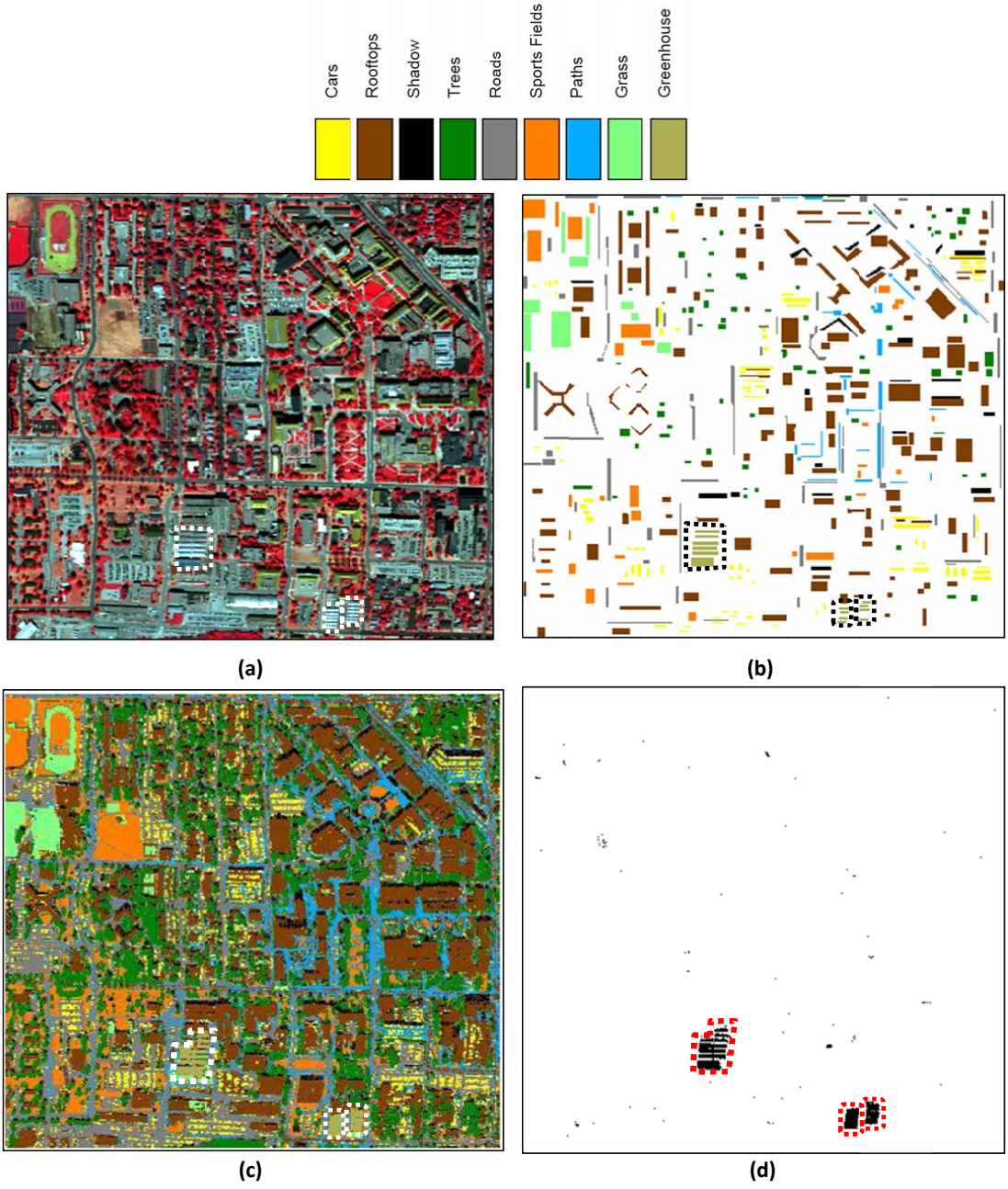


Figure 4.5. (a) 3-color (R:32, G:16, B:8) image of the flightline over the Purdue University West Lafayette campus. (b) Labeled field map. (c) Classification map obtained by the proposed SA-SSL approach. (d) Black and white version of the classification map with black regions indicating new classes and white regions existing ones.

