Predictive Modeling using Dimensionality Reduction and Dependency Structures

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Amrudin Agovic

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
Doctor of Philosophy

Arindam Banerjee, Maria Gini

July, 2011

Acknowledgements

There are many people that deserve my gratitude. First and foremost I would like to thank my family for their continued and unconditional support throughout my studies. I thank my adviser Arindam Banerjee for his patience, support and collaboration over the last six years. I also thank my adviser Maria Gini for her support, mentoring and advise that she has given me ever since I have been an undergraduate student. I am also very grateful to Hanhuai Shan for working with me on Bayesian Multivariate Regression, and Prof. Snigdhansu Chatterjee from the department of Statistics for his input on Probabilistic Matrix Addition.

My Mathematics Masters adviser Gilad Lerman deserves my gratitude for exposing me to dimensionality reduction methods. He made it possible for me attend the IPAM graduate summer school on "Intelligent Extraction of Information from High Dimensional Data". As a result of the experience the direction of my subsequent work turned towards dimensionality reduction.

The work in this thesis has been graciously supported by grants from the National Science Foundation (NSF), the National Aeronautics and Space Administration (NASA), and the Oak Ridge National Laboratory (ORNL).

Dedication

To my late father who was a Mathematician himself.

Abstract

As a result of recent technological advances, the availability of collected high dimensional data has exploded in various fields such as text mining, computational biology, health care and climate sciences. While modeling such data there are two problems that are frequently faced. High dimensional data is inherently difficult to deal with. The challenges associated with modeling high dimensional data are commonly referred to as the "curse of dimensionality." As the number of dimensions increases the number of data points necessary to learn a model increases exponentially. A second and even more difficult problem arises when the observed data exhibits intricate dependencies which cannot be neglected. The assumption that observations are independently and identically distributed (i.i.d.) is very widely used in Machine Learning and Data Mining. Moving away from this simplifying assumption with the goal to model more intricate dependencies is a challenge and the main focus of this thesis.

In dealing with high dimensional data, dimensionality reduction methods have proven very useful. Successful applications of non-probabilistic approaches include Anomaly Detection [1], Face Detection [2], Pose Estimation [3], and Clustering [4]. Probabilistic approaches have been used in domains such as Visualization [5], Image retrieval [6] and Topic Modeling [7]. When it comes to modeling intricate dependencies, the i.i.d. assumption is seldomly abandoned as in [8, 9]. As a result of the simplifying assumption relevant dependencies tend to be broken.

The goal of this work is to address the challenges of dealing with high dimensional data while capturing intricate dependencies in the context of predictive modeling. In particular we consider concepts from both non-probabilistic and probabilistic dimensionality reduction approaches.

From the perspective of non-probabilistic dimensionality reduction, we explore semi-supervised dimensionality reduction methods and their relationship to semi-supervised predictive methods. The predictive methods that we consider include label propagation and semi-supervised graph cuts. By introducing a uniform framework for graph-based semi-supervised learning we illustrate how both label propagation and semi-supervised graph cuts can be viewed as semi-supervised embedding. In addition to the gained

insights, a new family of label propagation methods is proposed based on existing dimensionality reduction methods. When it comes to capturing dependencies, non-probabilistic dimensionality reduction methods tend to utilize a graph Laplacian, a positive semi-definite matrix which captures relationships between data points only.

The main focus of the thesis is oriented towards concepts from probabilistic dimensionality reduction. In particular we examine to what extent ideas from dimensionality reduction can be used to create probabilistic multivariate models, capable of capturing covariance structure across multiple dimensions. Since most data naturally occurs in form of a matrix, modeling covariance structure within each dimension is relevant in a wide variety of applications. In essence this means moving away for the i.i.d. assumption. We propose models for multi-label classification, missing value prediction and topic modeling.

In multi-label classification each data point is associated with possibly multiple labels. Modeling covariances among labels in an effective and scalable manner is one of the major challenges. We propose Bayesian Multivariate Regression (BMR), as an efficient algorithm for multi-label classification, which takes into consideration covariances among labels. While it does not explicitly model covariances among data points, it captures similarities from the feature space by treating the mapping of from features to labels as an embedding. As illustrated by our empirical evaluations, BMR is a competitive to the state-of-the-art, at the same time it is scalable enough to be applied to very large data sets.

Making the i.i.d. assumptions in domains such as multi-label classification, missing value prediction or climate modeling can be expensive in terms of modeling accuracy. We consider the more general problem of modeling real-valued matrices and propose Probabilistic Matrix Addition (PMA), a novel model capable of modeling real-valued matrices of arbitrary size while capturing covariance structure across rows and across columns of the matrix simultaneously. Unlike approaches which vectorize matrices to model more intricate dependencies, PMA exhibits a very sparse dependency structure and has a generative model. As a result scalable inference is possible. We illustrate the effectiveness of the model in the domains of missing value prediction as well as multi-label classification.

Lastly we address the problem of modeling multiple covariance structures simultaneously in the domain of topic modeling. Topic models are typically implemented as hierarchical mixture models, whereby a document is represented by a multinomial distribution over topics, and topics in turn are assumed to be distributions over words. We propose Gaussian Process Topic Models (GPTM), the first topic model capable of incorporating a Kernel among documents while capturing covariances among data sets. This is accomplished by assuming a PMA prior over the latent document-topic matrix. Experiments on benchmark data sets show that the model is indeed effective.

Part of this work is also the development of an object-oriented machine learning toolbox for MATLAB. All algorithms in this thesis have been implemented in it. We briefly summarize its inner workings.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

While modeling data in most real-world applications there are two challenges that are frequently faced. Observations have the tendency to be high dimensional, making it difficult to infer accurate models from a limited number of observations. Another even bigger problem is the task of capturing intricate dependencies which might be present within the data. The assumption that observations are independently and identically distributed is very widely used. Moving away from this simplifying assumption with the goal to model more intricate dependencies is a challenge. The focus of this work is to address the issues of high dimensionality and capturing dependencies in the context of predictive modeling. Specifically, we consider the tasks of classification and missing value prediction.

1.1 Motivation

In a wide range of applications data tends to have a high dimensional representation. Images for instance are typically reshaped into long vectors, while text documents tend to be represented by high dimensional frequency count vectors. The ability to deal with high dimensional data effectively has become very important. In learning predictive models the amount of required training data increases exponentially with the number of feature dimensions. This is also referred to as the "curse of dimensionality." High dimensional data can also be problematic from a visualization perspective. In order to avoid serious problems, the ability to handle high-dimensional data effectively is crucial

in predictive modeling.

In addition to high dimensionality, an even more challenging issue that one faces is the modeling of potentially intricate dependencies within the data. Most existing approaches in Machine Learning and Data Mining make a simplifying assumption that observations or outputs are independently and identically distributed. While this assumption may lead to simpler models, it neglects potentially relevant relationships.

The domain of multi-label classification can be seen as an illustrative example. In multi-label classification every data point is assumed to be associated with possibly multiple labels. The goal is to obtain a predictive model for labels, given feature observations. In particular, consider the problem of protein function prediction. The input in this case would be possibly high-dimensional feature vectors representing proteins, indicating their physical structure. The output would be given by a binary vector reflecting which functions, out of a known set, are associated with the protein. The outputs, as in many applications, can be represented as a matrix, whereby each row represents a label vector. In this setting the assumption that label vectors (rows) are drawn i.i.d., translates to assuming that relationships between proteins can be neglected. Similarly assuming that individual labels (columns) are drawn i.i.d. leads to neglecting dependencies between functions. Most multi-label classification approaches will model dependencies in terms of a covariance structures either across data points or across labels. As it can be seen in the protein function prediction example, neglecting dependencies translates to discarding potentially relevant information. The main goal of this work is to examine models which are capable of capturing dependencies across multiple dimensions simultaneously. In the case of protein function prediction this means capturing the covariance structure across proteins (rows) and functions (columns) simultaneously. In other words covariances across inputs and outputs are modeled at the same time.

The applicability of models capable of capturing dependencies across multiple dimensions stretches far beyond multi-label classification. In particular such models in a general sense can be used to place priors over arbitrary real-valued matrices, whereby the covariance across rows and columns is modeled simultaneously. Applicability extends to any domains which involve the modeling of matrices. In particular climate modeling across the globe, or missing value prediction problems fall into this category. In missing value prediction frequently only a fraction of the values are known. Not

discarding any relevant structure in terms simplifying modeling assumptions is crucial.

In real-world applications the problems of dealing with high-dimensional data and capturing dependencies go hand in hand, since frequently observations tend to be provided in a high-dimensional space. We explore how dependencies can be captured while dealing with high dimensional data in predictive models.

1.2 Overview

Over the recent years a variety of dimensionality reduction methods have been proposed, both probabilistic and non-probabilistic to address the challenges associated with high dimensional data. We explore both types of methods in the context of predictive modeling. In particular we examine non-probabilistic approaches in the context of semi-supervised classification and graph cuts, whereby dependencies across data points are considered. We then explore probabilistic dimensionality reduction approaches for modeling more intricate dependency structures.

1.2.1 Dimensionality Reduction and Semi-Supervised learning

Non-probabilistic dimensionality reduction methods tend to be frequently used either to visualize or preprocess data. Label propagation is a family of graph-based semi-supervised learning methods, that has proven rather useful over the last decade. Its relationship to semi-supervised graph cuts is well known. However its interpretation as semi-supervised embedding has not been explored. In this thesis we examine the relationship between semi-supervised non-probabilistic dimensionality reduction, label propagation, and semi-supervised graph cuts. In particular by treating the mapping from features to labels, as an embedding, we show how label propagation as well as semi-supervised graph cuts can be understood as semi-supervised dimensionality reduction. In addition to providing valuable insights, we illustrate how existing embedding methods can be converted into label propagation algorithms. We propose a new family of label propagation methods derived from existing manifold embedding approaches [10]. When it comes to dependencies most graph-based semi-supervised methods rely on graph Laplacians, which a are positive semi-definite matrices capturing relationships among data points only.

1.2.2 Bayesian Multi-Variate Regression for Multi-Label Classification

Up until a few years ago the state-of-the art in multi-label classification was to assume that individual labels are drawn i.i.d. More recent approaches do take dependencies among labels into account, however the assumption that each label vector is drawn i.i.d. typically remains. Driven by the desire to address challenges in dealing with the data from the Aviation Safety Reporting System (ASRS), we propose Bayesian Multivariate Regression (BMR) [11], an efficient multi-label classification approach which models covariance structure across labels. While BMR does not explicitly model covariance structure across data points, it utilizes an embedding to map relationships from the higher dimensional feature space to the lower-dimensional label space. We illustrate the effectiveness of the model on a number of Benchmark data sets including ASRS.

1.2.3 Probabilistic Matrix Addition for Modeling Matrices

We propose Probabilistic Matrix Addition (PMA) [12], a novel model capable of modeling real-valued matrices of arbitrary size while capturing covariance structure across rows and across columns of the matrix simultaneously. PMA is based on Gaussian Processes (GP). It assumes a data matrix X to be composed as a sum of two matrices, where by the first matrix is drawn row-wise from a zero-mean Gaussian Process, and the second matrix is drawn column-wise from a second zero-mean Gaussian Process. The entries of the resulting data matrix X have dependencies across both rows and columns. PMA has a very sparse dependency structure and a generative model. As a result scalable inference can be devised and the model can naturally be extended to tensors. If we think of Gaussian Processes as modeling functions of the form $f(x)$, we can think of PMA as modeling functions of the form $f(x, y)$.

1.2.4 Capturing Multiple Covariance Structures in Topic Modeling

Topic models are typically implemented as hierarchical mixture models, whereby a document is represented by a multinomial distribution over topics, and topics in turn are assumed to be distributions over words. Since the output of topic models are lower

dimensional topic representations of documents, they also can be understood as performing dimensionality reduction. In the exiting literature one can find models which are capable of capturing covariances among topics. However there are no topic models capable of incorporating covariances among documents and among topics at the same time. We propose Gaussian Process Topic Models (GPTM) [13]. GPTM is the first topic model which can incorporate semi-supervised information among documents in form of a kernel while capturing covariance structure among topics. This is accomplished by imposing a PMA prior over the latent topic-document matrix. Using benchmark experiments we illustrate that the proposed model can indeed effectively utilize semi-supervised information in form of a kernel among documents to influence the resulting topics. All of this is accomplished without sacrificing the quality and interpret-ability of extracted topic distributions.

1.2.5 Machine Learning Toolbox for MATLAB

For purposes of conducting experimental evaluations in this thesis, we have developed MALT an object-oriented machine toolbox for MATLAB. Among other things it provides a unified user interface for all algorithms, the ability to auto-generate plots, a generic cross validation procedure as well as the ability to organize results, re-usable computations and data sets in a primitive database. All experiments and all plots in this work have been obtained using MALT.

1.3 Contributions of the Thesis

In summary, the contributions of this thesis revolve around using dimensionality reduction and capturing dependencies in predictive models. The focus is two-fold. For non-probabilistic dimensionality reduction we provide a unified view of label propagation, semi-supervised graph cuts and semi-supervised manifold embedding. As a result we propose a new family of label propagation methods, based on exiting manifold embedding methods. For probabilistic dimensionality reduction methods we note the inability of exiting approaches to model dependencies across multiple dimensions simultaneously. To address the problem we propose three models: (1) BMR, a simplistic approach for multi-label classification which takes the covariance among labels into consideration, (2)

PMA, a model for modeling arbitrary real-valued matrices, (3) GPTM, a topic model with PMA as prior. Lastly we provide an object-oriented machine learning toolbox (MALT) for conducting experiments.

The rest of this thesis is organized as follows. In Chapter 2 we describe the related work and background in the area of dimensionality reduction. In Chapter 3 we propose Bayesian Multivariate Regression, in chapter 4 we introduce Probabilistic Matrix Addition and in chapter 5 we describe Gaussian Process Topic Models. Finally we draw conclusions and describe future work in chapter 6.

Chapter 2

Literature Review

Since our work is based on dimensionality reduction, in this chapter we cover a number exiting approaches from literature, both non-probabilistic approaches as well as probabilistic ones.

2.1 Non-Probabilistic Methods

Non-probabilistic dimensionality reduction approaches typically formulate the embedding as an optimization problem, whereby the objective function characterizes what kind of properties are maintained in the lower-dimensional space. Over the last decade a large number of non-probabilistic dimensionality reduction methods have emerged, especially from the manifold embedding community. We briefly describe some of the best known methods, which can be considered as the state-of-the-art. For the purposes of the discussion let $X = \{x_1, \ldots, x_n\}$, where $x_i \in \mathbb{R}^d, i = 1, \ldots, n$, denote data points in the high dimensional space. The objective in non-probabilistic dimensionality reduction is to compute n corresponding data points $f_i \in \mathbb{R}^m$ where $m < d$. For simplicity and for the sake of the discussion we let $m = 1$.

Principal Component Analysis (PCA) One of the most well known dimensionality reduction approaches is Principal Component Analysis. In PCA the objective is to obtain an embedding while preserving as much of the variance from the original data set as possible (minimization of squared loss). PCA performs a linear projection, which is obtained using the top m eigenvectors of the data covariance matrix [14]. Due to

its linear nature, PCA is generally seen as not well suited for preserving non-linearities [15, 16, 17]. In recent years, PCA has been extended to work with exponential family distributions [18] and their corresponding Bregman divergences [19, 20].

Metric Multidimensional Scaling (MDS) Given a $n \times n$ dissimilarity matrix D and a distance measure, the goal of MDS is to perform dimensionality reduction in a way that will preserve dot products between data points as closely as possible [21]. We consider a particular form of MDS called classical scaling. In classical scaling, the Euclidean distance measure is used and the following objective function is minimized:

$$E_{MDS} = \sum_{i,j,i\neq j} (x_i^T x_j - f_i^T f_j)^2 = \sum_{i,j,i\neq j} D_{ij}^2 \ . \tag{2.1}$$

The first step of the method is to construct the Gram matrix XX^T from D. This can be accomplished by double-centering D^2 [22]:

$$x_i^T x_j = -\frac{1}{2}\left[D_{ij}^2 - D_{i\cdot}^2 - D_{\cdot j}^2 + D_{\cdot\cdot}^2\right] \ , \tag{2.2}$$

where

$$D_{i\cdot}^2 = \frac{1}{n}\sum_{a=1}^{n} D_{ia}^2,\ D_{\cdot j}^2 = \frac{1}{n}\sum_{b=1}^{n} D_{bj}^2,\ D_{\cdot\cdot}^2 = \frac{1}{n^2}\sum_{c=1}^{n}\sum_{d=1}^{n} D_{cd}^2 \ .$$

The minimizer of the objective function is computed from the spectral decomposition of the Gram matrix. Let V denote the matrix formed with the first m eigenvectors of $X^T X$ with corresponding eigenvalue matrix Λ that has positive diagonal entries $\{\lambda_i\}_{j=1}^m$. The projected data point in the lower dimensional space are the rows of $V\sqrt{\Lambda}$, i.e.,

$$\sqrt{\Lambda}V^T \ = [f_1 \ \ldots \ f_n].$$

The output of classical scaling maximizes the variance in the data set while reducing dimensionality. Distances that are far apart in the original data set will tend to be far apart in the projected data set. Since Euclidean distances are used, the output of the above algorithm is equivalent to the output of PCA [14, 23]. However, other variants of metric MDS are also possible where, for example, non-Euclidean distance measures or different objective functions are used.

Locally Linear Embedding (LLE): In LLE [15], the assumption is that each point in the high-dimensional space can be accurately approximated by a locally linear region. In particular, the neighborhood dependencies are estimated by solving $\min_W \sum_i ||x_i -$

$\sum_{j\in\mathcal{N}_i} w_{ij}x_j||^2$, such that $\sum_{j\in\mathcal{N}_i} w_{ij} = 1$, where $\mathcal{N}_i$ is the set of neighboring points of x_i. Then W is used to reconstruct the points in a lower-dimensional space by solving:

$$\min_{f\in\mathbb{R}} \sum_i ||f_i - \sum_j w_{ij} f_j||^2 \ , \quad s.t. \ \ f \perp \mathbb{1}, \ ||f||^2 = n \ . \tag{2.3}$$

While the computation of W is carried out locally, the reconstruction of the points is computed globally in one step. As a result, data points with overlapping neighborhoods are coupled. This way LLE can uncover global structure as well. The constraints on the optimization problems in the last two steps force the embedding to be scale and rotation invariant. LLE is a widely used method that has been successfully used on certain applications [24, 25] and has motivated several methods including supervised [15] and semi-supervised [26] extensions, as well as other embedding methods [27].

Laplacian Eigenmaps (LE): LE is based on the correspondence between the graph Laplacian and the Laplace Beltrami operator [28]. The symmetric weights between neighboring points are typically computed using the RBF kernel as $w_{ij} = \exp(-||x_i - x_j||^2/\sigma^2)$. Let D be a diagonal matrix with $D_{ii} = \sum_j w_{ij}$.

Then W is used to reconstruct the points in a lower-dimensional space by solving:

$$\min_f \ \frac{1}{2}\sum_{i,j} w_{ij}(f_i - f_j)^2 \ , \quad s.t. \ \ f \perp D\mathbb{1} \ , \ \ f^T D f = I \ . \tag{2.4}$$

Local Tangent Space Alignment (LTSA): In LTSA, the tangent space at each point is approximated using local neighborhoods and a global embedding is obtained by aligning the local tangent spaces. If X_i^N denotes the matrix of neighbors of x_i, then it can be shown [16] that the principal components of X_i^N give an approximation to the tangent space of the embedding f_i. Let $g_{i1}, \ldots, g_{ik}$ be the top k principal components for X_i^N. Let $G_i = [e/\sqrt{k}, g_{i1}, \ldots, g_{id}]^T$. If $\mathcal{N}_i$ are the indices of the neighbors of x_i, submatrices of the alignment matrix M are computed as $M(\mathcal{N}_i, \mathcal{N}_i) \leftarrow M(\mathcal{N}_i, \mathcal{N}_i) + I - G_i G_i^T$ for $i = 1, \ldots, n$. Finally, using M, which is guaranteed to be positive semidefinite, an embedding is subsequently obtained by minimizing the alignment cost:

$$\min_f \ \ f^T M f \ \ , \quad s.t. \ f \perp \mathbb{1} \ , \ \ ||f||^2 = n. \tag{2.5}$$

We refer the reader to [16] for a detailed analysis of LTSA.

Isometric Feature Mapping (ISOMAP): ISOMAP is another graph-based embedding method [29, 30]. The idea behind ISOMAP is to embed points by preserving

geodesic distances between data points. The method attempts to preserve the global structure in the data as closely as possible. Given a graph, geodesic distances are measured in terms of shortest paths between points. Once geodesic distances are computed MDS is used to obtain an embedding.

The algorithm consists of three steps. The first step is to construct a graph by computing k-nearest neighbors. In the second step, one computes pairwise distances D_{ij} between any two points. This can be done using Dijkstra's shortest path algorithm. The last step of ISOMAP is to run the metric MDS algorithm with D_{ij} as input. The resulting embedding will give $||f_i - f_j||^2$ approximately equal to D_{ij}^2 for any two points. By using a local neighborhood graph and geodesic distances, the ISOMAP method exploits both local and global information. In practice, this method works fairly well on a range of problems. One could prove [30] that as the density of data points is increased the graph distances converge to the geodesic distances. ISOMAP has been used in a wide variety of applications [31, 32], and has motivated several extensions in the recent past [17, 33, 26].

A methodology for converting non-linear dimensionality reduction methods to the semi-supervised setting was proposed in [26]. To the best of our knowledge none of the existing literature explores the relationship between dimensionality reduction, label propagation and semi-supervised graph-cuts.