Table 4.5
Classifier accuracies for the campus data set. The first column shows the overall accuracy on the test samples, second and third columns show the accuracies for the observed and unobserved classes, respectively.

Method	Acc	Acc-O	Acc-U
SA-SSL	**0.88**	**0.87**	**0.92**
SSL-SVM	0.79	0.88	0
SSL-EM	0.57	0.63	0
SSL-MOD	0.61	0.68	0.0
SELF	0.65	0.72	0
CO-TR	0.64	0.71	0
SL-ML	0.66	0.74	0
SL-NB	0.33	0.37	0
SL-EM	0.01	0.02	0

discovered and recovered with an almost perfect accuracy. A total of 104 components and 57 unique covariance matrices were generated across nine classes for this data set indicating that one half of the components shared covariance matrices with one of the other components.

Table 4.6
The number of components identified for each class in the campus data.

Classes	Number of Components
Cars	12
Rooftop	48
Shadow	4
Tree	5
Road	8
Sports Fields	3
Paths	7
Grass	10
Greenhouse	7
Total	104

5 SUMMARY

In this thesis, we described an ill-defined situation that may be confronted in supervised and semi-supervised settings due to a non-exhaustively defined training data set; namely learning with a partially-observed labeled data set. Traditional supervised and semi-supervised algorithms assume existence of a fixed set of classes and of an exhaustive training library representing all classes. However, in many real-world domains with evolving nature new classes may emerge on a continuous basis. Therefore obtaining a labeled data set exhaustively defined by a fixed set of classes is impractical, which leads to a partially-observed setting for training purposes. We tried to address those situations where the fixed model assumption is violated by a non-exhaustively defined training library.

In our research we introduced self-adjusting models to relax the fixed model assumption imposed on classes and their distributions. We utilize suitably chosen nonparametric priors for class distributions for both observed and unobserved classes and take advantage of the available labeled data to estimate initial parameters of the model. For any future data we allow the model to adapt itself by dynamically adding new classes/components as the data demand. This process gradually leads to a more representative model for the entire population.

Specific contributions by our research can be listed as follows:

1. We studied supervised and semi-supervised classification in the absence of some classes; in other words, with a non-exhaustive labeled data set.

2. We proposed self-adjusting generative models as an alternative to fixed ones.

3. We introduced nonparametric Bayesian models for partially-observed settings involving both Normal and non-Normal class distributions.

4. We achieved offline and online class discovery as a by-product of the self-adjusting model.

5.1 Future Work

We continue to extend the problem in two directions.

1. First, we will explore sharing of component parameters between different classes by taking into account random effects, which may be introduced in different phases of the data acquisition process. In other words, we let local components to be a noisy version of a shared component parameter to model random effects. This may help us associate newly introduced components with current classes.

2. Second, we will be exploring non-exhaustive classification of group data, where the goal is to jointly cluster group data and match clusters across groups by associating local clusters with global components. More specifically, we can allow local clusters to be perturbed versions of global components by incorporating random effects into the partially-observed hierarchical Dirichlet process model introduced in Chapter 4.

APPENDIX

A APPENDIX

A.1 Multi-Class Bacterial Dataset

Table A.1
The 28 classes from 5 species considered in this study.

Class	ID	Subclass	# of Samples
E. Coli sp.	1	O25:K98:NM ETEC	67
	2	O78:H11 ETEC	58
	3	O157:H7 01	64
	4	O157:H7 6458	87
	5	O157:H7 G5295	68
	6	K12 ATCC 29425	65
Listeria spp.	7	*L. innocua* F4248	59
	8	*L. ivanovii* 19119	81
	9	*L. monocytogenes* 19118 (4e)	94
	10	*L. monocytogenes* 7644 (1/2c)	91
	11	*L. monocytogenes* V7 (1/2a)	98
	12	*L. welshimeri* 35897	47
Salmonella spp.	13	*S.* Typhimurium (Copenhagen)	95
	14	*S.* Enteritidis 13096	89
	15	S. Enteritidis PT28	90
	16	*S.* Tennessee 825-94	78
Staphylococcus spp.	17	*S. aureus* 13301	46
	18	*S. aureus* PS103	50
	19	*S. aureus* S-41	67
	20	*S. epidermidis* PS302	31
	21	S. *epidermidis* 35547	45
	22	*S. hyicus* T6346	69
Vibrio spp.	23	*V. alginolyticus* CECT521	88
	24	*V. campbellii* CECT523	71
	25	*V. cincinnatiensis* CECT4216	89
	26	*V. hollisae* CECT5069	79
	27	*V. orientalis* CECT629	96
	28	*V. parahaemolyticus* CECT511	92
		Total	2054

LIST OF REFERENCES

LIST OF REFERENCES

[1] Terran Lane and Carla E. Brodley. Temporal sequence learning and data reduction for anomaly detection. *ACM Transactions on Information and System Security*, 2:150–158, 1998.

[2] E. Eskin, A. Arnold, M. Prerau, L. Portnoy, and S. Stolfo. A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data. In D. Barbará and S. Jajodia, editors, *Applications of Data Mining in Computer Security*. Kluwer, 2002.

[3] Dan Pelleg and Andrew Moore. Active learning for anomaly and rare-category detection. In *Advances in Neural Information Processing Systems 18*, pages 1073–1080. MIT Press, December 2004.

[4] James Theiler, , James Theiler, and D. Michael Cai. Resampling approach for anomaly detection in multispectral images. *Proceedings of SPIE*, 5093:230—240, 2003.

[5] Bernhard Schölkopf, Robert Williamson, Alex Smola, John Shawe-Taylor, and John Platt. Support vector method for novelty detection. *Advances in Neural Information Processing Systems*, 12, 2000.

[6] Jordi Muñoz-Marí, Lorenzo Bruzzone, and Gustavo Camps-Valls. A support vector domain description approach to supervised classification of remote sensing images. *IEEE Transaction on Geoscience and Remote Sensing*, 45(8):2683–2692, 2008.

[7] E. J. Spinosa and A. C. Carvalho. Support vector machines for novel class detection in bioinformatics. *Genetics Molecular Research*, 4(3):608–15, 2005.

[8] David M. J. Tax and Robert P. W. Duin. Support vector domain description. *Pattern Recognition Letters*, 20(11-13):1191–1199, 1999.

[9] Bernhard Schölkopf, John C. Platt, John C. Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Comput.*, 13(7):1443–1471, 2001.

[10] Jyrki Kivinen, Alex J. Smola, and Robert C. Williamson. Online learning with kernels. In Thomas G. Dietterich, Suzanna Becker, and Zoubin Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 785–792. MIT Press, 2001.

[11] Sethu Vijayakumar, Aaron D'Souza, and Stefan Schaal. Incremental online learning in high dimensions. *Neural Computation*, 17(12):2602–2634, 2005.

[12] Gert Cauwenberghs and Tomaso Poggio. Incremental and decremental support vector machine learning. In Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 409–415. MIT Press, 2000.

[13] Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585, 2006.

[14] Koby Crammer, Jaz S. Kandola, and Yoram Singer. Online classification on a budget. In Sebastian Thrun, Lawrence K. Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, 2004.

[15] Lehel Csató and Manfred Opper. Sparse on-line gaussian processes. *Neural Computation*, 14(3):641–668, 2002.

[16] Antoine Bordes, Seyda Ertekin, Jason Weston, and Léon Bottou. Fast kernel classifiers with online and active learning. *Journal of Machine Learning Research*, 6:1579–1619, 2005.