2.2 Probabilistic Methods

One of the oldest probabilistic dimensionality reduction approaches comes from the field of Geostatistics and is known as the Linear Model of Corregionalization. More recent developments include the extension of Gaussian Processes to model dimensionality reduction or topic modeling. Topic modeling reduces high dimensional document representation into lower-dimensional mixtures of topics. Gaussian Processes and Kernel methods generally model covariances among data points. Some approaches in topic modeling on the other hand capture covariances among the embedded dimensions. Our work differs from most of the methods in this section in that it models covariances both

across data points and across the embedded dimensions simultaneously.

Gaussian Process Latent Variable Model (GPLVM): The Gaussian Process Latent Variable Model (GPLVM) [5] is one of the most well known probabilistic embedding methods. GPLVM assumes that points in the higher dimensional space were generated using a zero-mean Gaussian Process. The kernel is defined over the lower-dimensional points and can be chosen depending on application. Using a kernel which is based on an inner product matrix leads to Probabilistic Principal Component analysis. GPLVM can be thought as modeling covariances across data points in the form of a kernel. However it does not model covariances across the latent dimensions. GPLVM has been applied successfully in several domains including localization [34], pose estimation [35] and fault detection [36]. Semi-supervised [37] as well as discriminative variants have been proposed [38].

Latent Dirichlet Allocation (LDA): Latent Dirichlet allocation (LDA) [39] is one of the most widely used topic modeling algorithms. It is capable of extracting topics from documents in an unsupervised fashion. In LDA, each document is assumed to be a mixture of topics, whereby a topic is defined to be a distribution over words. LDA assumes that each word in a document is drawn from a topic z, which in turn is generated from a discrete distribution Discrete(π) over topics. Each document is assumed to have its own distribution Discrete(π), whereby all documents share a common Dirichlet prior α. The graphical model of LDA is in Figure 2.1, and the generative process for each document $\mathbf{w}$ is as follows:

1. Draw $\pi \sim \text{Dirichlet}(\alpha)$.
2. For each of m words $(w_j, [j]_1^m)$ in $\mathbf{w}$:
   (a) Draw a topic $z_j \sim \text{Discrete}(\pi)$.
   (b) Draw w_j from $p(w_j|\beta, z_j)$.

where $\beta = \{\beta_i, [i]_1^k\}$ is a collection of parameters for k topic distributions over totally V words in the dictionary. The generative process chooses β_i corresponding to z_j. The chosen topic distribution β_i is subsequently used to generate the word w_j. The most likely words in β_i are used as a representation for topic i.

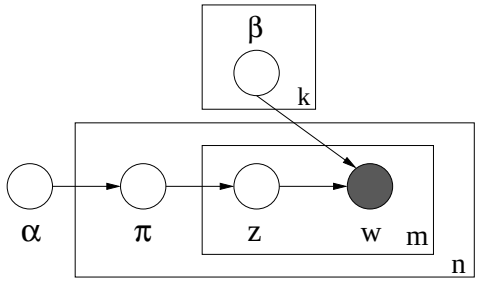


Figure 2.1: Graphical model for Latent Dirichlet Allocation.

Correlated Topic Model (CTM): The Correlated Topic Model [7] is an extension of Latent Dirichlet allocation to use a Gaussian prior. Rather than using a Dirichlet prior, the latent variable in CTM is drawn from a Gausian distribution. Unlike in LDA, the latent variable in CTM is real-valued. To obtain θ, a lognormal transformation is performed. The remainder of the generative model is identical to LDA. The CTM was an improvement over LDA in that it models the covariance across topics, leading to a higher quality model. The resulting covariance matrix can also be used to contract a "topic graph", visualizing how topics are related to each other. While CTM captures the covariance among topics, it does not have a natural way of incorporating semi-supervised information on the document level in terms of a kernel.

Linear Model of Corregionalization (LMC): Linear Models of Corregionalization (LMCs) are a broad family of related models widely studied in Geostatistics [8, 9]. LMCs were first introduced as a dimensionality reduction method. The simplest form of LMC, also known as the separable model or intrinsic specification [40, 9], works with vectors $X(s_j) \in \mathbb{R}^m$ at locations $s_j, j = 1, \ldots, n$. The objective is to capture associations within a given location and across locations. Following common notation from Geostatistics [9], let

$$X(s) = Aw(s), \tag{2.6}$$

be a process where $A \in \mathbb{R}^{m \times m}$ is a full rank matrix and $w_j(s) \sim N(0, 1)$ are i.i.d. processes with stationary correlation function $\rho(s - s') = corr(w_j(s), w_j(s'))$ not depending on j. $X(s)$ is assumed to have zero mean and variance 1. Let $T = AA^T \in \mathbb{R}^{m \times m}$ denote the local covariance matrix. The cross covariance $\Sigma_{X(s),X(s')}$ can then be expressed as

$$\Sigma_{X(s),X(s')} = C(s - s') = \rho(s - s')T \ . \tag{2.7}$$

Thus, by flattening out X as $vec(X) \in \mathbb{R}^{mn}$, the joint distribution of $vec(X) \sim N(0, \Sigma_{vec(X)})$ where $\Sigma_{vec(X)} = R \otimes T$, $R_{ss'} = \rho(s - s')$, and $\otimes$ denotes the Kronecker product. More general versions of LMC can be obtained by abandoning the i.i.d. assumption on $w_j(s)$ or by considering a nested covariance structure: [41, 9].

$$C(s - s') = \sum_u \rho_u(s - s')T^{(u)} \ . \tag{2.8}$$

Since the component processes are zero mean, the intrinsic formulation of LMC [9] only requires the specification of the second moment of the differences in measurements, given by

$$\begin{aligned} \Sigma_{X(s)-X(s')} &= \Psi(s - s') = C(0) - C(s - s') \\ &= T - \rho(s - s')T = \gamma(s - s')T \ . \end{aligned} \tag{2.9}$$

The function $\gamma(s - s') = \rho(0) - \rho(s - s')$, where $\rho(0) = 1$, is referred to as a variogram. Learning and inference in LMCs are typically performed by assuming a parametric form for the variogram [42, 8]. Several recent publications in machine learning [43, 44] can be seen as special cases of LMCs. Please note that LMC, is also considered to perform dimensionality reduction. It is the only existing approach which takes covariances across multiple dimensions into account. However this is accomplished by vectorizing the original matrix. As a result dependencies between every entry are modeled. The drawback of this approach is that it practically does not scale well. The Probabilistic Matrix Addition approach differs in that it assumes a sparse density structure. Further our proposed model does not consider the matrix in vectorized form.

Chapter 3

Embeddings, Graph Cuts, and Label Propagation

In this chapter we explore the relationships between dimensionality reduction, graph-based semi-supervised learning and semi-supervised graph-cuts.

3.1 Introduction

Semi-supervised learning is becoming a crucial part of data mining, since the gap between the total amount of data being collected in several problem domains and the amount of labeled data available for predictive modeling is ever increasing. Semi-supervised learning methods typically make assumptions about the problem based on which predictions are made on the unlabeled data [45]. A commonly used assumption, called the smoothness assumption, is that nearby points should have the same label. The assumption can be instantiated in several ways, and that has lead to several different algorithms for semi-supervised learning [46, 47, 48].

Graph-based semi-supervised learning algorithms are an instantiation of the smoothness assumption. In such a setting, a graph is constructed where each vertex corresponds to a point, and the edge connecting two vertices typically has a weight proportional to the proximity of the two points [46, 47]. Then, labels are "propagated" along the

weighted edges to get predictions on the unlabeled data. Recent years have seen significant interest in the design of label propagation algorithms for graph-based semi-supervised learning. While interpretations of label propagation methods in terms of random walks over graphs and semi-supervised graph cuts are not uncommon, the use of embedding methods for the purpose of label propagation has not been extensively explored. Further, empirical evaluation and comparison among the methods have been rather limited. To the extent that it is not quite clear how various existing methods compare to each other.

This chapter provides three major contributions. The first one is a unified view of label propagation, semi-supervised graph cuts and semi-supervised non-linear manifold embedding. A unification of label propagation in terms of a quadratic cost criterion was provided in [45]. It is no surprise that most existing approaches can be summarized using a common optimization framework. Unlike [45] the objective of our exposition is to explicitly draw out connections between various methods, semi-supervised graph cuts and semi-supervised embedding. This is not done in [45]. Another description of label propagation approaches is presented in [49]. In particular chapter 5 of [49] presents methods based on graph cuts, random walks and manifold regularization. These are presented as three different algorithms, without the connections between them being described. While we do not claim that our unification introduces previously unknown connections, to the best of our knowledge no existing literature draws out all of these connections explicitly. We believe that this exposition will serve as a good reference to those interested in future label propagation research. We present our unified framework by introducing a generic form of label propagation (GPL). We show how most existing label propagation approaches fit into this framework. We further show that both semi-supervised graph cuts and semi-supervised non-linear embeddings can also be described in terms of the same framework. Our unified framework has several advantages, including the ability to contrast exiting label propagation methods and to interpret them in terms of semi-supervised graph cuts and semi-supervised embedding.

Having drawn out the connections between semi-supervised non-linear embedding,

semi-supervised graph cuts and label propagation, the second contribution of this chapter is to explore existing non-linear embedding methods for the purposes of label propagation. In particular we provide a recipe for converting embedding methods to semi-supervised label propagation approaches. A novel aspect in this work is the direct application of semi-supervised embedding approaches to label propagation. In particular we introduce a label propagation algorithm based on Local Tangent Space Alignment, which we call LTSALP.

Finally, we present comprehensive empirical performance evaluation of the existing label propagation methods as well as the new ones derived from manifold embedding. Most existing publications on label propagation present very limited empirical evaluations. As a result it is not quite clear how state of the art approaches compare to each other. We performed extensive experiments. Among other things, we demonstrate that the new class of embedding-based label propagation methods is competitive on several datasets.

The rest of the chapter is organized as follows. We review background material in Section 3.2. In Section 3.3, we introduce the GLP formulation and present an unified view of existing label propagation methods. In Section 3.4, we show the relationship between semisupervised graph-cuts and the GLP formulation. We discuss semisupervised manifold embedding and introduce a set of embedding based label propagation methods in Section 3.5. We present empirical results in Section 3.6 and conclude in Section 4.5.

3.2 Background

In this section we review necessary background on graph-based semi-supervised learning and graph Laplacians.

3.2.1 Graph-based Semi-Supervised Learning

Let $D = \{(x_1, y_1), \ldots, (x_\ell, y_\ell), x_{\ell+1}, \ldots, x_{\ell+u}\}$ be partially labeled dataset for classification, where only ℓ out of the $n = (\ell + u)$ points have labels, and $y_i \in \{-1, +1\}$ for $i = 1, \ldots, \ell$.[1] Let $G = (V, E)$ be an undirected graph over the points, where each

[1] While we focus on the 2-class case for ease of exposition, the extensions to multi-class are mostly straightforward. We report results on multi-class problems in Section 3.6.

vertex v_i corresponds to a datapoint x_i, and each edge in E has a non-negative weight $w_{ij} \geq 0$. The weight w_{ij} typically reflects the similarity between x_i and x_j, and is assumed to be computed in a suitable application dependent manner. Given the partially labeled dataset D and the similarity graph G, our objective is to learn a function $f \in \mathbb{R}^n$, which associates each vertex to a discriminant score f_i and a final prediction sign(f_i) for classification. The problem has been extensively studied in the recent past [50, 48, 45, 46, 47].

3.2.2 Graph Laplacians

Let $G = (V, E)$ be an undirected weighted graph with weights $w_{ij} \geq 0$, and let D be a diagonal matrix with $D_{ii} = \sum_j w_{ij}$. In the existing literature, there are three related matrices that are called the graph Laplacian, and there does not appear to be a consensus on the nomenclature [51]. These three matrices are intimately related, and we will use all of them in our analysis. The *unnormalized graph Laplacian* L_u is defined as:

$$L_u = D - W \ . \tag{3.1}$$

The following property of the unnormalized graph Laplacian is important for our analysis: For any $f \in R^n$, we have

$$f^t L_u f = \frac{1}{2} \sum_{i,j} w_{ij} (f_i - f_j)^2 \ . \tag{3.2}$$

The matrix L_u is a symmetric and positive semidefinite. There are also two normalized graph Laplacians in the literature [52], respectively given by:

$$L_r = D^{-1} L_u = I - D^{-1} W \ , \tag{3.3}$$

$$L_s = D^{-1/2} L_u D^{-1/2} = I - D^{-1/2} W D^{-1/2} \ . \tag{3.4}$$

For the symmetrically normalized graph Laplacian, the following property holds: For any $f \in \mathbb{R}^n$, we have

$$f^t L_s f = \frac{1}{2} \sum_{i,j} w_{ij} \left(\frac{f_i}{\sqrt{D_{ii}}} - \frac{f_j}{\sqrt{D_{jj}}} \right)^2 \ . \tag{3.5}$$

We refer the reader to [53, 52, 51] for further details on Laplacians and their properties.

3.3 A Unified View of Label Propagation

In this section, we present a unified view of several label propagation formulations as a constrained optimization problem involving a quadratic form of the Laplacian where the constraints are obtained from the labeled data.

3.3.1 Generalized Label Propagation

The Generalized Label Propagation (GLP) formulation considers a graph-based semi-supervised learning setting as described in Section 3.2.1. Let W be the symmetric weight matrix and L be a corresponding graph Laplacian. Note that L may be any of the Laplacians discussed in Section 3.2, and we will see how different label propagation formulations result out of specific choices of the Laplacian. Let $f \in \mathbb{R}^n$, where $n = \ell + u$, be the predicted score on each data point x_i, $i = 1, \ldots, n$; the predicted label on x_i can be obtained as $\text{sign}(f_i)$. The generalized label propagation (GLP) problem can be formulated as follows:

$$\min_{f \in \mathcal{S}} f^T L f, \quad s.t. \ \sum_{i=1}^{\ell} (f_i - y_i)^2 \leq \epsilon \ , \tag{3.6}$$

where $\epsilon \geq 0$ is a constant and $\mathcal{S} \subseteq \mathbb{R}^n$. For most existing formulations $\mathcal{S} = \mathbb{R}^n$ whereas for a few $\mathcal{S} = \{f | f \in \mathbb{R}^n, f \perp \mathbb{1}\}$ where $\mathbb{1}$ is the all ones vector. The Lagrangian for the GLP problem is given by $L(f, \mu) = f^T L f + \mu \sum_{i=1}^{\ell} (f_i - y_i)^2$, where $\mu \geq 0$ is the Lagrangian multiplier. Some variants assume $y_i = 0$ for $i = (\ell + 1), \ldots, n$, so the constraint will be of the form $\sum_{i=1}^{n} (f_i - y_i)^2 \leq \epsilon$. Assuming the Laplacian to be symmetric, which is true for L_u and L_s, the first order necessary conditions are given by $(L + \mu I) f = \mu y$, where I is the identity matrix. Several existing methods work with the special case $\epsilon = 0$, which makes the constraints binding so that $\sum_{i=1}^{\ell} (f_i - y_i)^2 = 0$ and $f_i = y_i$ on the labeled points. The first order conditions for the special case is given by $Lf = 0$. In the next several sections, we show how most of the existing label propagation methods for semi-supervised learning can be derived directly as a special case of the GLP formulation or closely related to it with special case choices of the Laplacian L, the constant ϵ, and the subspace $\mathcal{S}$.

3.3.2 Gaussian Fields (GF)

Motivated by the assumption that neighboring points in a graph will have similar labels in Gaussian fields, the following energy function is considered [46]:

$$E(f) = \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2 \ . \tag{3.7}$$

The GF method computes labels by minimizing the energy function $E(f)$ with respect to f under the contraint that $f_i = y_i$ for all labeled points. As observed in [46], the energy function is harmonic, i.e., it is twice continuously differentiable and it satisfies Laplace's equation [54]. From the harmonic property of the energy function it follows that the predicted labels will satisfy: $f = D^{-1}Wf$. In terms block matrices corresponding to labeled and unlabeled points we have:

$$\begin{bmatrix} D_{\ell\ell} & 0 \\ 0 & D_{uu} \end{bmatrix} \begin{bmatrix} f_\ell \\ f_u \end{bmatrix} = \begin{bmatrix} W_{\ell\ell} & W_{\ell u} \\ W_{u\ell} & W_{uu} \end{bmatrix} \begin{bmatrix} f_\ell \\ f_u \end{bmatrix} \ .$$

Since $f_\ell = y_\ell$ due to the constraints,[2] the above system can be simplified to get a closed form for f_u given by

$$f_u = (D_{uu} - W_{uu})^{-1} W_{ul} y_l \ . \tag{3.8}$$

We can interpret the objective function in Gaussian Fields as a special case of the GLP problem in (3.6). In particular, using the identity in (3.2) and noting that the constraints on the labeled points are binding, GF can be seen as a special case of GLP with $L = L_u$ and $\epsilon = 0$, i.e.,

$$\min_{f\in\mathbb{R}^n} \ f^T L_u f \ , \quad s.t. \ \ \sum_{i=1}^{\ell}(f_i - y_i)^2 \leq 0 \ . \tag{3.9}$$

3.3.3 Tikhonov Regularization (TIKREG)

Given a partially labeled data set, TIKREG [55] is an algorithm for regularized regression on graphs, where the objective is to infer a function f over the graph. The objective function for TIKREG is given by

$$\min_{f} \ \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2 + \frac{1}{\gamma\ell}\sum_{i=1}^{\ell}(f_i - y_i)^2 \tag{3.10}$$

[2] We abuse notation and denote $[f_1, \ldots, f_\ell]^T$ by f_ℓ (similarly for y_ℓ) and $[f_{(\ell+1)}, \ldots, f_n]^T$ by f_u in the sequel.

with the constraint that $f \perp \mathbb{1}$, i.e., f lies in the orthogonal subspace of $\mathbb{1}$, the all ones vector. The parameter γ is assumed to be a real-valued number. A closed form solution for the above problem is obtained [55] as:

$$f = (\ell\gamma L_u + I_k)^{-1}(\hat{y} + \mu\mathbb{1}) \tag{3.11}$$

where $\hat{y} = (y_1, y_2, \ldots, y_l, 0, \ldots, 0)$, $I_k = diag(1, \ldots, 1, 0, \ldots, 0)$ with the number of ones equal to the number of labeled points. The orthogonality constraint on f is enforced through the lagrange multiplier μ, which is optimally computed as:

$$\mu = -\frac{\mathbb{1}^T(\ell\gamma L_u + I_k)^{-1}\hat{y}}{\mathbb{1}^T(\ell\gamma L_u + I_k)^{-1}\mathbb{1}} \ . \tag{3.12}$$

The objective function can be viewed as a special case of the GLP objective in (3.6). As before, the first term is $f^T L_u f$, where L_u is the unnormalized Laplacian. The second term corresponds to the constraint $\sum_i (f_i - y_i)^2 \leq \epsilon$, in (3.6) where $1/\gamma\ell$ is the optimal Lagrange multiplier corresponding to the constraint. In other words, if $\epsilon(1/\gamma\ell)$ is the constraint value that leads to the optimal Lagrange multiplier of $1/\gamma\ell$, the TIKREG problem can be seen as a special case of GLP:

$$\min_{f \in \mathbb{R}^n, f \perp \mathbb{1}} \ f^T L_u f \ , \quad s.t. \quad \sum_{i=1}^{\ell} (f_i - y_i)^2 \leq \epsilon(1/\ell\gamma) \ . \tag{3.13}$$

3.3.4 Local and Global Consistency (LGC)

The Local and Global Consistency (LGC) approach [47] gives an alternative graph based regularization framework for semi-supervised learning. In particular, the LGC is formulated based on the following objective function [47]:

$$\min_f \ \frac{1}{2}\left(\sum_{i,j=1}^{n} w_{ij} \left(\frac{1}{\sqrt{D_{ii}}} f_i - \frac{1}{\sqrt{D_{jj}}} f_j \right)^2 + \mu \sum_{i=1}^{n} (f_i - y_i)^2 \right) \ , \tag{3.14}$$

with $\mu > 0$ as the regularization parameter. Note that LGC assumes that there is a valid y_i for all points; operationally, the $y_i, i = 1, \ldots, \ell$ is set to the true given label, whereas $y_i, i = \ell + 1, \ldots, n$ is set to 0. The problem is solved using an iterative label propagation algorithm. Given a weight matrix W among the points, the weights are normalized to obtain $S = D^{-1/2} W D^{-1/2}$ with D diagonal and $D_{ii} = \sum_j w_{ij}$. Starting from an initial guess $f^{(0)}$, the iterative algorithm proceeds with the following updates:

$$f^{(t+1)} = \alpha S f^{(t)} + (1 - \alpha) y \ , \tag{3.15}$$

where $\alpha \in (0, 1)$. As shown in [47], this update equation converges to $f^* = (1 - \alpha)(I - \alpha S)^{-1} y$, which can be shown to optimize the objective function in (3.14) when $\alpha = 1/(1 + \mu)$. We now show that the LGC formulation is a special case of the GLP formulation in (3.6). From the identity involving the normalized Laplacian in (3.5), then the LGC can be seen as a special case of GLP as follows:

$$\min_f \; f^T L_s f \;, \quad s.t. \;\; \sum_{i=1}^{n} (f_i - y_i)^2 \leq \epsilon(\mu) \;, \tag{3.16}$$

where $\epsilon(\mu)$ is the constant corresponding to the optimal Lagrange multiplier μ. Note that since in LGC, one starts with an initial label $y_i, i = 1, \ldots, n$, the constraint involves terms corresponding to all the points.