[17] Nicolò Cesa-Bianchi, Claudio Gentile, and Luca Zaniboni. Incremental algorithms for hierarchical classification. *Journal of Machine Learning Research*, 7:31–54, 2006.

[18] Jian Zhang, Zoubin Ghahramani, and Yiming Yang. A probabilistic model for online document clustering with application to novelty detection. In Lawrence K. Saul, Yair Weiss, and Lean Bottou, editors, *Advances in Neural Information Processing Systems 17*. MIT Press, 2005.

[19] Charu C. Aggarwal, Jiawei Han, Jianyong Wang, and Philip S. Yu. A framework for clustering evolving data streams. In *Proceedings of the 29th International Conference on Very Large Data Bases*, VLDB '03, pages 81–92, 2003.

[20] Charles Elkan. Clustering documents with an exponential-family approximation of the Dirichlet compound multinomial distribution. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 289–296, 2006.

[21] Hal Daumé III and Daniel Marcu. A Bayesian model for supervised clustering with the Dirichlet process prior. *Journal of Machine Learning Research*, 6:1551–1577, December 2005.

[22] Katherine A. Heller and Zoubin Ghahramani. Bayesian hierarchical clustering. In *Proceedings of the 22nd International Conference on Machine Learning*, ICML '05, pages 297–304, 2005.

[23] Tianbing Xu, Zhongfei (Mark) Zhang, Philip S. Yu, and Bo Long. Dirichlet process based evolutionary clustering. *IEEE International Conference on Data Mining, ICDM '08*, pages 648–657, 2008.

[24] Deepayan Chakrabarti, Ravi Kumar, and Andrew Tomkins. Evolutionary clustering. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, pages 554–560, 2006.

[25] Yangqing Jia, Shuicheng Yan, and Changshui Zhang. Semi-supervised classification on evolutionary data. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 1083–1088, 2009.

[26] G. McLachlan and D. Peel. *Finite Mixture Models*. John Wiley & Sons, New York, NY, 2000.

[27] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.

[28] Thomas S. Ferguson. A Bayesian analysis of some nonparametric problems. *Annals of Statistics*, 1(2):209–230, 1973.

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

[30] R E Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[31] Jerome H. Friedman. Regularized discriminant analysis. *Journal of The American Statistical Association*, 84(405):165–175, 1989.

[32] Tom Greene and William Rayens. Partially pooled covariance matrix estimation in discriminant analysis. *Communications in Statistics – Theory and Methods*, 18(10):3679–3702, 1989.

[33] Theodore W. Anderson. *An Introduction to Multivariate Statistical Analysis, 3rd Edition*. Wiley-Interscience, 3rd edition, 2003.

[34] Padmapriya P Banada, Karleigh Huff, Euiwon Bae, Bartek Rajwa, Amornrat Aroonnual, Bulent Bayraktar, Abrar Adil, J Paul Robinson, E Daniel Hirleman, and Arun K Bhunia. Label-free detection of multiple bacterial pathogens using light-scattering sensor. *Biosensors & Bioelectronics*, 24(6):1685–92, Feb 2009. PMID: 18945607.

[35] Murat Dundar, E. Daniel Hirleman, Arun K. Bhunia, J. Paul Robinson, and Bartek Rajwa. Learning with a non-exhaustive training dataset. A case study: Detection of bacteria cultures using optical-scattering technology. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, pages 279–288, 2009.

[36] P. W. Frey and D. J. Slate. Letter recognition using Holland-style adaptive classifiers. *Machine Learning*, 6(2), 1991.

[37] Robert E. Kass and Adrian E. Raftery. Bayes factors. *Journal of the American Statistical Association*, 90(430):773–795, 1995.

[38] N.L. Hjort. *Bayesian Nonparametrics*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2010.

[39] F. Wood and M. J. Black. A non-parametric Bayesian alternative to spike sorting. *Journal of Neuroscience Methods*, 173:1–12, 2008.