3.3.5 Related Methods

We review three other methods, viz cluster kernels, Gaussian random walks, and local neighborhood propagation for graph-based semi-supervised learning which are closely related to the GLP framework.

Cluster Kernels (CK)

The main idea in cluster kernels [56] is to embed the data into a lower dimensional space based on its cluster structure and then subsequently build a classifier on the low-dimensional data. If K denotes a suitable kernel on the data space, the embedding method focuses on the k primary eigenvectors of the symmetrized matrix $D^{-1/2} K D^{-1/2}$. If K corresponds to the edge weights on the graph $G = (V, E)$ between the points, i.e., $K = W$, then the embedding corresponds to the k eigenvectors of the symmetrized Laplacian $L_s = I - D^{-1/2} W D^{-1/2}$ corresponding to the smallest k eigenvalues. In particular, for $k = 1$, the embedding is given by the eigenvector corresponding to the smallest eigenvalue of L_s which is the solution to LGC in absence of any semi-supervision. CK trains a suitable (linear) classifier on the low-dimensional embedding to obtain the final predictions.

Gaussian Random Walks EM (GWEM)

Consider a random walk on the graph with transition probability $P = D^{-1}W$. The GWEM method [50] works with the m-step transition probability matrix P^m so that the probability of going from x_i to x_j is given by $p_{m|0}(x_j|x_i) = (P^m)_{ij}$. The random walk is assumed to start with uniform probability from any one of the nodes, so $P(x_i) = 1/n$. Using Bayes rule, one can obtain the posterior probabilities $P_{0|m}(x_i|x_j)$. Now, each point is assumed to have a (possibly unknown) distribution $p(y|x_i)$ over the class labels. For any point x_j, the posterior probability of class label y is given by $P(y_j = c|x_j) = \sum_i P(y_i = c|x_i)p_{0|m}p(x_i|x_j)$. The prediction is based on $y_j = \text{argmax}_c \ P(y_j = c|x_j)$. Now, since $P(y|x_i)$ is unknown for the unlabeled points, an EM algorithm can be used to alternately maximize the log-posterior probability of known labels on the labeled points

$$\sum_{k=1}^{\ell} \log P(y_k|x_k) = \sum_{k=1}^{\ell} \log \sum_{i=1}^{N} P(y_i|x_i)P_{0|m}(x_i|x_k).$$

The EM algorithm alternates between the E step which estimates

$$P(x_i|x_k, y_k) \propto P(y_k|x_i)P_{0|m}(x_i|x_k) \tag{3.17}$$

where k denotes an index over labeled points, and the M step, which computes

$$P(y = c|x_i) = \frac{\sum_{k:y_k=c}^{\ell} P(x_i|x_k, y_k)}{\sum_{h=1}^{\ell} P(x_i|x_h, y_h)}$$

.

We now show that GWEM can be interpreted in terms of spectral decomposition of a suitable asymmetrically normalized Laplacian L_r as in (3.3). For a fixed number of steps m for the random walk, let $Z^T = P^m = (D^{-1}W)^m$. Note that Z^T itself is a transition probability matrix, and $Z_{ij} = P_{m|0}(x_i|x_j)$. Let D_Z be a diagonal matrix such that $D_{Z,ii} = \sum_j Z_{ij}$. Since the prior probability $P(x_i) = 1/n$, by Bayes rule we have

$$P_{0|m}(x_j|x_i) = \frac{P_{m|0}(x_i|x_j)}{\sum_{i'} P_{m|0}(x_{i'}|j)} = (D_Z^{-1}Z)_{ij}.$$

Let $f_j = P(y_j|x_j)$. When the EM algorithm converges we will have:

$$f = D_Z^{-1}Zf \Rightarrow (I - D_Z^{-1}Z)f = 0$$

, where $f_i = y_i$ for the labeled points. Since $D_Z^{-1}Z$ is a transition probability matrix, from (3.3) we note that $(I - D^{-1}Z)$ can be viewed as a asymmetrically normalized Laplacian L_r so that $L_r f = 0$. Finally, since $D_Z f = Zf$ resembles the fixed point equation for GFs, a block decomposition as in (3.8) yields $f_u = (D_{z,uu} - Z_{uu})^{-1} Z_{u\ell} y_\ell$.

Linear Neighborhood Propagation (LNP)

Linear Neighborhood Propagation (LNP) [48] is another recent approach, which differs from the other methods as LNP computes a stochastic transition matrix U directly from the data. In particular, one computes a probability distribution over neighboring points so that their expectation best approximates the point under consideration: $\min_{\mathbf{u}_i} \left\| x_i - X_i^N \mathbf{u}_i \right\|^2$, where $\mathbf{u}_i$ is probability distribution over the neighbors of x_i and X_i^N is a matrix each of whose columns is a neighbor of x_i. Once the transition probability matrix U is computed, the semisupervised learning problem is posed as follows:

$$\min_{f \in \mathbb{R}} \sum_{i,j} u_{ij}(f_i - f_j)^2 + \mu \sum_{i=1}^{n} (f_i - y_i)^2 \ , \tag{3.18}$$

where, similar to LGC [47], the labels $y_i, i = 1, \ldots, \ell$ are set to their true values, and the unknown labels $y_i, i = \ell+1, \ldots, n$ are set to 0. Similar to LGC, the LNP problem is solved by an iterative label propagation algorithm. Starting from an initial guess $f^{(0)}$, the iterative algorithm proceeds with the following updates:

$$f^{(t+1)} = \alpha U f^{(t)} + (1 - \alpha) y \ , \tag{3.19}$$

where $\alpha = 1/(1 + \mu) \in (0, 1)$. The updates are the same as in (3.15) for LGC [47] with the difference that U is not normalized symmetrically, but is a transition probability matrix of a random walk. In spite of the similarities, a careful consideration of the analysis in [48] reveals that update equation in (3.19) does not solve the problem in (3.18). On convergence, the iterative updates in (3.19) leads to $f = (I - \alpha U)^{-1}(1 - \alpha)y$. On the other hand, setting derivatives of (3.18) to zero leads to $f = \left(I - \alpha(U + U^T)/2\right)^{-1} (1 - \alpha)y$. The issue arises in the analysis [48] when one assumes $[(I - U) + (I - U)^T]f \approx 2(I - U)f$, which is not true unless U is symmetric. For empirical evaluation, we use the iterative updates in (3.19).

3.3.6 Label Propagation and Green's Function

We briefly describe an interesting relationship between label propagation and the discrete Green's function. Green's functions are typically used to convert inhomogenous partial differential equations with boundary conditions into an integral problem. In particular the inverse Laplace operator with the zero mode removed can be interpreted as a Green's function for the discrete Laplace operator [57]. Let $\mathcal{G} = L^{\dagger}$ be the generalized inverse of the Laplacian L. The solutions for both GF and GWEM can be expressed as: $f_u = (D_{uu} - W_{uu})^{-1} W_{ul} y_l = L_u^{\dagger} z_u$ where $z_u = W_{ul} y_l$. Discarding the zero mode of L_u, we have $f_u \approx \mathcal{G}_u z_u$. As argued in [57], discarding the zero mode is important to ensure that the Green's function exists; further, it does not affect the final result. Then f_u can be viewed as a solution to a partial differential equation with boundary value constraints. The interpretation is intuitive if the labeled points are treated as electric charges. In particular one assumes labeled points to be postive and negative charges. Using the Green's function one then computes the influence of these charges on unlabeled points [57]. For methods such as LGC and LNP the solution has the form $f = (I - A/(1+\mu))^{-1} \mu y/(1+\mu)$, with $A = D^{-1/2} W D^{-1/2}$ for LGC and $A = U$ for LNP. Considering the strong regularization limit as $\mu \to 0$ and removing the zero mode in L we obtain: $f = L^{\dagger} y \approx \mathcal{G} y$.

3.4 Semi-Supervised Graph Cuts

We now demonstrate how label propagation formulations can be viewed as solving a relaxed version of semi-supervised graph-cut problems. Let $G = (V, E)$ be a weighted undirected graph with weight matrix W. If V_1, V_2 is a partitioning of V, i.e., $V_1 \cap V_2 = \emptyset, V_1 \cup V_2 = V$, then the value of the cut implied by the partitioning (V_1, V_2) is given by: $cut(V_1, V_2) = \frac{1}{2} \sum_{v_i \in V_1, v_j \in V_2} w_{ij}$. The minimum cut problem is to find a partitioning (V_1, V_2) such that $cut(V_1, V_2)$ is minimized. Due to practical reasons, one often works with a normalized cut objective, such as the ratio-cut [58] or normalized-cut [59], which encourage the partitions V_1, V_2 to be more balanced. The objective for ratio-cut is $Rcut(V_1, V_2) = \frac{cut(V_1,V_2)}{|V_1|} + \frac{cut(V_2,V_1)}{|V_2|}$. The objective for normalized-cut is similar, however it normalizes cuts by the weight of the edges in each partition. Letting $Vol(V) = \sum_{i \in V} D_{ii}$, we have: $Ncut(V_1, V_2) = \frac{cut(V_1,V_2)}{Vol(V_1)} + \frac{cut(V_2,V_1)}{Vol(V_2)}$.

While the graph-cut problems outlined above are unsupervised, given labels on some of the nodes, one can construct a semi-supervised graph cut problem that respects the labeling [60, 61]. Let A_1 be the subset of vertices with label +1, and A_2 be the subset with label -1. Clearly, A_1 and A_2 are disjoint subsets of V. The *semi-supervised unnormalized cut* problem can be posed as follows: Find a partitioning (V_1, V_2) such that $cut(V_1, V_2)$ is minimized subject to the constraint $A_1 \subseteq V_1, A_2 \subseteq V_2$. In order to achieve balanced cuts, we also consider semi-supervised versions of the ratio-cut (or normalized-cut) problem. In particular, the *semi-supervised ratio-cut* problem can be posed as follows: Find a partitioning (V_1, V_2) such that $Rcut(V_1, V_2)$ is minimized subject to the constraint $A_1 \subseteq V_1, A_2 \subseteq V_2$. Similarly, one can pose the semi-supervised normalized-cut problem using $Ncut(V_1, V_2)$ instead of $Rcut(V_1, V_2)$ above. The problems outlined above are NP-hard, and there has been some work on developing polynomial-time approximation schemes (PTASs) for related problems [60, 61]. In this section, we show that relaxed versions of these problems lead to special cases of the GLP formulation for a suitable choice of the Laplacian L and the constraint ϵ, and hence can be solved using label propagation methods.

3.4.1 Semi-Supervised Unnormalized Cut

Consider a graph partitioning given by V_1 and V_2. Let f be defined as follows

$$f_i = \begin{cases} 1 & \text{if } v_i \in V_1 \\ -1 & \text{if } v_i \in V_2 \ . \end{cases} \tag{3.20}$$

From (3.2), we now have

$$f^t L_u f = \frac{1}{2} \sum_{i,j=1}^{n} \mathbf{w}_{ij} (f_i - f_j)^2 = 4cut(V_1, V_2) \ . \tag{3.21}$$

For any given disjoint sets A_1, A_2 which constitute the semi-supervision, we construct constraints on the labels as $y_i = +1$ if $v_i \in A_1$ and $y_i = -1$ if $v_i \in A_2$. Then, for all nodes in the labeled set, i.e., $v_i \in A_1 \cup A_2 = \mathcal{L}$, we have the constraint that $f_i = y_i$. Then, the semi-supervised unnormalized cut problem can be written as:

$$\min_{V_1, V_2} f^t L_u f, \quad s.t. \ f_i \text{ is as in (3.20)}, \ \ \forall v_i \in \mathcal{L}, f_i = y_i \ . \tag{3.22}$$

By relaxing the problem such that $f \in \mathbb{R}^n$ and noting that the constraint above is equivalent to $\sum_{i=1}^{\ell}(f_i - y_i)^2 \leq 0$, we obtain the following formulation:

$$\min_{f \in \mathbb{R}^n} f^t L_u f, \quad s.t. \sum_{i=1}^{\ell}(f_i - y_i)^2 \leq 0 \tag{3.23}$$

Clearly, the objective function is a special case of our GLP formulation using an unnormalized graph Laplacian and $\epsilon = 0$. In particular, the above is *exactly the same* as the formulation for Gaussian Fields [46] described in section (3.2).

3.4.2 Semi-Supervised Ratio Cut

In the context of the ratio cut problem, consider again a graph partitioning given by V_1 and V_2. Let f be defined as

$$f_i = \begin{cases} +\sqrt{|V_2|/|V_1|} & \text{if } v_i \in V_1 \\ -\sqrt{|V_1|/|V_2|} & \text{if } v_i \in V_2 \ . \end{cases} \tag{3.24}$$

Now, following (3.2), we can express

$$f^T L_u f = \frac{1}{2}\sum_{i,j} w_{ij}(f_i - f_j)^2 = |V| Rcut(V_1, V_2) \tag{3.25}$$

where $|V|$ is a constant. From the predefined values of f we can see that $f^T \mathbb{1} = 0$, and $||f||^2 = n$. The objective function for the semi-supervised ratio-cut problem can therefore be expressed as:

$$\min_{V_1, V_2} f^T L_u f, \quad s.t.\ f \perp \mathbb{1},\ ||f||^2 = n,\ f \text{ as in (3.24)},\ \forall v_i \in \mathcal{L},\ f_i = y_i \ . \tag{3.26}$$

We relax the problem and perform the optimization over $f \in \mathbb{R}^n$ such that $f \perp \mathbb{1}$. Note that in the unsupervised case, i.e., $\mathcal{L} = \emptyset$, the empty set, the solution to the problem is simply the second eigenvector of L corresponding to the second smallest eigenvalue. Now, relaxing the constraint[3] on $||f||$ and allowing f_i to mildly deviate from y_i on $v_i \in \mathcal{L}$, we get the following problem:

$$\min_{f \in \mathbb{R}^n} f^T L_u f, \quad s.t. \ \ f \perp \mathbb{1} \ , \quad \sum_{i=1}^{\ell}(f_i - y_i)^2 \leq \epsilon \ , \tag{3.27}$$

[3] Since the final prediction depends on sign(f_i), the norm constraint $||f||^2 = n$ does not have an effect on the accuracy.

which is exactly the problem TIKREG solves [55]. The key difference between the relaxed unnormalized formulation in (3.23) and the normalized formulation in (3.27) is the constraint $f \perp \mathbb{1} \Rightarrow \sum_i f_i = 1$, which ensures f lies in the subspace of $\mathbb{R}^n$ orthogonal to $\mathbb{1}$. The balancing constraint ensures the total score on positive predictions is the same as that on the negative predictions.

3.4.3 Semi-Supervised Normalized Cut

In the context of normalized cut, for a graph partitioning given by V_1 and V_2, let f be defined as follows

$$f_i = \begin{cases} \sqrt{vol(V_2)/vol(V_1)} & \text{if } v_i \in V_1 \\ -\sqrt{vol(V_1)/vol(V_2)} & \text{if } v_i \in V_2 \ . \end{cases} \tag{3.28}$$

Following an analysis similar to that of ratio cut, a semi-supervised normalized cut can be posed as the following optimization problem:

$$\min_{V_1,V_2} \ f^t L_u f, \ \ s.t.\ Df \perp \mathbb{1},\ f^T D f = vol(V)\ , \ \ f \text{ as in (3.28)}, \forall v_i \in \mathcal{L}, f_i = y_i\ . \tag{3.29}$$

First, we relax the problem and perform the optimization over $f \in \mathbb{R}^n$ such that $f \perp \mathbb{1}$. With $g = D^{1/2} f$, the relaxed problem is

$$\min_{g \in \mathbb{R}^n} \ g^T D^{-1/2} L_u D^{-1/2} g \ \ s.t.\ \ g \perp D^{1/2}\mathbb{1},\ \ ||g||^2 = vol(V),\ \forall v_i \in \mathcal{L}, g_i = D^{1/2} y_i. \tag{3.30}$$

Note that if $\mathcal{L} = \emptyset$, then the solution to the problem is simply the second eigenvector of the symmetrically normalized Laplacian $L_s = D^{-1/2} L_u D^{-1/2}$ corresponding to the second smallest eigenvalue. Now, relaxing the constraint on $||g||$ and allowing g_i to mildly deviate from $D^{1/2} y_i$ on $v_i \in \mathcal{L}$, we get the following problem:

$$\min_{g \in \mathbb{R}^n} \ g^T L_s g \ s.t.\ g \perp D^{1/2}\mathbb{1},\ \ \sum_{i=1}^{\ell} ||g_i - D^{1/2} y_i||^2 \leq \epsilon\ . \tag{3.31}$$

Algorithms for the above formulation have not been explored in the literature. The formulation is nearest to that of CK, but not the same since CK is a heterogeneous method which uses the normalized Laplacian for embedding, and then applies a classification algorithm on the embedding. It is also similar to LGC [47], although the constraint in LGC includes all points with $y_i = 0$ for $i = (\ell + 1), \ldots, n$ and does not involve the $D^{1/2}$ scaling on y_i in the constraint.

There has been notable attempts in the literature to directly solve some of the semi-supervised graph cut problems [60, 61]. Among such methods, the spectral graph transducer (SGT) [62] solves a problem closely related to the semi-supervised ratio cut problem, and reduces to TIKREG [55] under certain assumptions.

3.5 Semi-Supervised Embedding

In this section, we show how label propagation methods can be viewed as doing semi-supervised embedding. The geometric perspective helps in identifying relationships between existing embedding and label propagation methods, e.g., between Laplacian Eigenmaps [28] and Gaussian Fields [46]. More generally, we derive a new family of label propagation methods based on existing embedding methods, including Locally Linear Embedding (LLE) [63], Local Tangent Space Alignment (LTSA) [16] and Laplacian Eigenmaps (LE) [28]. While all such methods can be seen as a special case of the GLP formulation, they differ in the details—in particular, in the choice of the postive semi-definite matrix L and nature of constraints. Since our exposition is focussed on two class classification, the embedding will always be on $\mathbb{R}$, a one dimensional space.

3.5.1 Non-linear Manifold Embedding

Manifold embedding methods obtain a lower dimensional representation of a given dataset such that some suitable neighborhood structures are preserved. In this section we briefly review three popular embedding methods by formulating them in a similar form and demonstrate that their semi-supervised generalizations solve a variant of the GLP formulation.

Locally Linear Embedding (LLE):

Letting $M = (I - W)^T(I - W)$, which is positive semi-definite and can be viewed as an iterated Laplace operator [28], we can rewrite the objective function as:

$$\min_f \; f^T M f \quad , \quad s.t. \;\; f \perp \mathbb{1} \; , \; ||f||^2 = n \; . \tag{3.32}$$

Laplacian Eigenmaps (LE): Using (3.2), the objective function is $f^T L_u f$. Letting $g = D^{1/2} f$, with $M = L_s = D^{-1/2} L_u D^{-1/2}$ we can express the objective function as

$$\min_g \; g^T M g \; , \quad s.t. \;\; g \perp D^{1/2}\mathbb{1} \; , \;\; ||g||^2 = 1 \; . \tag{3.33}$$

Local Tangent Space Alignment (LTSA): In LTSA we have:

$$\min_f \ f^T M f \ , \quad s.t.\ f \perp \mathbb{1} \ , \ \ ||f||^2 = n. \tag{3.34}$$

3.5.2 Semi-Supervised Embedding

In this section, we consider two variants of semi-supervised embedding and its applications to label propagation. The variants differ in whether they consider the constraints associated with the corresponding unsupervised embedding problem. As discussed in Section 3.5.1, there are typically two types of constraints: $f \perp A\mathbb{1}$, where $A = I$ or $D^{1/2}$, and $||f||^2 = c$, a constant. Since the prediction is based on $\text{sign}(f_i)$, the norm constraint does not play any role in a classification setting, and will be ignored for our analysis. The two variants we consider are based on whether $f \perp A\mathbb{1}$ is enforced or not, in addition to the constraints coming from the partially labeled data.

Unconstrained Semi-Supervised Embedding

Following [64], we want to obtain an embedding $f = [f_\ell \ \ f_u]^T$, where the exact embeddings of the first ℓ points are known and given by y_ℓ.[4] The objective for semi-supervised embedding is given by

$$\min_f \ f^T M f \ , \quad s.t \ \ f_\ell = y_\ell \ , \tag{3.35}$$

where M is a suitable positive semi-definite matrix. Since f_ℓ is fixed, the problem can be cast in terms of block matrices as:

$$\min_{f_u} \begin{bmatrix} f_\ell^T & f_u^T \end{bmatrix} \begin{bmatrix} M_{\ell\ell} & M_{\ell u} \\ M_{u\ell} & M_{uu} \end{bmatrix} \begin{bmatrix} f_\ell \\ f_u \end{bmatrix} \ , \tag{3.36}$$

Setting the first derivative to zero one obtains:

$$f_u = -M_{uu}^{-1} M_{u\ell} y_\ell \ . \tag{3.37}$$

In the context of label propagation for a 2-class classification setting, we will have $y_i = +1$ or $y_i = -1$ for $i = 1, \ldots, \ell$. In other words, *the labeled points are being embedded to its true class label, and the rest will be embedded while trying to maintain*

[4] While the constraints can be relaxed to consider $\sum_{i=1}^{\ell}(f_i - y_i)^2 \leq \epsilon$, we do not focus on the general case here.

the neighborhood structure. For LLE, $M = (I - W)^T(I - W)$ and we call the corresponding label propagation algorithm LLELP. Similarly, for LTSA, M is as discussed in Section 3.5.1, and the corresponding algorithm will be called LTSALP. For unconstrained LE from (2.4), $M = L_u$, and the corresponding algorithm will be called LELP. For LELP, since $M = L_u$, the unnormalized Laplacian, from (3.37) we have

$$\begin{aligned} f_u &= -L_{uu}^{-1}L_{u\ell}y_\ell = -(D_{uu} - W_{uu})^{-1}(D_{u\ell} - W_{u\ell})y_\ell \\ &= (D_{u\ell} - W_{u\ell})^{-1}W_{u\ell}y_\ell \ , \end{aligned}$$

since $D_{u\ell} = 0$ as D is a diagonal matrix. We note that the solution is exactly the same as that for GF as in (3.8) implying *the equivalence of GF and LELP*.

Constrained Semi-Supervised Embedding

In this section, we consider embedding problems when the orthogonality constraint of the form $f \perp A\mathbb{1}$ is enforced. In particular, we consider the following problem

$$\min_f \ f^T M f \ \ , \quad s.t. \ \ \sum_i^\ell (f_i - y_i)^2 \leq \epsilon \ , \ \ f \perp A\mathbb{1} \ , \tag{3.38}$$

where $A = I$ for LLE and LTSA, and $A = D^{1/2}$ for LE. Let α and μ be the Lagrange multipliers for the two constraints respectively. The first order necessary conditions obtained from the Lagrangian corresponding to (3.38) yields

$$f = (M + \alpha I_k)^{-1}(\alpha y + \mu A\mathbb{1}/2) \tag{3.39}$$

Since $\mathbb{1}^T A^T f = 0$, a direct calculation gives the optimal Lagrange multiplier as

$$\mu = -2\alpha \frac{\mathbb{1}^T A^T (M + \alpha I_k)^{-1} y}{\mathbb{1}^T A^T (M + \alpha I_k)^{-1} A\mathbb{1}} \ . \tag{3.40}$$

For LLE, $M = (I - W)^T(I - W)$ and $A = I$, and we call the corresponding algorithm LLELPC. For LTSA, M is as discussed in Section 3.5.1 and $A = I$, and we call the corresponding algorithm LTSALPC. For LE as in (3.33), $M = L_{sym} = I - D^{1/2}WD^{1/2}$ and $A = D^{1/2}$ and we call the corresponding algorithm LELPC.

3.6 Experiments

In this section we provide an empirical evaluation of label propagation methods discussed in this chapter. To our knowledge, this is the most comprehensive empirical comparison of the methods till date. Our experiments are divided into two parts: First we compare seven methods on 14 benchmark data sets in terms of their accuracy; next, we take a closer look at the performance of the new label propagation methods obtained from the perspective of semi-supervised manifold embedding. For the first set of experiments, the methods we consider include 6 standard methods: GF, LNP, CK, GWEM, TIKREG and LGC, as discussed in Section 3.3. In addition, we include LTSALP, the novel approach based on semi-supervised LTSA embedding. For the second set of experiments, the methods we consider are the 6 embedding based methods introduced in Section 3.5.

Methodology: We conducted our experiments on 14 well known benchmark data sets. They include the following 8 UCI data sets: Hepatitis, Cancer, Pima, Wine, Iris, Glass, USPS (1-4 only), and Letter (E and F only). In addition we also ran experiments on 6 text datasets, which are all subsets of the 20Newsgroup data set: Different100, Similar100, Same100, Different1000 and Same1000 [65]. Each dataset contains a subset of 3 newsgroups with varying degree of difficulty for classification. Different100 (1000) includes alt.atheism, rec.sport.baseball, and sci.space, which are easy to separate; Same100 (1000) includes comp.graphics, comp.os.ms-windows, comp.windows.x, which are difficult to separate; whereas Similar100 includes talk.politics.guns, talk. politics.mideast, talk.politics.misc, which are moderately difficult to separate. We also used the well known Classic-3 benchmark data set containing 3893 documents, whereby 1033 are from medical journals, 1400 are aeronautical system papers and 1460 are information retrieval papers. For each method and each data set we ran five-fold semi-supervised cross validation. Specifically, the training points were chosen from four folds and the test error was measured on the fifth fold. All points were used to construct the neighborhood graph. Further, for each approach involving parameters, e.g., CK using SVMs with RBF kernel, parameter values were selected by cross-validation.

3.6.1 Experimental Results

The performance of the 6 methods from the literature as well as the novel LTSALP in shown in Figures 3.4, 3.5 and 3.6. All results reported are on the test set. To avoid clutter, we display the results for the top five methods based on average test-set error in Figures 3.1 and 3.2 . We make the following observations based on the results:

- There is no dominating method across all datasets. However, there is a set of methods that seem to be fairly consistently among the top few, and the performance of the top few methods is typically close. While across the top methods the differences typically don't appear significant, when compared to the worst methods the improvements do tend to be significant. For instance, if we examine the results in Figure 4 for the USPS data set we can see that the performance for the top methods is around $1.5 - 1.8\%$, while the performance for the worst methods is between 3.0% and 7.6%. The improvements of the best methods are clearly significant in comparison to the worst judging by the standard deviation in error.

- Our results also indicate that no given method is always among the best methods for each data set. For any method we could find at least one data set where its performance is among the worst. It appears that the assumptions made by various approaches do not work well across all manifold structures encountered in various data sets. From an embedding perspective this makes sense, since different embedding approaches tend to capture slightly different aspects of a manifold.

- The new method LTSALP is among the top performing methods in most of the UCI datasets. However, it performs poorly on the text datasets, possibly indicating that the idea of aligning the tangent spaces may have to be suitably modified for sparse high-dimensional datasets.

- TIKREG is among the most consistent methods on the text datasets. However its performance is not as consistent on the UCI data sets.

- In spite of an issue with its formulation, LNP is found to be quite competitive across several UCI and text datasets. Its parameter insensitivity make it rather easy to tune.

- When CK does well, it outperforms all the other methods. However, it does not have a consistent performance, which probably can be addressed by more thorough cross-validation over its parameter choices.

- GF seems to be a slow starter, not performing well for small number of labeled points, but improving significantly as the number of labeled points increases. This can particularly be seen on the text data sets. As more labeled points are considered GF becomes the best performing method on several text data sets. Generally GF is among the consistently well-performing methods.

- GWEM does not demonstrate a consistent performance and seems quite sensitive to the predefined number of steps in the random walk.

In summary, while several of the methods perform well on some datasets, the embedding-based proposed method LTSALP seems to be either the best or among the best methods for 6 out of 8 UCI data sets. Further, GF and TIKREG also stand out as quite consistent in terms of their performance, GF on both the UCI and text datasets, and TIKREG mostly on the text data sets.

3.6.2 Semi-Supervised Embedding Methods

We now compare the six label propagation methods based on semi-supervised manifold embedding. In particular we examine both variants of Laplacian Eigenmaps based Label Propagation (LELP, LELPC), Locally Linear Embedding based Label Propagation (LLELP, LLELPC), and Local Tangent Space Alignment based Label Propagation (LTSALP, LTSALPC). Note that LTSALP has already been compared to the existing methods in Figures 3.1, 3.2, 3.4, 3.5 and 3.6 to serve as a reference. We compare the methods on the UCI datasets, and show representative plots in Figure 3.3.[5] Based on our experiments, we observe that LELP and LTSALP performed most consistently well, while LLELP did well only on specific data sets such as Wine. The performance of LLELP seems to be more sensitive to the geometry of a data set. The effect of the orthogonality constraint on these methods can be understood when comparing the restults for Cancer and Hepatitis in Figure 3.3. While the performance is clearly affected by the

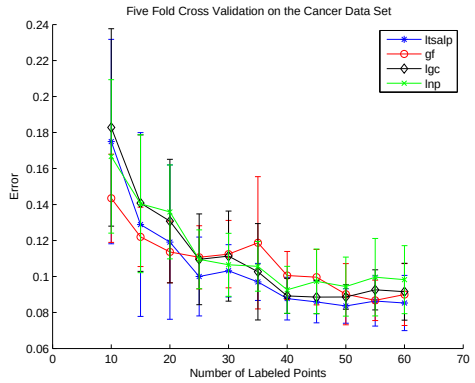
[5] We report results on 2-class problems in Figure 3.3. Since Wine is a 3-class dataset, we constructed a 2-class subset Wine(2) for these experiments.

constraints, it does not necessarily result in improved performance. For example, the constraints lead to better performance in Hepatitis but worse performance in Cancer.

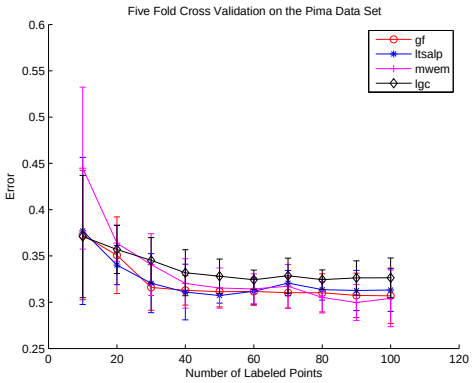
For the embedding methods, the quality of semi-supervised label propagation seems to depend on how well the unsupervised embedding preserves the class separation. Figure 3.8 illustrates the difference between unsupervised embedding and semi-supervised embedding on Wine. Note that the unsupervised embedding obtained from LLE and LTSA maintains the class separation better than LE for this particular dataset. When semisupervision is added, the embedding obtained from all the methods changes suitably. The class separation is most clear in LLELP followed by LTSALP and LELP, a fact reflected in the test-set error rates in Figure 3.5 (right panel). In general, label propagation based on a semi-supervised embedding method works well if the geometric structure of the class labels is well aligned with the biases of the embedding method.

3.7 Conclusions

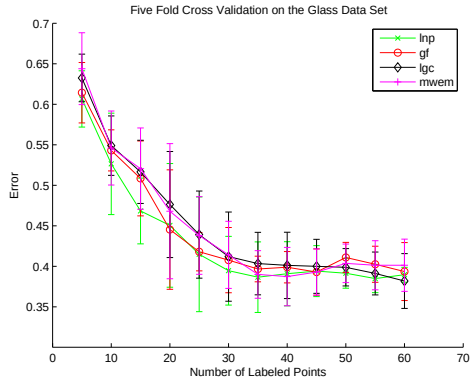
In this chapter, we have developed a unified perspective to a large set of graph-based semi-supervised learning methods. Using our framework we have showed that semi-supervised approaches to graph-cuts or manifold embeddings lead to the same computational problem. We have provided a recipe for converting embedding methods to label propagation algorithms. While the individual theoretical connections are known among experts in the community, unlike existing literature, we uniformly present label propagation from the perspective of graph cuts and embedding methods. In particular we provide interpretations of existing approaches both in terms of graph cuts and semi-supervised embedding methods. Our extensive empirical evaluation reveals that while there are no clear winners in terms of performance, certain methods seem consistent across several datasets, including some of the new embedding based methods introduced in this chapter. Our analysis improves our theoretical understanding of an important class of methods in semi-supervised learning, introduces several new methods to solve the problem, and, to our knowledge, provides the first comprehensive empirical performance evaluation of this family of models. The competitive performance of the embedding-based label propagation methods such as LTSALP provides strong incentive to investigate the embedding-based family of methods further.



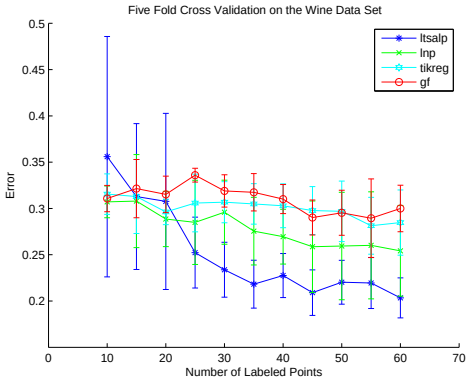
(a) Cancer



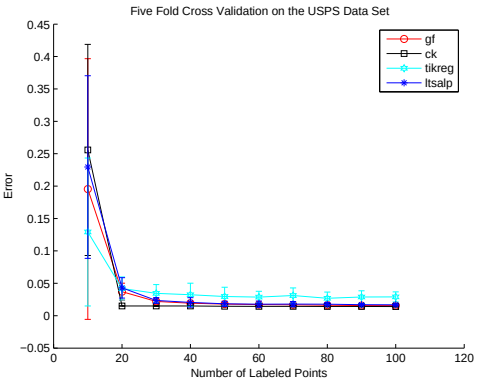
(b) Pima



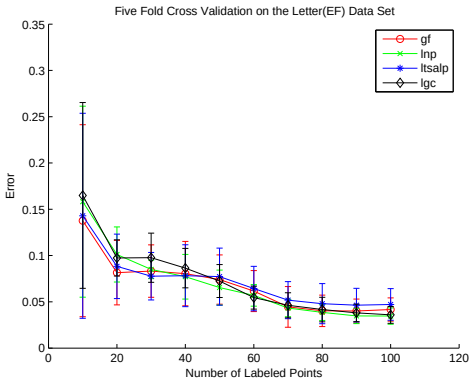
(c) Glass



(d) Wine

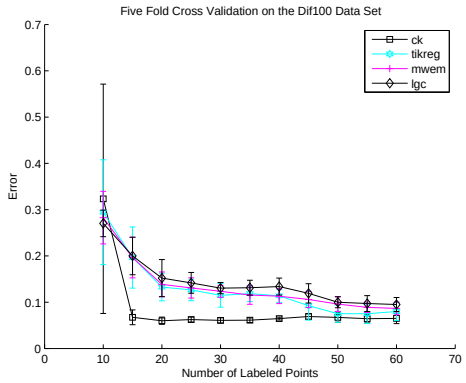


(e) USPS

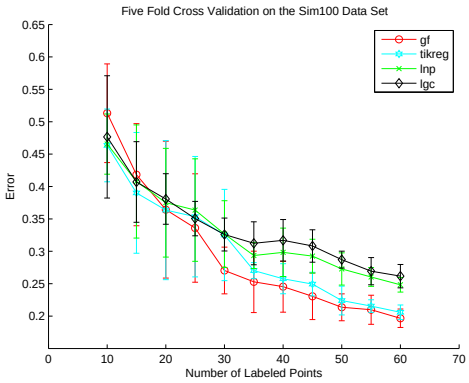


(f) Letter

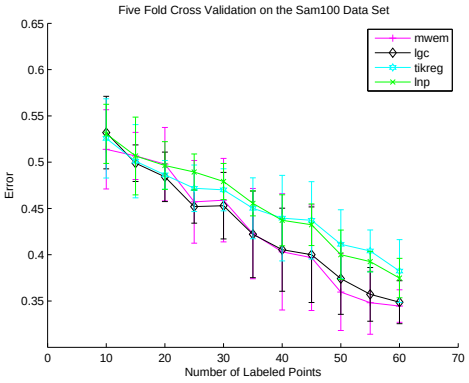
Figure 3.1: Five-fold cross validation as increasingly many points are labeled.



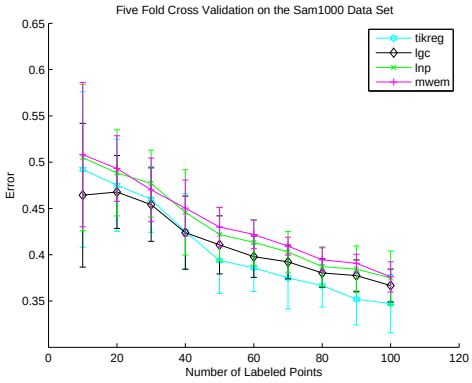
(a) Dif100



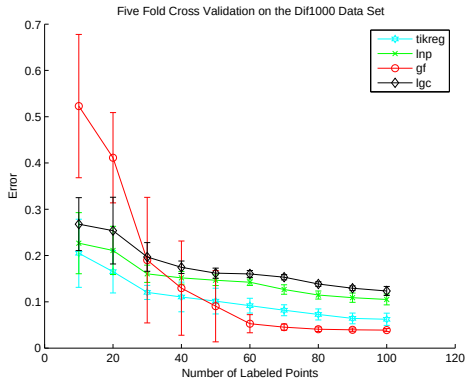
(b) Sim100



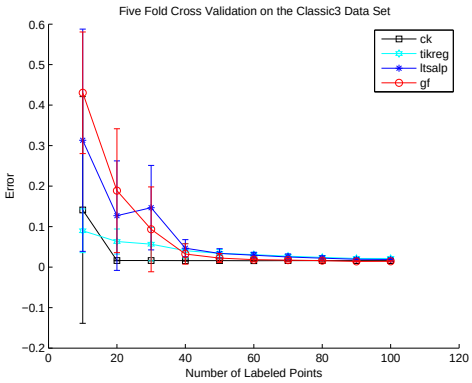
(c) Sam100



(d) Sam1000

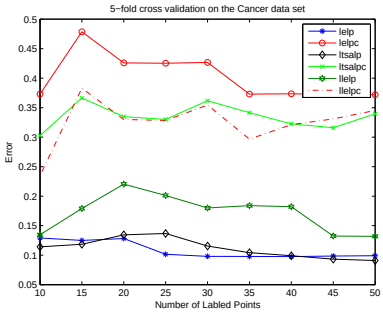


(e) Dif1000

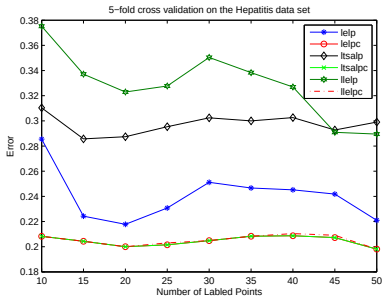


(f) Classic3

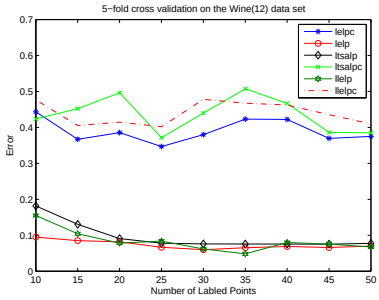
Figure 3.2: Five-fold cross validation as increasingly many points are labeled.



(a) Cancer



(b) Hepatitis



(c) Wine(2)

Figure 3.3: Comparison of embedding based label propagation methods: LELP(C), LTSALP(C), LLELP(C).

	gf	mwem	lnp	lgc	tikreg	ck	ltsalp
Hepatitis	21.1 ± 0.4	21.4 ± 0.7	25.8 ± 3.0	22.7 ± 1.2	**20.8 ± 0.0**	20.8 ± 0.0	24.3 ± 2.5
Cancer	11.2 ± 1.9	12.7 ± 2.5	10.6 ± 1.7	11.1 ± 2.5	10.8 ± 1.9	11.3 ± 1.3	**10.3 ± 1.5**
Wine	31.9 ± 1.7	32.2 ± 0.9	29.6 ± 3.5	32.4 ± 2.1	30.7 ± 2.2	30.1 ± 2.2	**23.4 ± 3.0**
Iris	5.5 ± 2.5	**4.8 ± 4.1**	6.0 ± 2.5	5.3 ± 3.8	5.7 ± 3.4	9.2 ± 1.0	5.3 ± 1.9
Glass	40.8 ± 4.0	41.4 ± 4.1	**39.5 ± 4.2**	41.2 ± 5.5	42.8 ± 5.5	49.6 ± 4.5	45.3 ± 4.9
Pima	**31.6 ± 2.5**	34.1 ± 3.3	36.0 ± 3.0	34.5 ± 2.5	33.2 ± 2.1	34.8 ± 0.2	32.1 ± 3.2
USPS	2.2 ± 0.4	13.0 ± 2.1	6.3 ± 0.9	4.9 ± 0.7	3.4 ± 1.4	**1.5 ± 0.1**	2.4 ± 0.4
Letter(EF)	8.3 ± 2.8	15.4 ± 2.8	8.5 ± 1.0	9.8 ± 2.7	8.3 ± 1.7	48.5 ± 0.5	**7.8 ± 2.6**
Dif100	7.3 ± 1.4	12.4 ± 1.3	13.6 ± 2.4	13.0 ± 1.2	11.5 ± 2.6	**6.1 ± 0.6**	12.7 ± 2.2
Sim100	**27.0 ± 3.6**	34.3 ± 5.3	32.7 ± 5.1	32.6 ± 2.5	32.5 ± 7.0	63.4 ± 2.1	41.0 ± 5.8
Sam100	48.8 ± 8.5	45.9 ± 4.5	47.9 ± 1.9	**45.3 ± 3.6**	47.0 ± 2.3	65.7 ± 2.0	51.7 ± 7.6
Dif1000	19.0 ± 13.6	23.3 ± 2.6	16.0 ± 1.9	19.7 ± 3.1	**12.0 ± 1.5**	65.8 ± 2.5	28.2 ± 14.6
Sam1000	54.9 ± 8.9	47.0 ± 3.4	47.7 ± 3.6	**45.4 ± 4.0**	46.0 ± 3.6	66.4 ± 1.4	61.0 ± 4.7
Classic3	9.3 ± 10.5	12.1 ± 1.7	29.2 ± 1.1	28.8 ± 0.7	5.7 ± 4.1	**1.6 ± 0.1**	14.7 ± 10.4

Figure 3.4: Performance comparisons with 30 labeled points on 14 datasets and 7 methods.