[40] Hemant Ishwaran and Lancelot F. James. Gibbs sampling methods for stick-breaking priors. *Journal of the American Statistical Association*, 96(453):pp. 161–173, 2001.

[41] David J. Aldous. Exchangeability and related topics. In *École d'Été St Flour 1983*, pages 1–198. Springer-Verlag, 1985. Lecture Notes in Math. 1117.

[42] Michael D. Escobar and Mike West. Bayesian density estimation and inference using mixtures. *Journal of the American Statistical Association*, 90:577–588, 1994.

[43] Murat Dundar, Ferit Akova, Yuan Qi, and Bartek Rajwa. Bayesian nonexhaustive learning for online discovery and modeling of emerging classes. In *29th International Conference on Machine Learning, June 26–July 1*, ICML '12, 2012.

[44] Ferit Akova, Murat Dundar, V. Jo Davisson, E. Daniel Hirleman, Arun K. Bhunia, J. Paul Robinson, and Bartek Rajwa. A machine-learning approach to detecting unknown bacterial serovars. *Statistical Analysis and Data Mining*, 3(5):289–301, 2010.

[45] Ingo Steinwart, Don R. Hush, and Clint Scovel. A classification framework for anomaly detection. *Journal of Machine Learning Research*, 6:211–232, 2005.

[46] Tong Zhang and Frank J. Oles. A probability analysis on the value of unlabeled data for classification problems. In *Proceedings of 17th International Conference on Machine Learning*, pages 1191–1198, 2000.

[47] F. Cozman and I. Cohen. Unlabeled data can degrade classification performance of generative classifiers. In *15th International Florida Artificial Intelligence Society Conference*, pages 327–331, 2002.

[48] Yuanyuan Guo, Xiaoda Niu, and H. Zhang. An extensive empirical study on semi-supervised learning. In *IEEE International Conference on Data Mining*, ICDM '10, pages 186–195, Dec. 2010.

[49] V. Castelli and T. M. Cover. The relative value of labeled and unlabeled samples in pattern recognition with an unknown mixing parameter. *IEEE Transactions on Information Theory*, 42(6):2102–2117, November 1996.

[50] B. Shahshahani and D. Landgrebe. The effect of unlabeled samples in reducing the small sample size problem and mitigating the hughes phenomenon. *IEEE Transactions on Geoscience and Remote Sensing*, 32(5):1087–1095, 1994.

[51] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the 11th Annual Conference on Computational Learning Theory*, COLT' 98, pages 92–100, New York, NY, USA, 1998. ACM.

[52] K. Nigam and R. Ghani. Analyzing the effectiveness and applicability of co-training. In *Proceedings of the 9th International Conference on Information and Knowledge Management*, pages 86–93. ACM, 2000.

[53] M.F. Balcan and A. Blum. A pac-style model for learning from labeled and unlabeled data. *Learning Theory*, pages 69–77, 2005.

[54] M.F. Balcan, A. Blum, and Y. Ke. Co-training and expansion: Towards bridging theory and practice. *Computer Science Department, Carnegie Mellon University, Pittsburgh, PA*, page 154, 2004.

[55] R. Johnson and T. Zhang. Two-view feature generation model for semisupervised learning. In *24th International Conference on Machine Learning*, ICML'07.

[56] S. Goldman and Y. Zhou. Enhancing supervised learning with unlabeled data. In *Machine Learning International Workshop Then Conference*, volume 17, pages 327–334, 2000.

[57] Y. Zhou and S. Goldman. Democratic co-learning. In *16th IEEE International Conference on Tools with Artificial Intelligence, 2004. ICTAI 2004.*, pages 594–602. IEEE, 2004.

[58] Z.H. Zhou and M. Li. Tri-training: Exploiting unlabeled data using three classifiers. *IEEE Transactions on Knowledge and Data Engineering*, 17(11):1529–1541, 2005.

[59] A. Blum and S. Chawla. Learning from labeled and unlabeled data using graph mincuts. In *18th International Conference on Machine Learning*, ICML '01, pages 19–26, 2001.