	gf	mwem	lnp	lgc	tikreg	ck	ltsalp
Hepatitis	21.4 ± 1.7	21.0 ± 1.0	23.8 ± 3.0	22.3 ± 1.8	**20.9 ± 0.6**	20.9 ± 0.6	25.4 ± 1.4
Cancer	10.1 ± 1.3	9.5 ± 0.9	9.3 ± 1.3	8.9 ± 1.0	10.6 ± 2.3	12.1 ± 2.3	**8.8 ± 1.2**
Wine	31.0 ± 1.6	31.4 ± 1.0	27.0 ± 3.0	31.2 ± 1.5	30.3 ± 2.4	30.1 ± 1.8	**22.8 ± 2.4**
Iris	**2.5 ± 2.0**	3.1 ± 0.5	6.4 ± 2.7	3.6 ± 1.9	4.0 ± 2.1	9.6 ± 1.4	4.4 ± 2.3
Glass	39.9 ± 1.9	**38.7 ± 3.6**	39.1 ± 4.0	40.1 ± 4.1	42.4 ± 2.1	44.4 ± 3.7	41.3 ± 3.8
Pima	31.3 ± 1.6	32.0 ± 2.7	35.0 ± 1.4	33.2 ± 2.5	33.7 ± 2.9	34.8 ± 0.2	**31.1 ± 3.0**
USPS	1.9 ± 0.3	9.0 ± 2.1	5.4 ± 1.5	4.0 ± 0.8	3.2 ± 1.8	**1.5 ± 0.1**	2.1 ± 0.7
Letter(EF)	8.0 ± 3.5	12.7 ± 2.6	**7.7 ± 2.4**	8.6 ± 2.1	7.9 ± 1.5	48.8 ± 0.6	7.8 ± 3.3
Dif100	6.8 ± 1.1	11.3 ± 1.6	13.5 ± 1.7	13.4 ± 1.8	11.4 ± 1.4	**6.5 ± 0.3**	11.8 ± 2.3
Sim100	**24.5 ± 3.9**	31.6 ± 4.5	29.8 ± 3.7	31.7 ± 3.2	25.8 ± 2.3	60.2 ± 3.0	37.8 ± 6.6
Sam100	41.6 ± 6.0	**40.3 ± 6.3**	43.7 ± 2.8	40.5 ± 4.5	44.0 ± 4.6	66.0 ± 2.9	44.3 ± 5.4
Dif1000	13.0 ± 10.2	20.8 ± 1.8	15.2 ± 1.4	17.5 ± 1.3	**11.1 ± 3.2**	65.7 ± 2.6	19.6 ± 8.1
Sam1000	48.7 ± 10.9	45.0 ± 3.0	44.6 ± 4.6	**42.4 ± 4.0**	42.5 ± 4.1	65.9 ± 1.0	55.4 ± 9.0
Classic3	3.2 ± 2.5	9.2 ± 0.5	28.5 ± 0.2	28.5 ± 0.3	4.1 ± 1.8	**1.6 ± 0.1**	4.7 ± 2.1

Figure 3.5: Performance comparisons with 40 labeled points on 14 datasets and 7 methods.

	gf	mwem	lnp	lgc	tikreg	ck	ltsalp
Hepatitis	22.1 ± 1.0	$\mathbf{21.7 \pm 1.2}$	25.0 ± 2.2	22.1 ± 1.6	21.7 ± 1.2	21.7 ± 1.2	24.8 ± 2.1
Cancer	9.0 ± 1.7	9.0 ± 0.7	9.4 ± 1.6	8.9 ± 0.7	11.3 ± 2.7	10.6 ± 1.8	$\mathbf{8.4 \pm 1.0}$
Wine	29.5 ± 2.4	29.1 ± 1.4	25.9 ± 5.8	30.2 ± 2.6	29.7 ± 3.3	28.0 ± 1.9	$\mathbf{22.0 \pm 2.4}$
Iris	$\mathbf{3.2 \pm 1.1}$	3.8 ± 0.4	6.6 ± 2.9	4.4 ± 1.1	4.2 ± 1.3	9.2 ± 2.3	3.2 ± 1.1
Glass	41.1 ± 1.9	40.4 ± 2.4	$\mathbf{39.1 \pm 1.8}$	39.9 ± 2.3	42.0 ± 2.8	42.9 ± 2.8	42.1 ± 4.4
Pima	31.1 ± 1.6	31.5 ± 2.2	35.3 ± 2.4	32.8 ± 1.9	33.7 ± 2.7	34.8 ± 0.3	$\mathbf{30.7 \pm 0.8}$
USPS	1.8 ± 0.2	7.6 ± 1.6	4.6 ± 1.2	3.6 ± 0.4	3.0 ± 1.4	$\mathbf{1.5 \pm 0.1}$	1.8 ± 0.4
Letter(EF)	7.4 ± 2.7	10.8 ± 3.3	6.5 ± 1.9	7.2 ± 1.8	$\mathbf{6.2 \pm 2.8}$	48.8 ± 0.6	7.7 ± 3.1
Dif100	6.9 ± 1.1	9.6 ± 1.6	10.7 ± 2.3	10.0 ± 1.2	7.5 ± 1.9	$\mathbf{6.7 \pm 1.0}$	10.2 ± 1.3
Sim100	$\mathbf{21.4 \pm 2.1}$	28.5 ± 3.1	27.3 ± 2.5	28.7 ± 1.3	22.4 ± 2.2	62.6 ± 3.0	30.6 ± 2.6
Sam100	$\mathbf{35.6 \pm 3.6}$	36.0 ± 4.2	40.0 ± 2.7	37.4 ± 3.9	41.1 ± 3.7	65.4 ± 2.0	39.6 ± 5.4
Dif1000	$\mathbf{9.1 \pm 7.7}$	18.9 ± 1.8	14.7 ± 1.2	16.2 ± 1.1	10.2 ± 2.8	63.5 ± 2.5	18.0 ± 6.6
Sam1000	43.0 ± 9.9	43.0 ± 2.1	42.2 ± 2.9	41.1 ± 3.1	$\mathbf{39.4 \pm 3.6}$	65.3 ± 1.4	55.6 ± 5.8
Classic3	2.2 ± 1.1	7.6 ± 1.5	28.5 ± 0.1	28.5 ± 0.4	3.4 ± 1.0	$\mathbf{1.6 \pm 0.1}$	3.4 ± 1.2

Figure 3.6: Performance comparisons with 50 labeled points on 14 datasets and 7 methods. LTSALP performs well on UCI datasets, GF performs well on text datasets.

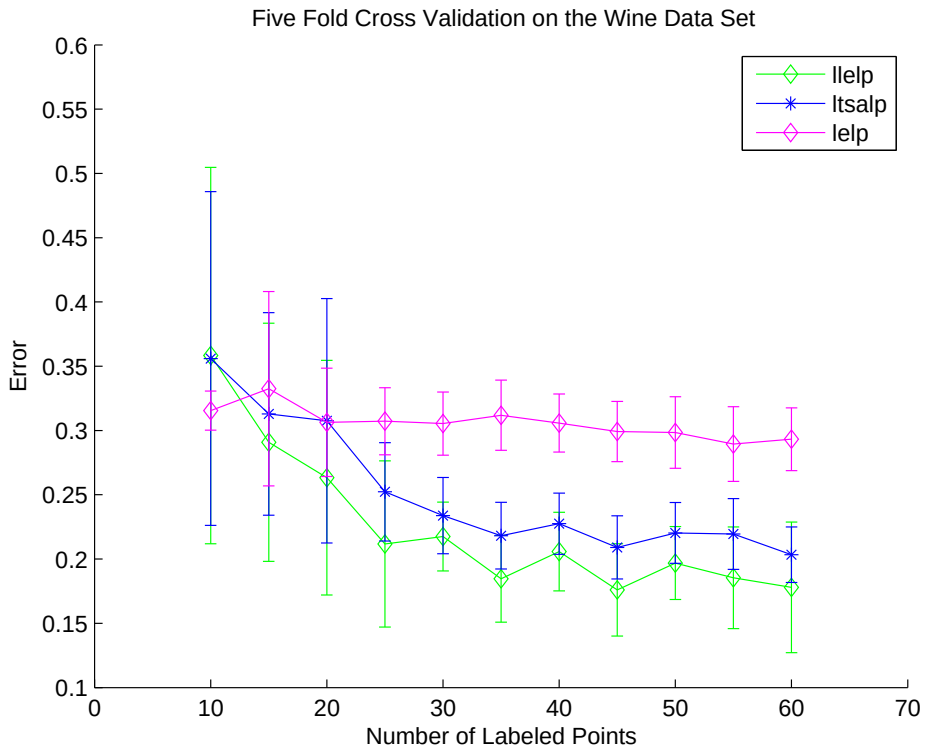
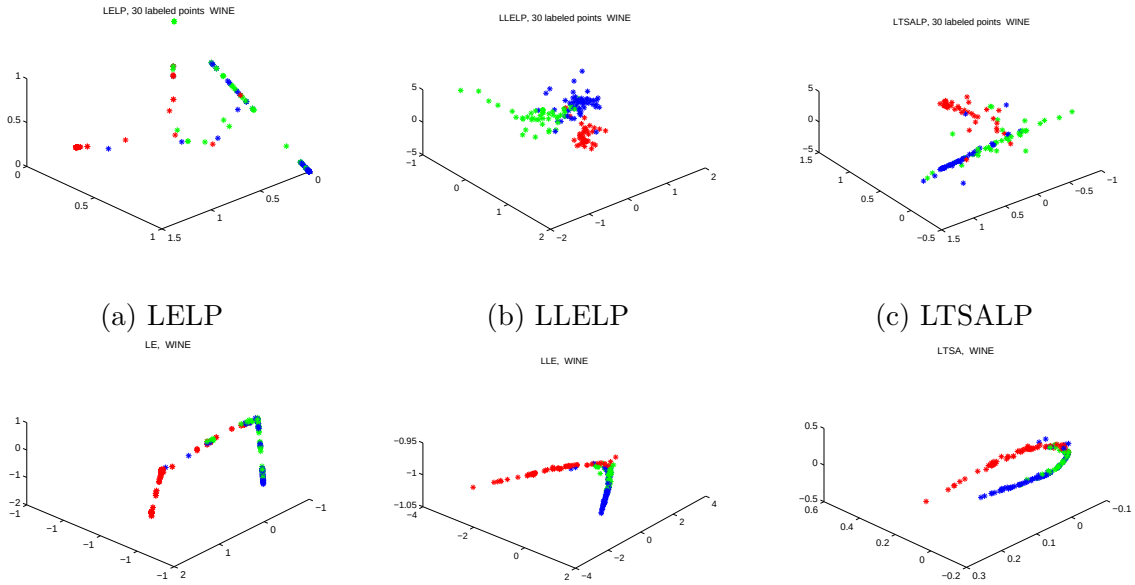


Figure 3.7: Embedding based LP methods on Wine corresponding to the Figure 3.8.



(a) LELP (b) LLELP (c) LTSALP

(d) LE (e) LLE (f) LTSA

Figure 3.8: Unsupervised and unconstrained semi-supervised embedding on Wine. The prediction performance (top) is better if the unsupervised embedding (bottom) keeps the classes separate.

Chapter 4

Bayesian Multi-Variate Regression

In this chapter we introduce a scalable algorithm for multi-label classification capable of modeling the covariance across labels.

4.1 Introduction

Multi-label classification problems, where a data point can have multiple labels, are becoming important in a variety of domains, such as protein function prediction in computational biology, image annotation in computer vision, and text categorization in document analysis [66, 67, 68]. For a problem with K possible labels, there are $M = (2^K - 1)$ distinct categories, so a direct multi-class classification with M classes would be computationally prohibitive, and may miss out on any covariance among the labels. Until a few years ago, the convention was to train and apply binary one-vs-rest classifiers to each label separately. However, even this approach ignores the possibility of label covariance, which is typically present in real world domains.

In addition to accounting for label covariance, multi-label classification methods should be scalable and capable of handling partially labeled data. Real-world data sets tend to be ever-growing. Furthermore, labels are sometimes only partially observed. While a data point can have any subset of K possible labels, in reality we often know only some of the labels it has for sure, with no information whatsoever on certain

other labels. The problem of partial labels is unique to multi-label classification, and is important in real world applications, such as in protein function prediction [66].

We propose Bayesian Multivariate Regression (BMR), a novel and highly scalable algorithm for multi-label classification and missing label prediction. The proposed approach is a probabilistic model based on multi-variate regression. In BMR, every data point is mapped to a Gaussian distribution, whereby each distribution shares the same covariance matrix. The resulting distributions are eventually utilized in the generation of the individual labels. As a result BMR is capable of modeling covariances among labels while being very efficient and applicable to large data sets. The prediction in BMR reduces to a mere matrix multiplication. We illustrate the performance of BMR on a variety of benchmark data sets ranging from 593 to 10,000 in size. The multi-label classification results indicate a competitive performance across all evaluation measures, both in terms of prediction and execution time.

In large real-word data sets frequently only partially labeled data are available. Typically two questions arise: (1) Can we predict the missing labels accurately? (2) Can we utilize the structure from the partially labeled data to obtain an improved classifier for future points? We extend the BMR model to handle missing labels. This is done by utilizing the covariance structure captured by the BMR model. To evaluate performance on predicting missing labels we compare it to Probabilist Matrix Factorization (PMF) [69], a well-known state-of-the-art approach. Despite the simplicity of the BMR model, the results indicate a superior performance. To address the second question we conduct experiments in which we contrast performance in cases where partially labeled data is used for training and cases where partially labeled data is discarded. The results illustrate a significant improvement in performance when partially labeled data are used.

The rest of the chapter is organized as follows: In Section 4.2, we give a brief overview on related work. In Section 6.2, we propose our Bayesian Multivariate Regression approach and a variational inference algorithm to learn the model. We present the experimental results in Section 4.4, and conclude in Section 4.5.

4.2 Related Work

Conventionally, multi-label classification problems are solved by decomposing them into multiple independent binary classification problems, while ignoring relationships between labels. In recent years, several approaches have been proposed which attempt to utilize the correlation structure among labels.

Kernel methods for multi-label classification tend to be extensions of the maximum margin idea. In [70], a maximum margin approach is proposed which minimizes the ranking loss. In [71], a method is proposed to learn a kernel which is shared across labels, to be subsequently used in individual label classifiers. While the ability to handle kernels is important in several domains, most existing approaches do not have a natural way of dealing with missing labels and are not probabilistic, i.e., no direct uncertainty quantification.

A number of probabilistic models have also been proposed for multi-label classification. In [68], a mixture model is proposed for text classification. More recently, in [72], a fully Bayesian model was proposed based on sparse and infinite canonical correlation analysis. It directly models correlations among labels and is one of few models which has the flexibility of dealing with missing labels. An extension of Gaussian Process prediction to the multi-label setting was proposed in [73].

The state-of-the-art also includes two approaches based on the k-nearest neighbor idea. In [42], label statistics from neighborhoods are used to build a Bayesian classifier. In [74], features are constructed based on label information from neighborhoods and subsequently used in logistic regression. In recent years, a family of methods based on multi-label dimensionality reduction has emerged [75, 76]. Our proposed model also falls in this category. Another interesting approach is presented in [77], where semi-supervised multi-label classification is proposed using the Sylvester equation.

There are two major problems with most existing approaches. They have a tendency not to explicitly model correlations among labels, but rather attempt to indirectly incorporate them. The second issue is that most existing approaches are too complex to be applicable to large scale datasets. Unlike most existing methods, our approach is a scalable probabilistic method which explicitly models the correlation structure among labels.

4.3 Bayesian Multivariate Regression

In multi-label classification, every data object is associated with a subset of possible labels. Assuming a total of c possible labels $L = \{\ell_1, \ldots, \ell_c\}$, for any given data object $\mathbf{x}$, the label information can be captured by a c-length bit vector $\mathbf{h} \in \{0,1\}^c$, where $h_s = 1$ denotes the membership of $\mathbf{x}$ in class s.

4.3.1 The Model

We now introduce our novel approach which we call Bayesian Multivariate Regression (BMR). Given a real valued feature vector $\mathbf{x} \in \mathbb{R}^k$ we assume a mapping $W \in \mathbb{R}^{c \times k}$, such that $\mu(\mathbf{x}) = W\mathbf{x}$. Subsequently we draw a latent label vector representation $\boldsymbol{\eta}$ from $N(\mu(\mathbf{x}), \Sigma)$, where $\Sigma \in R^{c \times c}$ denotes a covariance matrix among classes. While the covariance Σ is global in our model, the mean $\mu(\mathbf{x})$ differs for every data point. Our latent variable can alternatively be expressed as

$$\boldsymbol{\eta} = W\mathbf{x} + \varepsilon$$

where $\varepsilon \sim N(0, \Sigma)$. From this we can see that the empirical covariance of $\boldsymbol{\eta}$ will not be solely determined by Σ, but rather jointly by the mean function $\mu(\mathbf{x})$ and Σ.

Each entry $\eta_{ni} \in \mathbb{R}$ is treated as the natural parameter of a Bernoulli distribution determining if the data point $\mathbf{x}_n$ has label ℓ_i. Thus, the label $y_{ni} \in \{0,1\}$ is generated from the Bernoulli distribution with mean parameterization: $\theta_i(\eta_n) = \left[\frac{1}{1+\exp(\eta_{ni})} \ , \ \frac{\exp(\eta_{ni})}{1+\exp(\eta_{ni})}\right]$.

Since it does not consider the marginal distribution over $\mathbf{x}$, BMR is a discriminative model.

Let $\mathbf{x}_n$ denote a k-dimensional data point, the generative process for each label c-dimensional label vector $\mathbf{y}_n$ can be specified as follows:

1. $\boldsymbol{\eta}_n \sim N(W\mathbf{x}_n, \Sigma)$.
2. $\mathbf{y}_{ni} \sim \text{Discrete}(\theta_i(\eta_n))$, for $i = 1 \ldots c$.

For convenience, we express the Bernoulli distribution $p(y_{ni}|\eta_n)$ of the labels given the natural parameterization in standard exponential family form as: $p(y_{ni}|\eta_n) =$

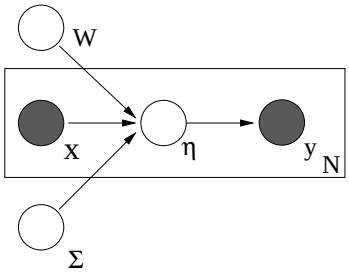


Figure 4.1: Graphical model for Bayesian Multivariate Regression.

$\exp\{y_{ni}\eta_{ni} - \ln(1+\exp^{\eta_{ni}})\}$. With $C(\eta_n) = \sum_{i=1}^{c} \ln(1+\exp^{\eta_{ni}})$, for $\mathbf{y}_n$ we have:

$$p(\mathbf{y}_n|\eta_n) = \prod_{i=1}^{c} p(y_{ni}|\eta_n) = \exp\left\{\mathbf{y}_n^T\eta_n - C(\eta_n)\right\} \ . \tag{4.1}$$

The graphical model for BMR is shown in Figure 4.1. Given the model, the likelihood function of $\mathbf{y}_n$ is given by

$$\begin{aligned} p(\mathbf{y}_n|\mathbf{x}_n, \Sigma, W) &= \int_{\boldsymbol{\eta}_n} p(\boldsymbol{\eta}_n, \mathbf{y}_n|\mathbf{x}_n, \Sigma, W) d\boldsymbol{\eta}_n \\ &= \int_{\boldsymbol{\eta}_n} p(\boldsymbol{\eta}_n|W\mathbf{x}_n, \Sigma) p(\mathbf{y}_n|\boldsymbol{\eta}_n) d\boldsymbol{\eta}_n \ . \\ &= E_{\boldsymbol{\eta_n}}[p(\mathbf{y}|\boldsymbol{\eta_n})] \end{aligned} \tag{4.2}$$

Therefore, for a dataset with N data points $X = \{\mathbf{x}_n, [n]_1^N\}$ ($[n]_1^N \equiv n = 1 \ldots N$) and $Y = \{\mathbf{y}_n, [n]_1^N\}$, the likelihood function is

$$\begin{aligned} p(Y|X, \Sigma, W) &= \prod_{n=1}^{N} \int_{\boldsymbol{\eta}_n} p(\boldsymbol{\eta}_n|W\mathbf{x}_n, \Sigma) p(\mathbf{y}_n|\boldsymbol{\eta}_n) d\boldsymbol{\eta}_n \ . \\ &= \prod_{n=1}^{N} E_{\boldsymbol{\eta_n}}[p(\mathbf{y}|\boldsymbol{\eta_n})] \ . \end{aligned} \tag{4.3}$$

4.3.2 Inference and learning

For given data points X and corresponding Y, the learning task of BMR involves finding the model parameters W and Σ, such that the likelihood of $p(Y|X, \Sigma, W)$ as in Equation (4.3) is maximized. A general approach for such a task is to use multivariate optimization algorithms. However, the likelihood function in (4.3) is intractable, implying that

a direct application of optimization is infeasible. Therefore, we propose a variational inference method, which alternates between obtaining a tractable lower bound to the true log-likelihood and choosing the model parameters W and Σ to maximize the lower bound.

In order to obtain a tractable lower bound to (4.2), instead of using the true latent variable distribution $p(\boldsymbol{\eta}_n|W\mathbf{x}_n, \Sigma)$ in expectation calculation, we introduce a family of parameterized variational distributions $q(\boldsymbol{\eta}_n|\hat{\boldsymbol{\mu}}_n, \hat{\Sigma}_n)$ as an approximation to $p(\boldsymbol{\eta}_n|W\mathbf{x}_n, \Sigma)$, where $q(\boldsymbol{\eta}_n|\hat{\boldsymbol{\mu}}_n, \hat{\Sigma}_n)$ is a Gaussian distribution, and $\hat{\boldsymbol{\mu}}_n$ and $\hat{\Sigma}_n$ are variational parameters denoting the mean and covariance. Following Jensen's Inequality [39], we have

$$\begin{aligned}\log p(\mathbf{y}_n|\mathbf{x}_n, \Sigma, W) &\geq E_q[\log p(\boldsymbol{\eta}_n, \mathbf{y}_n|\mathbf{x}_n, W, \Sigma)] - E_q[\log q(\boldsymbol{\eta}_n|\hat{\boldsymbol{\mu}}_n, \hat{\Sigma}_n)] \\ &= E_q[\log p(\boldsymbol{\eta}_n|\mathbf{x}_n, W, \Sigma)] + E_q[\log p(\mathbf{y}_n|\boldsymbol{\eta}_n)] - E_q[\log q(\boldsymbol{\eta}_n|\hat{\boldsymbol{\mu}}_n, \hat{\Sigma}_n)] \ .\end{aligned} \tag{4.4}$$

We can denote the lower bound (4.4) using $L(\hat{\boldsymbol{\mu}}_n, \hat{\Sigma}_n, W, \Sigma)$, and each term in $L(\hat{\boldsymbol{\mu}}_n, \hat{\Sigma}_n, W, \Sigma)$ are given by

$$\begin{aligned}E_q[\log p(\boldsymbol{\eta}_n|\mathbf{x}_n, W, \Sigma)] &= -\frac{1}{2}\left(\mathrm{Tr}(\Sigma^{-1}\hat{\Sigma}_n) + (\hat{\boldsymbol{\mu}}_n - W\mathbf{x}_n)^T\Sigma^{-1}(\hat{\boldsymbol{\mu}}_n - W\mathbf{x}_n)\right) \\ &\quad - \frac{c}{2}\log 2\pi + \frac{1}{2}\log|\Sigma^{-1}| \\ E_q[\log p(\mathbf{y}_n|\boldsymbol{\eta}_n)] &= E_q[\mathbf{y}_n^T\eta_n - C(\eta_n)] \approx \mathbf{y}_n^T\hat{\boldsymbol{\mu}}_n - [C(\zeta_n) + g(\zeta_n)^T(\hat{\boldsymbol{\mu}}_n - \zeta_n)] \\ E_q[\log q(\boldsymbol{\eta}_n|\hat{\boldsymbol{\mu}}_n, \hat{\Sigma}_n)] &= -\frac{k}{2} - \frac{k}{2}\log 2\pi + \frac{1}{2}\log|\hat{\Sigma}_n^{-1}|\end{aligned}$$

Note, we approximated the term $E_q[\log p(\mathbf{y}_n|\eta_n)]$ by considering a first order Taylor expansion for $C(\eta_n)$. We note that $\frac{\partial C}{\partial \eta_{ni}} = \frac{\exp \eta_{ni}}{1+\exp \eta_{ni}} = g_i(\eta_n)$, where g denotes the gradient. For any fixed vector $\zeta_n \in \mathrm{int}(\mathrm{dom}(C))$ the first order Taylor approximation for $C(\eta_n)$ is given by:

$$C(\zeta_n) + g(\zeta_n)(\eta_n - \zeta_n) \ .$$

The best lower bound can be obtained by maximizing each $L(\hat{\boldsymbol{\mu}}_n, \hat{\Sigma}_n, W, \Sigma)$ with respect to the variational parameters $\hat{\boldsymbol{\mu}}_n$ and $\hat{\Sigma}_n$, which gives

$$\hat{\boldsymbol{\mu}}_n = W\mathbf{x}_n + \Sigma[\mathbf{y}_n - g(\zeta_n)] \tag{4.5}$$

$$\hat{\Sigma}_n = \Sigma. \tag{4.6}$$

The lower bound of the log-likelihood on the whole dataset Y is given by summing the lower bound over the individual points $\sum_{n=1}^{N} L(\hat{\boldsymbol{\mu}}_n, \hat{\Sigma}_n, W, \Sigma)$. To obtain the estimate for model parameters, we use this lower bound function as a surrogate objective to be maximized. Given a fixed value of $(\hat{\boldsymbol{\mu}}_n^*, \hat{\Sigma}_n^*)$ from (4.5) and (4.6), the lower bound function $\sum_{n=1}^{N} L(\hat{\boldsymbol{\mu}}_n^*, \hat{\Sigma}_n^*, W, \Sigma)$ is a function of model parameters (W, Σ). By maximizing $\sum_{n=1}^{N} L(\hat{\boldsymbol{\mu}}_n^*, \hat{\Sigma}_n^*, W, \Sigma)$ with respect to W and Σ, we have

$$W = \left(\sum_{n=1}^{N} \hat{\boldsymbol{\mu}}_n \mathbf{x}_n^T\right) \left(\sum_{n=1}^{N} \mathbf{x}_n \mathbf{x}_n^T\right)^{-1} \tag{4.7}$$

$$\Sigma = \frac{1}{N} \sum_{n=1}^{N} \left(\hat{\Sigma}_n + (\hat{\boldsymbol{\mu}}_n - W\mathbf{x}_n)(\hat{\boldsymbol{\mu}}_n - W\mathbf{x}_n)^T\right) \ . \tag{4.8}$$

4.3.3 Variational optimization

Following the update equations in (4.5)-(4.8), we construct a variational optimization algorithm to learn the model. Starting from an initial guess of $(W^{(0)}, \Sigma^{(0)})$, the algorithm alternates between the following two steps in each iteration t:

1. Inference-step: Given $(W^{(t-1)}, \Sigma^{(t-1)})$, for each $(\mathbf{x}_n, \mathbf{y}_n)$, find the optimal variational parameters

$$(\hat{\boldsymbol{\mu}}_n^{(t)}, \hat{\Sigma}_n^{(t)}) = \arg\max_{(\hat{\boldsymbol{\mu}}_n, \hat{\Sigma}_n)} L(\hat{\boldsymbol{\mu}}_n, \hat{\Sigma}_n, W^{(t-1)}, \Sigma^{(t-1)}) \ ,$$

   which can be done using (4.5) and (4.6).

2. Optimization-step: Maximizing the aggregate lower bound gives us an improved estimate of the model parameters:

$$(W^{(t)}, \Sigma^{(t)}) = \arg\max_{(W,\Sigma)} \sum_{n=1}^{N} L(\hat{\boldsymbol{\mu}}_n^{(t)}, \hat{\Sigma}_n^{(t)}, W, \Sigma) \ ,$$

   which can be done following (4.7) and (4.8).

After t iterations, the objective function becomes $L(\hat{\boldsymbol{\mu}}_n^{(t)}, \hat{\Sigma}_n^{(t)}, W^{(t)}, \Sigma^{(t)})$. In the $(t+1)^{th}$ iteration, we have

$$\begin{aligned}\sum_{n=1}^{N} L(\hat{\boldsymbol{\mu}}_n^{(t)}, \hat{\Sigma}_n^{(t)}, W^{(t)}, \Sigma^{(t)}) &\leq \sum_{n=1}^{N} L(\hat{\boldsymbol{\mu}}_n^{(t+1)}, \hat{\Sigma}_n^{(t+1)}, W^{(t)}, \Sigma^{(t)}) \\ &\leq \sum_{n=1}^{N} L(\hat{\boldsymbol{\mu}}_n^{(t+1)}, \hat{\Sigma}_n^{(t+1)}, W^{(t+1)}, \Sigma^{(t+1)}) \ .\end{aligned}$$

The first inequality holds because $(\hat{\boldsymbol{\mu}}_n^{(t+1)}, \hat{\Sigma}_n^{(t+1)})$ maximizes $L(\hat{\boldsymbol{\mu}}_n, \hat{\Sigma}_n, W^{(t)}, \Sigma^{(t)})$ in the Inference-step. The second inequality holds because $(W^{(t+1)}, \Sigma^{(t+1)})$ maximizes $\sum_{n=1}^{N} L(\hat{\boldsymbol{\mu}}_n^{(t+1)}, \hat{\Sigma}_n^{(t+1)}, W^{(t+1)}, \Sigma^{(t+1)})$ in the Optimization-step. Therefore, the objective function is non-decreasing until convergence.

We note that the computations involved per iteration during training are scalable. Most operations involved are simple matrix multiplications or matrix-vector products. There is a matrix inversion involving a $d \times d$ matrix in (4.7), but since the matrix only depends on the feature vectors $\mathbf{x}_n$, the inverse can be computed offline, even before starting the iterations. The algorithm does need to invert Σ in every iteration. Since Σ is a $c \times c$ matrix where c is the number of classes, the inverse can be computed efficiently even for hundreds of classes.

4.3.4 Learning With Missing Labels

In certain application domains, such as protein function prediction in computational biology [66], one encounters multi-label classification problems where data points have partial labels, i.e., instead of knowing $\mathbf{y} \in \{0,1\}^K$, we know only some of the entries in $\mathbf{y}$, the others being missing. Most existing multi-label approaches were not designed to handle labels with missing values [70, 42]. However information contained even in partially observed label vectors is potentially valuable and it is important for a model to be able to utilize it. Since BMR provides a generative model for labels and it captures correlations among labels, it can naturally handle data sets with partial labels. The training of our model when missing labels are present has to be adjusted slightly. From a generative model perspective, the missing labels are assumed not to be generated by the model. However, during training, the corresponding $\hat{\mu}$ and all parameters which depend on it still have to be estimated.

To allow learning when some labels are missing for a data point, we need to adjust the estimation of $\hat{\boldsymbol{\mu}} \in \mathbb{R}^K$. We break down the component variables into entries corresponding to known labels and unknown labels. In particular we let

$$\hat{\boldsymbol{\mu}} = \begin{bmatrix} \hat{\boldsymbol{\mu}}_l \\ \hat{\boldsymbol{\mu}}_u \end{bmatrix} , \qquad \Sigma^{-1} = \begin{bmatrix} \Sigma_{ll}^{-1} & \Sigma_{lu}^{-1} \\ \Sigma_{ul}^{-1} & \Sigma_{uu}^{-1} \end{bmatrix} , \qquad W = \begin{bmatrix} W_l \\ W_u \end{bmatrix} ,$$

where l denotes set of known labels and u denotes a set of unknown labels. In a similar fashion we break down all the other parameters: $\hat{\mu}$ and $\mathbf{y}$. For a data point with missing labels, the problem of estimating $\hat{\mu}$ can be posed as:

$$\max_{\hat{\boldsymbol{\mu}}_l, \hat{\boldsymbol{\mu}}_u} \left\{ -\frac{1}{2} \left(\begin{bmatrix} \hat{\boldsymbol{\mu}}_l \\ \hat{\boldsymbol{\mu}}_u \end{bmatrix} - \begin{bmatrix} W_l \mathbf{x}_n \\ W_u \mathbf{x}_n \end{bmatrix} \right)^T \Sigma^{-1} \left(\begin{bmatrix} \hat{\boldsymbol{\mu}}_l \\ \hat{\boldsymbol{\mu}}_u \end{bmatrix} - \begin{bmatrix} W_l \mathbf{x}_n \\ W_u \mathbf{x}_n \end{bmatrix} \right) + \mathbf{y}_l^T \hat{\boldsymbol{\mu}}_l - g(\zeta_l)^T \hat{\boldsymbol{\mu}}_l \right\} .$$

The first order conditions yield the updates:

$$\hat{\boldsymbol{\mu}}_l = W_l \mathbf{x}_n + (\Sigma_{ll}^{-1})^{-1} (\Sigma_{lu}^{-1} (\hat{\boldsymbol{\mu}}_u - W_u \mathbf{x}_n) + \mathbf{y}_l - g(\zeta_l)) \tag{4.9}$$

$$\hat{\boldsymbol{\mu}}_u = \left(\Sigma_{uu}^{-1}\right)^{-1} \Sigma_{ul}^{-1} (\hat{\boldsymbol{\mu}}_l - W_l \mathbf{x}_n) + W_u \mathbf{x}_u . \tag{4.10}$$

These updates are coupled, therefore we perform them iteratively. The computation of all other variables/parameters remains the same (in terms of $\hat{\boldsymbol{\mu}}$) when missing values are present.

4.3.5 Prediction

Assuming that Σ and W have been estimated from training data, we wish to predict the label vector $\bar{\mathbf{h}}$ for an unseen data point $\bar{\mathbf{x}}$. First note that the maximum likelihood estimate of $\bar{\boldsymbol{\eta}}$, given W and Σ is obtained by $\bar{\boldsymbol{\eta}}^* = W\bar{\mathbf{x}}$, since $\bar{\boldsymbol{\eta}} \sim N(W\bar{\mathbf{x}}, \Sigma)$. With $\bar{\boldsymbol{\eta}}^*$ interpreted as a vector of log odds we can obtain the maximum likelihood estimate for $\bar{\mathbf{y}}$, by considering the sign of the individual entries of $\bar{\boldsymbol{\eta}}^*$:

$$\bar{\boldsymbol{\eta}}^* = W\bar{\mathbf{x}} \tag{4.11}$$

with

$$\bar{h}_i = \begin{cases} 1 & \text{if } \bar{\boldsymbol{\eta}}_i^* > 0 \\ 0 & \text{otherwise} . \end{cases} \tag{4.12}$$

Effectively the prediction task in our model reduces to a matrix multiplication. For this reason our model can be seen as rather simple, and unlike most existing approaches, it can be easily used on millions of data points.

4.4 Empirical Evaluation

In this section we present our experimental results on both multi-label classification and missing label prediction. Our empirical evaluation consists of four parts. We first evaluate BMR on multi-label classification with fully labeled data. Subsequently we contrast BMR to PMF on missing label prediction. We then examine to what extent BMR can take advantage of partially labeled data, when making predictions on unseen points. Lastly we evaluate the execution time for learning and making predictions.

4.4.1 Data Sets

We evaluated the BMR model using a number of data sets ranging in size from 593 to 10,000 data points. The number of labels in these data sets range from 6 to 374. The main large data set that we have used, which also motivated the development of BMR [11] stems from the Aviation Safety Reporting System (ASRS). It encompasses a subset of 10,000 aviation safety reports assigned to 58 anomaly categories. The ASRS-10000 data set contains 200 features obtained from running Latent Dirichlet Allocation (LDA) [39] on the original reports. Further data sets that we used include Mediamill-10000. A subset of 10,000 data points from the Mediamill data set [78]. It includes 120 features and 101 labels. Two smaller data sets that we used include Scene [79] and Emotions [80]. In table 5.2 a summary is provided for all of the data sets used.

Data Set	**Data Points**	**Features**	**Labels**
Scene	2407	294	6
Emotions	593	72	6
ASRS-10000	10000	200	58
Mediamill-10000	10000	120	101

Table 4.1: Data sets used for empirical evaluation.

4.4.2 Algorithms

We compare BMR with three multi-label classification algorithms. As baselines, we consider one-vs-rest SVM as a multi-label classifier, which we refer to as MLSVM. In addition we use a one-vs-rest implementation of logistic regression, which we call MLLR. We also consider two state-of-the-art approaches for multi-label learning: Multi-label K-nearest Neighbors (MLKNN) [42], a method which applies the k-nearest neighbor idea to the multi-label setting; and Instance Based Learning by Logistic Regression (IBLR) [74], where features are first transformed to incorporate label information from local neighborhoods prior to applying logistic regression. Additionally we use a chain of logistic regression classifiers (CCML) [81], a recently proposed approach which models each label conditioned on all other labels. To evaluate performance on missing value prediction we use Probabilistic Matrix Factorization (PMF) as a baseline.

4.4.3 Methodology

All of our multi-label classification and missing value prediction experiments are conducted using five-fold cross validation. Five-fold cross validation is also used to tune any parameters that are needed. Partially labeled data is utilized in two ways. We evaluate the performance in predicting missing label entires in the training data. We also evaluate the prediction performance on unseen points, given a partially labeled training set. Lastly in our computational efficiency experiments execution time is measured in CPU time, as given by the *cputime* function in MATLAB.

Evaluation Measures

We evaluated classification performance using five different measures: one error, precision, coverage, ranking loss, and Hamming loss. Let $g(x, l)$ denote a real-valued function which assigns a score to label l for data point $\mathbf{x}$, such that a larger score is considered better. Also, let $f(\mathbf{x})$ denote the classifier whose output is the predicted multi-label vector. Further, let L_x denote a set of true labels associated with $\mathbf{x}$.

1) *One error* evaluates how frequently the top ranked predicted label is not among

the true labels. If $\mathbb{1}[\cdot]$ denotes the indicator function, we have:

$$OneError(g) = \frac{1}{D}\sum_{d=1}^{D} \mathbb{1}\left[\text{argmax}_{l\in L}\, g(\mathbf{x}_d, l) \notin L_{x_d}\right] \ . \tag{4.13}$$

2) For true labels $l \in L_x$, *average precision* evaluates the fraction of labels in L_x that rank at least as high as l according to the scoring rule g on average. For any data point x and any label $l \in L_x$, let $\mathcal{R}(\mathbf{x}, l) = \{l' \in L_{\mathbf{x}} | rank_g(\mathbf{x}, l') \leq rank_g(\mathbf{x}, l)\}$, where the ranking is among all possible labels. Then, average precision is:

$$AvePrec(g) = \frac{1}{D}\sum_{d=1}^{D} \frac{1}{|L_{\mathbf{x}_d}|} \sum_{l\in L_{\mathbf{x}_d}} \frac{|\mathcal{R}(\mathbf{x}_d, l)|}{rank_g(\mathbf{x}_d, l)} \ . \tag{4.14}$$

3) Coverage reflects on average how far one needs to go down in the label ranking to cover all actual labels of an instance:

$$Coverage(g) = \frac{1}{D}\sum_{d=1}^{D} (\max_{l\in L_{\mathbf{x}_d}} rank_g(\mathbf{x}_d, l) - 1) \ . \tag{4.15}$$

4) Hamming loss evaluates the fraction of label instance pairs that were misclassified:

$$HammingLoss(f) = \frac{1}{D}\sum_{d=1}^{D} \frac{1}{c} |f(\mathbf{x}_d) \triangle L_{\mathbf{x}_d}| \ . \tag{4.16}$$

where $\triangle$ denotes the symmetric difference between two sets.

5) Ranking loss reflects the average number of labels that are reversely ordered for a given instance. Let $\mathcal{T}(\mathbf{x}_d) = \{(l_1, l_2) \mid g(\mathbf{x}_d, l_1) \leq g(\mathbf{x}_d, l_2),\ (l_1, l_2) \in L_{\mathbf{x}_d} \times \bar{L}_{x_d}\}$, where $\bar{L}_{\mathbf{x}_d}$ denotes the complement of L_{x_d}. Ranking loss is defined as:

$$RankLoss(g) = \frac{1}{D}\sum_{d=1}^{D} \frac{|\mathcal{T}(\mathbf{x}_d)|}{|\bar{L}_{\mathbf{x}_d}||L_{\mathbf{x}_d}|} \ . \tag{4.17}$$

For both Hamming loss and ranking loss, smaller values are better. In particular for a perfect performance $HammingLoss(h) = RankLoss(g) = 0$.

The prediction of missing labels within training data is evaluated using Hamming loss only.

4.4.4 Multi-Label Classification

Tables 4.2-4.5 list the prediction results when using five fold cross validation. MLSVM and MLLR, the two one vs. rest approaches perform the worst, as expected. These results clearly illustrate that looking at Hamming loss alone is actually quite misleading. For instance MLSVM has a hamming loss of 11.9% for ASRS-10000, however its one error is at 85.8%. Even for a degenerate classifier which predicts only zeros, one would obtain a low hamming loss. For this reason we have opted to evaluate our results using a range of five different evaluation measures, commonly used in multi-label classification.

Our proposed model performs either better or very competitively compared to all other approaches, including MLKNN, IBLRML, and the two state-of-the-art methods across all five evaluation measures. Across all evaluation measures our approach seems to be followed by MLKNN and then IBLRML. Considering the simplicity of our approach, these results are quite interesting. After all, the predictive step in our model merely involves a matrix multiplication, and yet we are outperforming very complex algorithms such as SVMs or even state-of-the-art multi-label learning methods such as MLKNN and IBLRML. CCML appears to be more competitive on the larger data sets. However, overall it does not seem to be standing out.

For the top four algorithms, BMR, MLKNN, CCML and IBLRML, we also examined what happens when a smaller fraction of the data set is labeled. We omitted the one vs. rest approaches to prevent clutter, and also since we already established that their performance is substantially inferior. We ran 5-fold cross validation while gradually increasing the set of labeled points. The results can be seen in Figures 5.3-4.5. The first thing that we can note is that overall the performance of IBLRML appears to be worse than that of MLKNN. CCML seems to perform either worse or as good as IBLRML. Across all evaluation measures our proposed method, BMR, consistently performs either competitively or better compared to all three algorithms. This seems to indicate that our approach is robust, even when the size of the training set is reduced.

4.4.5 Partially Labeled Data

In our first set of experiments involving partially labeled data we evaluate the Hamming loss in predicting missing labels. The results for the Emotions and Scene data sets are

given in Table 4.6. We compare the performance to Probabilistic Matrix Factorization (PMF), as the percentage of missing values is varied between 10% and 30% of the label entries. The results are obtained by using again five-fold cross validation. As we can see across both benchmark data sets BMR outperforms the well-known PMF. Even though BMR is a rather simplistic model, it performs quite well. While PMF only considers the label matrix when making predictions, BMR considers both observed labels and observed features.

The second set of experiments involving partially labeled data examines the prediction performance on unseen points, given a training set with partially labeled data. On both the Emotions and Scene data sets we set randomly 30% of the label entries to be missing. We than evaluate the prediction performance on unseen data points. This is done using five-fold cross validation and all five evaluation measures described earlier. The performance of BMR is compared to BMR-D, CCML-D, MLKNN-D and IBLRML-D. The suffix "D" indicates that partially labeled data in the training set is discarded in these versions of the algorithms. The point of this experiment is see whether BMR can benefit from the additional structure provided by partially labeled data. The results are given in Table 4.7. BMR clearly dominates. The results seem to suggest that BMR can indeed take advantage of the structure learned from partially labeled data.