[60] M. Szummer and T. Jaakkola. Information regularization with partially labeled data. *Advances in Neural Information Processing Systems*, 15:1025–1032, 2002.

[61] X. Zhu and Z. Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical report, Technical Report CMU-CALD-02-107, Carnegie Mellon University, 2002.

[62] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Machine Learning International Workshop Then Conference*, volume 20, page 912, 2003.

[63] M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *The Journal of Machine Learning Research*, 7:2399–2434, 2006.

[64] V. Vapnik. *Statistical Learning Theory.* Wiley, New York, 1998.

[65] T. Joachims. Transductive inference for text classification using support vector machines. In *Machine Learning International Workshop Then Conference*, volume 16.

[66] T. De Bie and N. Cristianini. Semi-supervised learning using semi-definite programming. *Semi-supervised learning. MIT Press, Cambridge-Massachussets*, 2006.

[67] N.D. Lawrence and M.I. Jordan. Semi-supervised learning via gaussian processes. *Advances in Neural Information Processing Systems*, 17:753–760, 2005.

[68] V. Chu, W. Sindhwani, Z. Ghahramani, and S.S. Keerthi. Relational learning with gaussian processes. In *Advances in Neural Information Processing Systems*, volume 19, page 289. The MIT Press, 2007.

[69] A. Corduneanu and T. Jaakkola. On information regularization. In *Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence*, pages 151–158. Morgan Kaufmann Publishers Inc., 2002.

[70] D.J. Miller and H.S. Uyar. A mixture of experts classifier with learning based on both labelled and unlabelled data. *Advances in Neural Information Processing Systems*, pages 571–577, 1997.

[71] K. Nigam, A.K. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine learning*, 39(2):103–134, 2000.

[72] Q. Jackson and D.A. Landgrebe. An adaptive classifier design for high-dimensional data analysis with a limited training data set. *IEEE Transactions on Geoscience and Remote Sensing*, 39(12):2664–2679, 2001.

[73] D. J. Miller and J. Browning. A mixture model and EM-based algorithm for class discovery, robust classification, and outlier rejection in mixed labeled/unlabeled data sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(11):1468–1483, 2003.

[74] Avinava Dubey, Indrajit Bhattacharya, Mrinal Kanti Das, Tanveer A. Faruquie, and Chiranjib Bhattacharyya. Learning Dirichlet processes from partially observed groups. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, pages 141–150, 2011.

[75] Y.W. Teh, M.I. Jordan, M.J. Beal, and D.M. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.

[76] David A Landgrebe. *Signal Theory Methods in Multispectral Remote Sensing*. Wiley, Newark, NJ, 2005.

[77] Saldju Tadjudin and David A. Landgrebe. Robust parameter estimation for mixture model. *IEEE Transactions on Geoscience and Remote Sensing*, 38:439–445, 2000.

[78] Murat Dundar and David Landgrebe. Toward an optimal supervised classifier for the analysis of hyperspectral data. *IEEE Transactions on Geoscience and Remote Sensing*, 42(1):271–277, 2004.

[79] L. Bruzzone, Mingmin Chi, and M. Marconcini. A novel transductive svm for semisupervised classification of remote-sensing images. *IEEE Transactions on Geoscience and Remote Sensing*, 44(11):3363 –3373, nov. 2006.

VITA

VITA

Ferit Akova was born in Istanbul, Turkey in 1981. He received his B.S. degree in Computer Engineering and Information Science at Bilkent University in Ankara, Turkey. Upon graduation from Bilkent in 2004, he moved to the U.S. to work as a research intern at Siemens Corporate Research in Princeton, New Jersey. After working at Siemens for nearly one year he started his graduate studies at Purdue University in West Lafayette, Indiana. He obtained his Ph.D. degree in computer science from Purdue in 2013. His research interests lie in the field of machine learning where he was privileged to work with Dr. Murat Dundar and Dr. Alan Qi as his co-advisors.