4.4.6 Scalability

To contrast the computational cost involved in utilizing MLKNN, IBLRML and BMR we conducted an experiments in which we tested how long it takes to learn the respective models and make predictions. We excluded CCML from these experiments due to its poor performance in the previous evaluations. The number of data points both in the training set and the test set was varied from 2000 points to 6000 points. The Mediamill data set with 120 features and 101 labels was used for this purpose. The experiments were conducted on a 8 core Dell PowerEdge machine, with cores clocked at 2.3 GHz and 32 GB of RAM. The execution time for learning the models can be seen in figure 4.6. IBLRML clearly takes the longest time to be trained. This can be attributed to the fact that it requires 101 Logistic Regression models to be learned. MLKNN on the other hand only needs to be trained once. Overall we can see that BMR is the most efficient out of the three when it comes to learning.

Figure 4.7 illustrates the execution time for making predictions. In this case MLKNN performs the worst. Even when making predictions it has to compute the K-nearest neighbors. IBLRML is somewhat better, even though it maintains 101 Logistic Regression models, once they are trained making predictions can be done in a reasonable time. BMR significantly outperforms both approaches. This can be attributed to the fact that in BMR making predictions is reduced to a mere matrix multiplication. As a result BMR can be easily applied to millions of records in less than a second, a property which is very desirable in many real-world domains.

There appears to be a trade off between IBLRML and MLKNN. While one method dominates the other for learning, for making predictions the picture is reversed. No such trade off has to be made with BMR, which is consistently efficient both for learning and making predictions.

Table 4.2: Five-fold cross validation on the ASRS-10000 data set. BMR clearly outperforms all the other methods.

	BMR	CCML	MLKNN	IBLRML	MLLR	MLSVM
OneError	$\mathbf{39.6 \pm 0.8}$	47.6 ± 1.3	43.7 ± 0.8	44.3 ± 1.4	50.7 ± 1.6	85.8 ± 18.1
AvePrec	$\mathbf{63.4 \pm 0.7}$	59.9 ± 0.9	60.1 ± 0.6	60.3 ± 0.6	57.0 ± 0.9	33.6 ± 8.2
Coverage	$\mathbf{7.77 \pm 0.19}$	10.3 ± 0.5	9.18 ± 0.36	8.39 ± 0.29	9.63 ± 0.51	13.81 ± 0.87
HammingLoss	$\mathbf{4.3 \pm 0.0}$	4.8 ± 0.1	4.6 ± 0.1	4.7 ± 0.0	5.5 ± 0.1	11.9 ± 1.1
RankLoss	$\mathbf{5.5 \pm 0.2}$	8.9 ± 0.8	6.9 ± 0.3	6.7 ± 0.3	7.9 ± 0.6	12.9 ± 1.7

Table 4.3: Five-fold cross validation on the Mediamill-10000 data set. BMR clearly outperforms all the other methods.

	BMR	CCML	MLKNN	IBLRML	MLLR	MLSVM
OneError	$\mathbf{16.7 \pm 0.6}$	25.3 ± 5.8	18.2 ± 1.0	28.7 ± 4.3	45.2 ± 2.1	76.7 ± 9.2
AvePrec	$\mathbf{67.8 \pm 0.3}$	62.7 ± 1.4	67.7 ± 0.3	62.3 ± 2.6	58.4 ± 1.3	37.3 ± 1.9
Coverage	$\mathbf{17.2 \pm 0.4}$	29.3 ± 1.8	17.4 ± 0.5	21.4 ± 2.4	22.1 ± 1.0	29.8 ± 1.3
HammingLoss	$\mathbf{3.1 \pm 0.0}$	3.4 ± 0.0	3.3 ± 0.0	4.5 ± 2.0	3.9 ± 0.0	4.8 ± 0.5
RankLoss	$\mathbf{4.8 \pm 0.1}$	12.9 ± 1.2	5.0 ± 0.2	7.5 ± 2.5	8.6 ± 0.6	10.5 ± 1.0

Table 4.4: Five-fold cross validation on the Emotions data set. BMR clearly outperforms all the other methods.

	BMR	CCML	MLKNN	IBLRML	MLLR	MLSVM
OneError	$\mathbf{26.7 \pm 0.8}$	35.9 ± 7.3	38.9 ± 4.3	44.9 ± 5.8	32.2 ± 5.4	70.9 ± 7.3
AvePrec	$\mathbf{80.1 \pm 1.4}$	75.6 ± 4.9	71.0 ± 1.4	67.3 ± 4.3	76.8 ± 2.3	55.0 ± 5.1
Coverage	$\mathbf{1.8 \pm 0.1}$	2.0 ± 0.2	2.2 ± 0.1	2.4 ± 0.2	1.9 ± 0.1	2.8 ± 0.2
HammingLoss	$\mathbf{20.4 \pm 0.8}$	23.7 ± 3.5	25.8 ± 0.4	36.7 ± 2.4	23.1 ± 1.5	25.1 ± 2.4
RankLoss	$\mathbf{16.7 \pm 1.3}$	20.6 ± 4.5	25.7 ± 1.7	30.0 ± 5.5	18.8 ± 1.7	39.2 ± 6.2

Table 4.5: Five-fold cross validation on the Scene data set. BMR clearly outperforms all the other methods.

	BMR	CCML	MLKNN	IBLRML	MLLR	MLSVM
OneError	27.1 ± 0.9	36.1 ± 2.0	$\mathbf{23.2 \pm 1.4}$	29.7 ± 3.0	35.6 ± 2.5	87.3 ± 6.1
AvePrec	83.2 ± 0.7	75.2 ± 1.2	$\mathbf{86.1 \pm 0.6}$	81.7 ± 1.6	75.0 ± 2.2	48.6 ± 4.7
Coverage	0.6 ± 0.0	1.0 ± 0.1	$\mathbf{0.5 \pm 0.0}$	0.7 ± 0.0	0.9 ± 0.1	1.6 ± 0.2
HammingLoss	11.2 ± 0.6	14.1 ± 0.6	$\mathbf{9.0 \pm 0.4}$	12.7 ± 1.3	14.3 ± 0.3	19.1 ± 1.9
RankLoss	10.0 ± 0.7	23.9 ± 1.2	$\mathbf{8.1 \pm 0.3}$	11.5 ± 1.0	19.8 ± 2.3	30.6 ± 4.1

4.5 Conclusions

In this chapter we have proposed Baysian Multivariate Regression (BMR), a rather simple but effective model for multi-label classification. Our proposed model addresses three very important challenges when it comes to multi-label data. It models the covariance structure among labels, thereby capturing more intricate dependencies. It is very scalable, which makes the approach applicable to a wide variety of real-world problems with ever-growing data sets. The scalability is possible since the learning step only involves matrix multiplications and inverting small matrices and the prediction step involves only a matrix multiplication. Lastly, BMR can naturally handle missing labels. This is especially important for huge data sets where a full labeling is not always available. For instance when it comes to micro-array data or recommendation systems not all values might be observed.

Our empirical evaluation shows that BMR is competitive with the state of the art both in multi-label classification and missing label prediction. We have tested the approach using a number of different data sets. BMR appears to be rather consistent in its performance. The most significant improvements can be seen in cases where partially

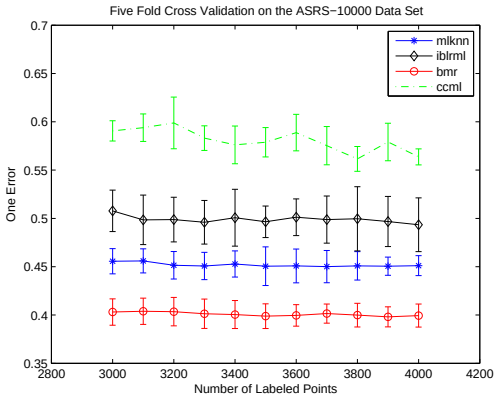
Table 4.6: Error rates for recovering missing labels obtained using five-fold cross validation on the Emotions and Scene data sets. Performance of BMR and PMF is evaluated while an increasing percentage of labels are missing in the training data. Missing Labels are randomly selected. The error rates reflect the percentage of missing labels incorrectly recovered.

	10%	15%	20%	25%	30%
Emotions					
PMA	27.1 ± 5.1	27.1 ± 4.8	26.8 ± 3.0	27.9 ± 2.0	28.6 ± 1.9
BMR	$\mathbf{19.8 \pm 4.6}$	$\mathbf{21.1 \pm 3.0}$	$\mathbf{21.6 \pm 4.1}$	$\mathbf{24.2 \pm 3.3}$	$\mathbf{24.6 \pm 2.1}$
Scene					
PMA	14.6 ± 5.6	$\mathbf{13.8 \pm 3.0}$	16.2 ± 3.4	18.5 ± 2.8	20.1 ± 3.0
BMR	$\mathbf{11.9 \pm 4.4}$	$\mathbf{13.8 \pm 2.3}$	$\mathbf{12.3 \pm 2.4}$	$\mathbf{13.2 \pm 2.0}$	$\mathbf{14.5 \pm 1.6}$

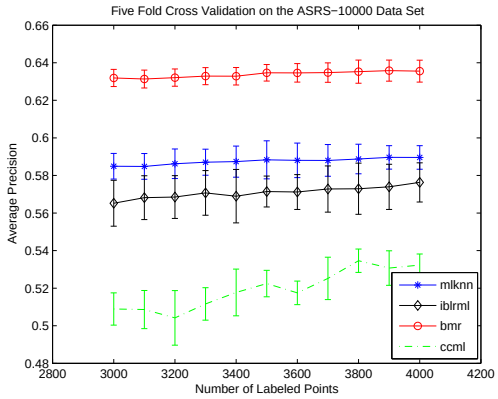
labeled data is used in training the model. BMR appears to capable of taking advantage of the additional structure provided by partially labeled training points.

Table 4.7: Five-fold cross validation on the Emotions and Scene data sets. BMR clearly outperforms all the other methods.

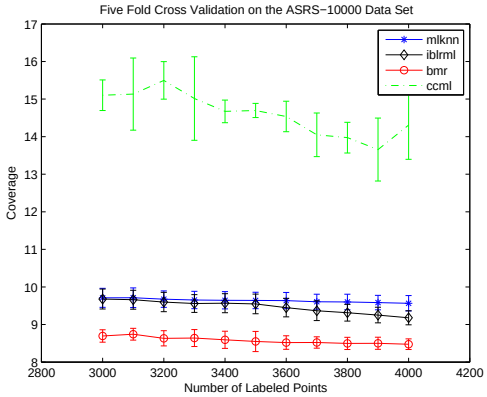
	BMR	BMR-D	CCML-D	MLKNN-D	IBLRML-D
Emotions					
OneError	$\mathbf{29.3 \pm 1.2}$	42.8 ± 5.7	49.1 ± 5.8	52.9 ± 8.6	59.1 ± 8.4
AvePrec	$\mathbf{77.9 \pm 3.4}$	70.9 ± 3.3	47.3 ± 1.0	59.4 ± 4.9	58.5 ± 5.1
Coverage	$\mathbf{1.9 \pm 0.2}$	2.2 ± 0.3	2.4 ± 0.2	2.9 ± 0.4	3.1 ± 0.3
HammingLoss	$\mathbf{21.2 \pm 3.9}$	28.3 ± 3.1	53.5 ± 1.0	31.6 ± 1.7	39.7 ± 4.4
RankLoss	$\mathbf{18.3 \pm 3.5}$	23.7 ± 3.6	44.9 ± 5.2	39.1 ± 4.9	40.1 ± 7.4
Scene					
OneError	$\mathbf{29.6 \pm 4.2}$	37.8 ± 9.3	67.7 ± 3.2	43.1 ± 8.4	64.4 ± 9.5
AvePrec	$\mathbf{82.3 \pm 4.3}$	76.2 ± 5.6	37.4 ± 1.6	74.9 ± 5.6	53.6 ± 7.8
Coverage	$\mathbf{0.6 \pm 0.2}$	0.8 ± 0.2	1.4 ± 0.2	0.8 ± 0.2	1.9 ± 0.3
HammingLoss	$\mathbf{12.8 \pm 2.3}$	15.5 ± 3.0	51.7 ± 2.1	14.6 ± 1.5	28.9 ± 4.8
RankLoss	$\mathbf{10.1 \pm 3.6}$	14.8 ± 3.5	48.1 ± 3.5	14.1 ± 5.2	41.8 ± 10.4



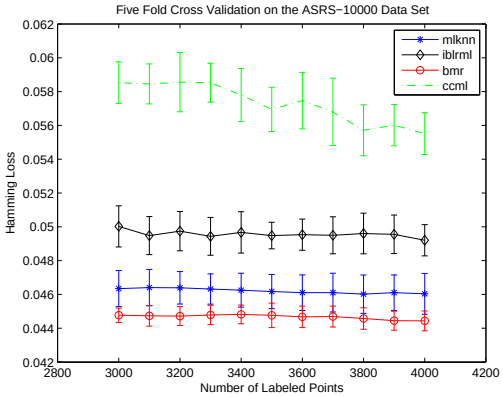
(a) One Error



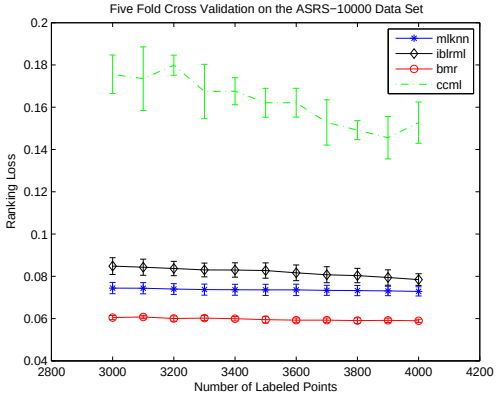
(b) Average Precision



(c) Coverage

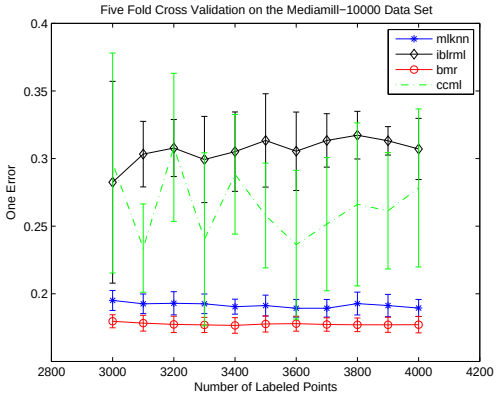


(d) Hamming Loss

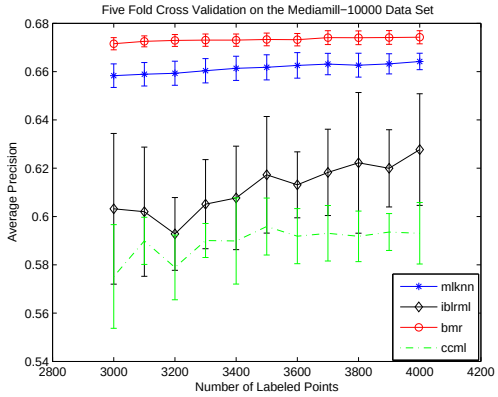


(e) Ranking Loss

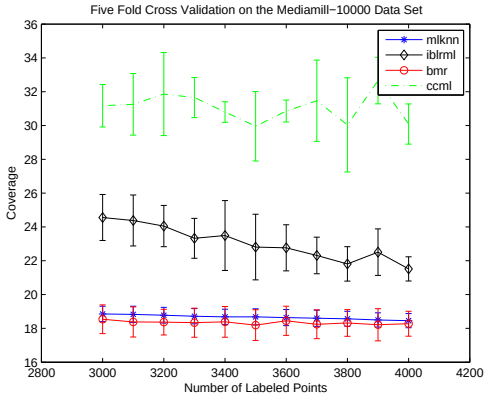
Figure 4.2: Five fold cross validation on ASRS-10000 data set. To avoid clutter we only include the top three algorithms. These plots indicate what happens when a smaller fraction of the data set is labeled. Even in this setting BMR consistently outperforms both MLKNN and IBLRML.



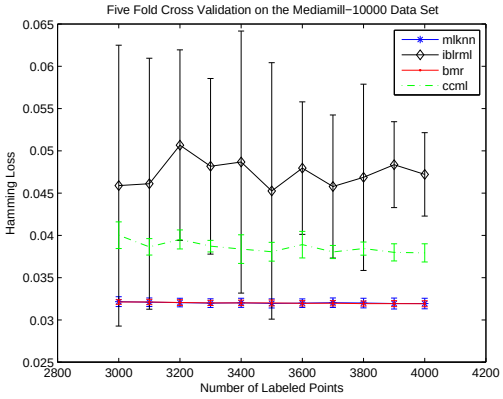
(a) One Error



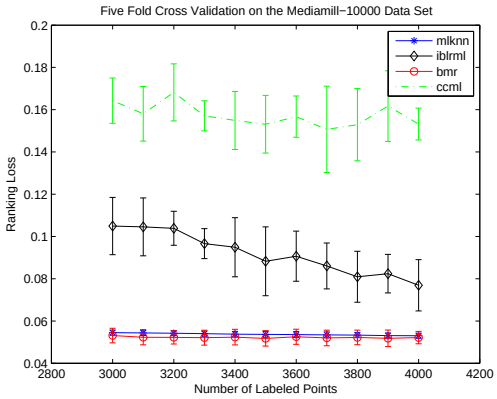
(b) Average Precision



(c) Coverage

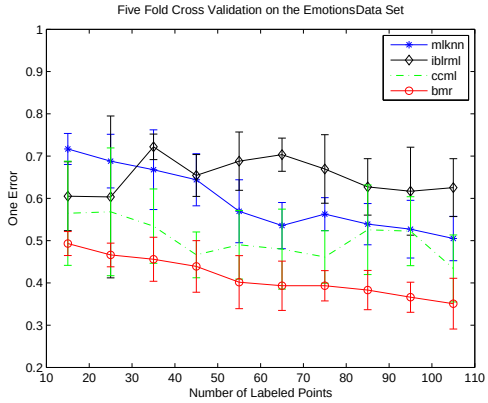


(d) Hamming Loss

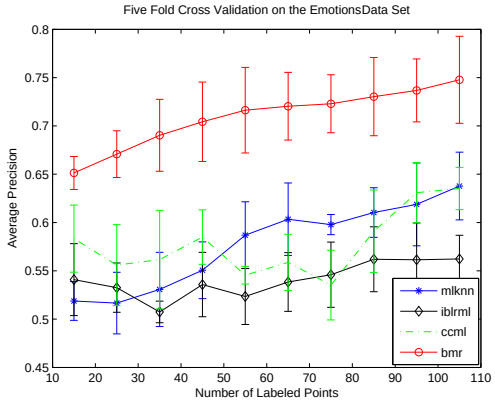


(e) Ranking Loss

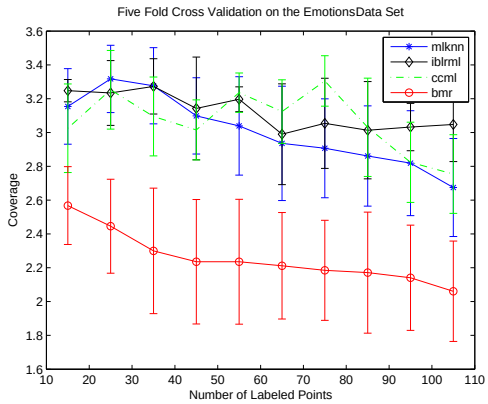
Figure 4.3: Five fold cross validation on the Mediamill-10000 data set.



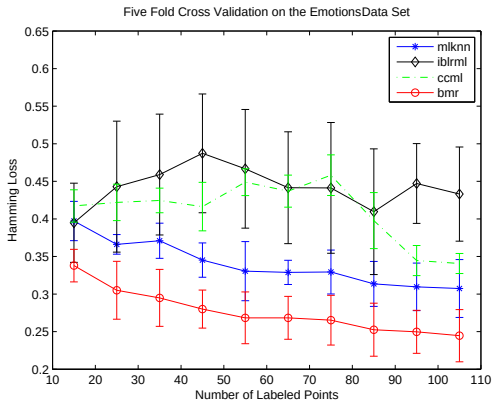
(a) One Error



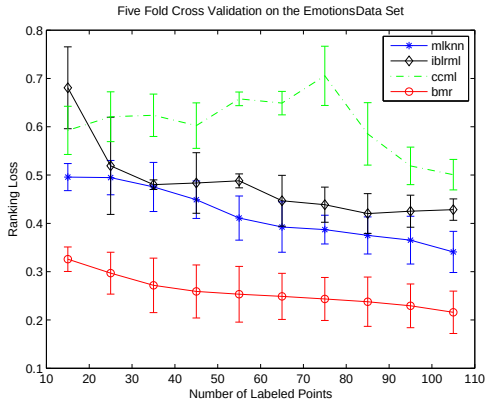
(b) Average Precision



(c) Coverage

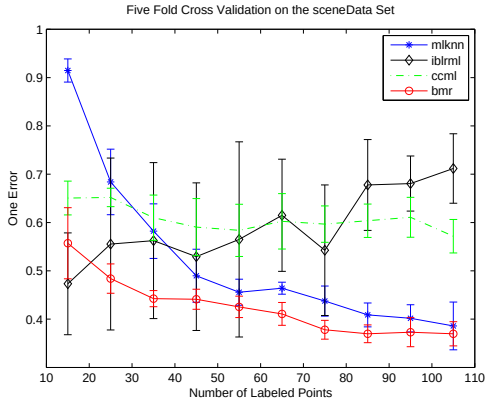


(d) Hamming Loss

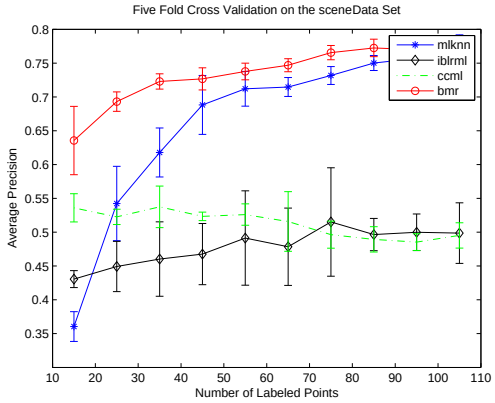


(e) Ranking Loss

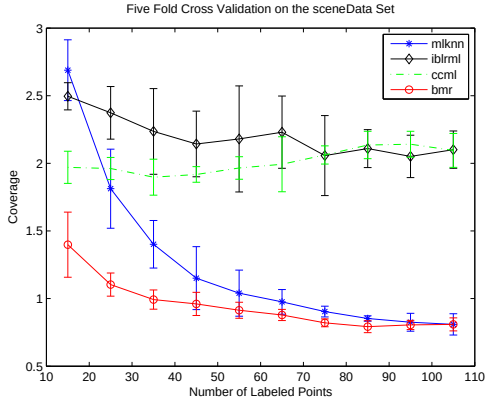
Figure 4.4: Five fold cross validation on the Emotions data set.



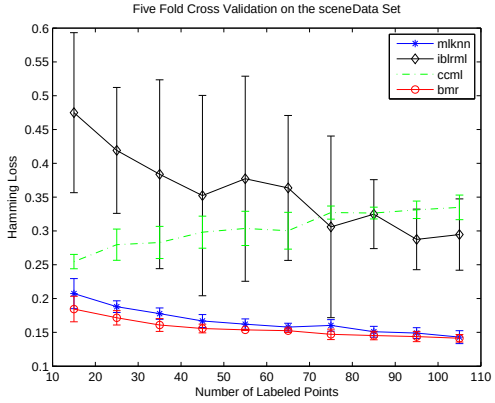
(a) One Error



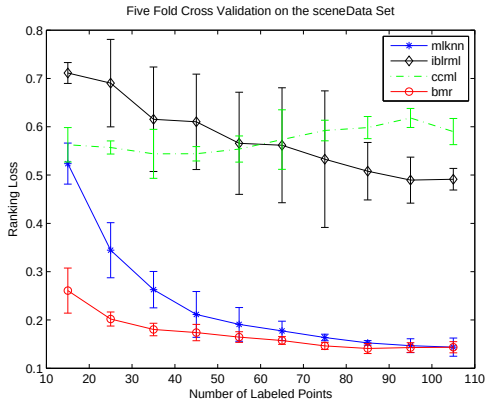
(b) Average Precision



(c) Coverage



(d) Hamming Loss



(e) Ranking Loss

Figure 4.5: Five fold cross validation on the Scene data set.

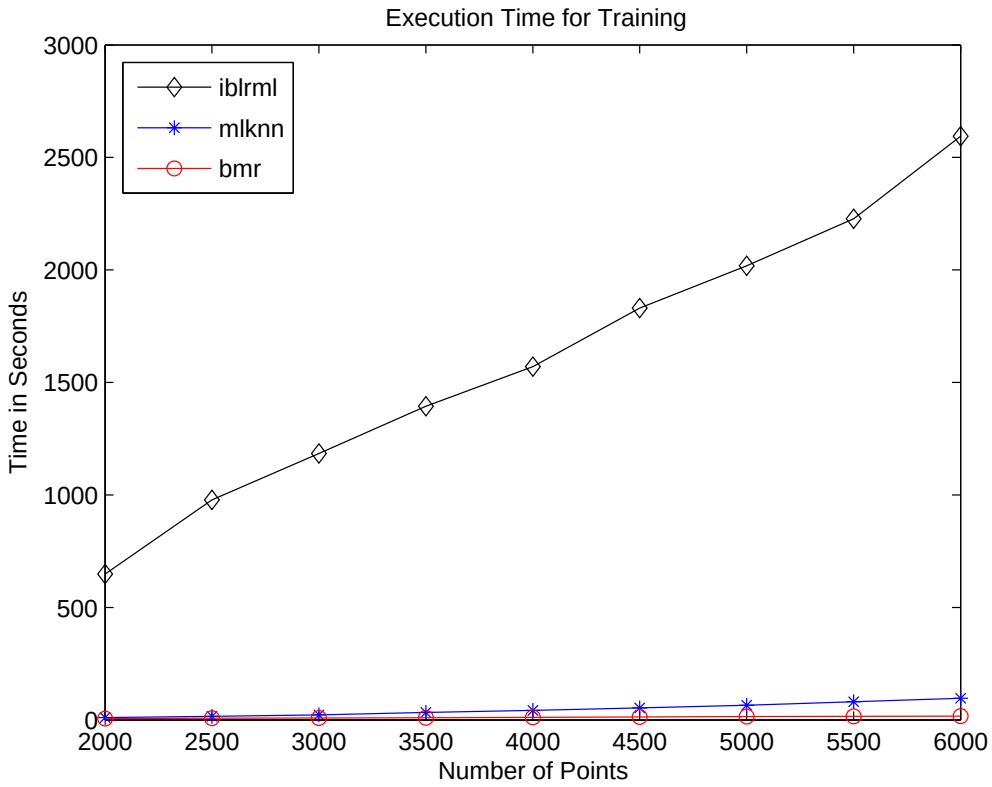


Figure 4.6: Computational time to train the model as more and more data points are considered. BMR outperforms MLKNN and IBLRML.

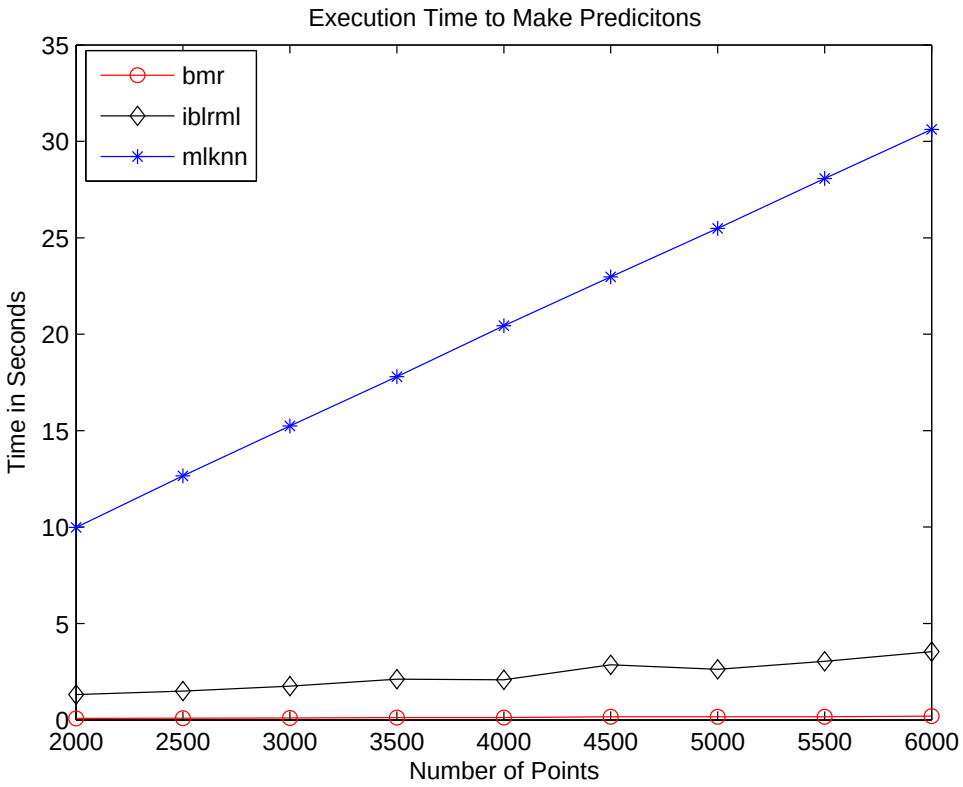


Figure 4.7: Computational time to make predictions as more and more data points are considered. BMR outperforms MLKNN and IBLRML.

Chapter 5

Probabilistic Matrix Addition

In the previous chapter we considered explicitly modeling covariances across one dimension. In this chapter we propose a model capable of modeling covariances across both rows and columns of a matrix.

5.1 Introduction

We introduce a novel approach for modeling data matrices which can simultaneously capture (nonlinear) covariance structures among rows as well as columns. For a $n \times m$ matrix X, existing approaches which consider both covariance structures can be broadly divided into two categories: Gaussian Process approaches, which suitably modify a given kernel to incorporate relational information and subsequently draw outputs (rows) i.i.d. from a single GP [82, 83, 84]; and Linear Models of Corregionalization (LMC) [8, 9], a widely used family of models from Geostatistics, which effectively flattens out the data matrix into a long vector, and uses suitable covariance structures over the vectorized form. As a result, the entries, rows, or columns of the matrix are not independent as covariances between all entries are modeled. However, the lack of (conditional) independence can lead to serious scalability problems for inference in LMCs.

In this chapter, we introduce the Probabilistic Matrix Addition (PMA) model which simultaneously considers two (nonlinear) kernels $\mathcal{K}_1$ and $\mathcal{K}_2$ corresponding to the rows and the columns of the matrix respectively. The kernels are utilized in two Gaussian Processes (GPs), from which we draw two latent matrices with independence along rows

and along columns respectively. The latent matrices from the two GPs are combined in an additive fashion to obtain the final matrix, yielding a generative model for real-valued data matrices of any size. As GPs define priors over functions $f(x)$, PMA can be viewed as a simple but non-trivial way to define priors over functions $f(x, y)$ which when instantiated lead to finite sized matrices.

Similar to LMCs, the joint distribution of PMA over the matrix entries does not factorize over entries, rows, or columns, and thus can capture intricate dependencies among the entries. Unlike LMC, PMA does not assume stationarity. It exhibits a conditional independence structure over the latent variables, which allows for fast approximate inference algorithms. We present two methods for approximate inference in PMA, respectively based on Gibbs sampling and MAP inference. The Gibbs sampler is efficient since it takes full advantage of the conditional independence structure, and precision matrices over the conditioning variables can be computed using a suitable application of the Sherman-Morrison formula. The MAP inference is obtained by solving a Sylvester equation [85, 86], where both row and column covariances play a role in determining a latent variable matrix. For parameter estimation and missing values prediction, the inference methods are used in a suitable alternating update method.

We illustrate the effectiveness of PMA on two tasks: matrix missing value prediction, where the goal is to infer multiple missing values in a given data matrix; and multi-label classification, where the goal is to predict an entire new row given a matrix which may also have missing values. For matrix missing value prediction, we compare PMA to a single GP capturing covariances either across rows or columns, Probabilistic Matrix Factorization (PMF) [69] and LMC [9]. PMA clearly outperforms a single GP, and is competitive or better than PMF and LMC. For multi-label classification, we compare PMA to three baselines including state-of-the art approaches designed specifically for multi-label classification. Across all evaluation measures and datasets, PMA consistently outperforms the other methods.

The rest of the chapter is organized as follows. In Section 6.2, we introduce PMA, discuss its properties, and contrast it with LMCs. We consider the missing value prediction problem in Section 5.3, propose two inference approaches for PMA, and present empirical evaluation of the ideas. In Section 5.4, we discuss how new rows (or columns) can be predicted using PMA, and present empirical evaluation on the multi-label prediction

problem. We briefly discuss related work in Section 5.5 and conclude in Section 6.6.

5.2 The Model

The Probabilistic Matrix Addition (PMA) model defines distributions over real valued matrices. Let X be a $n \times m$ matrix. We start by outlining a generative model for any such matrix for arbitrary n and m. Consider two Gaussian processes $G_1 \equiv GP(0, \mathcal{K}_1)$ with covariance function $\mathcal{K}_1$ corresponding to rows and $G_2 \equiv GP(0, \mathcal{K}_2)$ with covariance function $\mathcal{K}_2$ corresponding to columns. For n rows, we get the following distribution over any column $f \in \mathbb{R}^n$ from G_1:

$$p(f|G_1) = \frac{1}{(2\pi)^{D/2}|\mathcal{K}_1|^{1/2}} \exp\left(-\frac{1}{2} f^T \mathcal{K}_1^{-1} f\right) . \tag{5.1}$$

Since the matrix will have m columns, we sample $f_1, \ldots, f_m \in \mathbb{R}^n$ independently following the above distribution. The samples form the following $n \times m$ matrix F:

$$F = \begin{bmatrix} f_1 & \cdots & f_m \end{bmatrix} = \begin{bmatrix} f_1(1) & \cdots & f_m(1) \\ \vdots & \ddots & \vdots \\ f_1(n) & \cdots & f_m(n) \end{bmatrix} . \tag{5.2}$$

For m columns, we get the following distribution over any row $g \in \mathbb{R}^m$ from G_2:

$$p(g|G_2) = \frac{1}{(2\pi)^{D/2}|\mathcal{K}_2|^{1/2}} \exp\left(-\frac{1}{2} g^T \mathcal{K}_2^{-1} g\right) . \tag{5.3}$$

Since the matrix will have n rows, we sample $g_1, \ldots, g_n \in \mathbb{R}^m$ independently following the above distribution. The samples form the following $n \times m$ matrix G:

$$G = \begin{bmatrix} g_1^T \\ \vdots \\ g_n^T \end{bmatrix} = \begin{bmatrix} g_1(1) & \cdots & g_1(m) \\ \vdots & \ddots & \vdots \\ g_n(1) & \cdots & g_n(m) \end{bmatrix} . \tag{5.4}$$

Given the two random matrices F and G, we generate the $n \times m$ random matrix X as

$$X = F + G . \tag{5.5}$$

In particular, each entry of X is (Figure 5.1)

$$x_{ij} = f_j(i) + g_i(j) . \tag{5.6}$$

While the generative process for X is simple, it leads to intricate dependencies between its entries, in particular capturing (nonlinear) covariance structures along rows as well as columns.

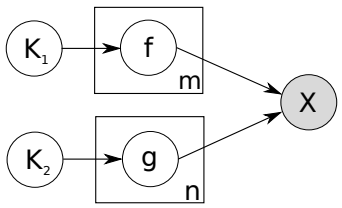


Figure 5.1: Graphical model for PMA: X is generated as the sum of F, sampled by column, and G, and sampled by row.

5.2.1 Joint and Conditional Distributions

Joint Distribution: First, we consider the joint distribution of the components of the entire matrix $X = [x_{ij}] \in \mathbb{R}^{m \times n}$. Since $f_j(i) \sim N(0, \mathcal{K}_{1,(i,i)})$ and $g_i(j) \sim N(0, \mathcal{K}_{2,(j,j)})$, the marginal distribution of x_{ij} is a univariate Gaussian: $x_{ij} \sim N(0, \mathcal{K}_{1,(i,i)} + \mathcal{K}_{2,(j,j)})$. To compute the joint covariance, first note that

$$E[f_j(i)f_j(\ell)] = K_1(i,\ell), \quad E[f_j(i)f_k(\ell)] = 0 \ . \tag{5.7}$$

Similarly,

$$E[g_i(j)g_i(k)] = K_2(j,k), \quad E[g_i(j)g_\ell(k)] = 0 \ . \tag{5.8}$$

Since $E[x_{ij}] = 0$, we have $Cov(x_{ij}, x_{\ell k}) = E[x_{ij}x_{\ell k}]$. From (**??**), for the diagonal elements we know

$$E[x_{ij}x_{ij}] = K_1(i,i) + K_2(j,j) \ .$$

For the off-diagonal entries we have

$$\begin{aligned} E[x_{ij}x_{ik}] &= E[(f_j(i) + g_i(j))(f_k(i) + g_i(k))] \\ &= E[f_j(i)f_k(i)] + E[f_j(i)g_i(k)] \\ &+ E[g_i(j)f_k(i)] + E[g_i(j)g_i(k)] \\ &= 0 + 0 + 0 + K_2(j,k) \\ &= K_2(j,k) \ . \end{aligned} \tag{5.9}$$

Similarly,

$$E[x_{ij}x_{\ell j}] = K_1(i,\ell) \ . \tag{5.10}$$

Further, note that

$$\begin{aligned} E[x_{ij}x_{\ell k}] &= E[(f_j(i) + g_i(j))(f_k(\ell) + g_\ell(k))] \\ &= E[f_j(i)f_k(\ell)] + E[f_j(i)g_\ell(k)] \\ &+ E[g_i(j)f_k(\ell)] + E[g_i(j)g_\ell(k)] \\ &= 0 + 0 + 0 + 0 \\ &= 0 \ . \end{aligned} \tag{5.11}$$

As a result, $E[x_{ij}^2] = K_1(i,i) + K_2(j,j)$, $E[x_{ij}x_{ik}] = K_2(j,k)$, $E[x_{ij}x_{\ell j}] = K_1(i,\ell)$, and $E[x_{ij}x_{\ell k}] = 0$. Putting everything together, if $vec(X)^T = [X_{(:,1)}^T, \ldots, X_{(:,m)}^T]$ denotes the vectorized version of X, then the joint distribution of $vec(X) \in \mathbb{R}^{mn}$ is a multivariate Gaussian, i.e., $vec(X) \sim N(0, \Sigma_{vec(X)})$ where

$$\Sigma_{vec(X)} = (\mathbb{I}_m \otimes \mathcal{K}_1) + (\mathcal{K}_2 \otimes \mathbb{I}_n) = \mathcal{K}_1 \oplus \mathcal{K}_2. \tag{5.12}$$

where $\oplus$ denotes the Kronecker sum [87].

Conditional Distributions: We now consider the conditional distribution of each f_{ij}, g_{ij}, and x_{ij} given the rest of the latent matrices, i.e., $F_{(-i,-j)}$ and $G_{(-i,-j)}$. Given $F_{(-i,-j)}$, $f_j(i)$ only depends on $f_j(-i)$, the other elements of f_j. Further, $f_j(i)$ is conditionally independent of $G_{(-i,-j)}$ given $F_{(-i,-j)}$. To see this, note that in the PMA graphical model there are two types of paths connecting $f_j(i)$ to elements in $G_{(-i,-j)}$: paths going through the collider x_{ij} and paths going through elements in $f_j(-i)$. The conditional d-separation between $f_j(i)$ and $G_{(-i,-j)}$ given $F_{(-i,-j)}$ stems from the fact

that there is no conditioning on the collider x_{ij} for paths of the first type and conditioning on the non-colliders (elements of $f_j(-i)$) for paths of the second type. By a similar argument, $g_i(j)$ depends only on $g_i(-j)$, the other elements of g_i, and is conditionally independent of $F_{(-i,-j)}$ given $G(-i,-j)$. As a result, we have

$$\begin{aligned} f_j(i)|(F_{(-i,-j)}, G_{(-i,-j)}) &\sim N(m_j^f(i), s_j^f(i)) \ , \\ g_i(j)|(F_{(-i,-j)}, G_{(-i,-j)}) &\sim N(m_i^g(j), s_i^g(j)) \ , \end{aligned} \tag{5.13}$$

where

$$\begin{aligned} m_j^f(i) &= \mathcal{K}_{1,(i,-i)}\mathcal{K}_{1,(-i,-i)}^{-1} f_j(-i) \ , \\ s_j^f(i) &= \mathcal{K}_{1,(i,i)} - \mathcal{K}_{1,(i,-i)}\mathcal{K}_{1,(-i,-i)}^{-1}\mathcal{K}_{1,(-i,i)} \ , \\ m_i^g(j) &= \mathcal{K}_{2,(j,-j)}\mathcal{K}_{2,(-j,-j)}^{-1} g_i(-j) \ , \\ s_i^g(j) &= \mathcal{K}_{2,(j,j)} - \mathcal{K}_{2,(j,-j)}\mathcal{K}_{2,(-j,-j)}^{-1}\mathcal{K}_{2,(-j,j)} \ . \end{aligned} \tag{5.14}$$

Since $x_{ij} = f_j(i) + g_i(j)$, we have

$$x_{ij}|(F_{(-i,-j)}, G_{(-i,-j)}) \sim N(m_{ij}, s_{ij}) \ , \tag{5.15}$$

where

$$m_{ij} = m_j^f(i) + m_i^g(j) \ , s_{ij} = s_j^f(i) + s_i^g(j) \ . \tag{5.16}$$

5.2.2 Relationship with LMCs

Linear Models of Corregionalization (LMCs) are a broad family of related models widely studied in Geostatistics [8, 9]. We compare and contrast the proposed PMA with LMCs. The proposed PMA is related to LMCs as is evident from the structure of the joint covariance matrices. However, there are important differences between the two models, including modeling assumptions as well as efficiency of inference algorithms. We briefly discuss these aspects below. First, LMCs are stationary models where the covariance depends on $(s-s')$, whereas PMA does not make such an assumption. Further, generally LMCs do not have an explicit latent variable based generative model in their specification. In particular, the statistical dependency structure of the elements of X tends to be complete. As a result, inference in LMCs typically involve one or both of the following

possible issues: (i) Inverting large covariance matrices, say $\mathbb{R}^{(mn-p)\times(mn-p)}$ matrices for p missing entries, which is computationally prohibitive, (ii) Assuming a parametric form of the variogram which greatly restricts modeling flexibility [9]. In contrast, PMA has a latent variable based model specification and the statistical dependency structure in PMA is significantly sparse. The sparsity can be exploited to develop efficient approximate inference algorithms (see Section 5.3). Since the joint distribution is Gaussian, exact inference can be done in PMA but has the same computational issues as in LMCs.

5.3 Predicting Missing Values

For missing value prediction, we are given a partially observed data matrix X. The goal is to infer the missing values based on the structure of the known observations. In this section we outline two approaches for missing value prediction, respectively based on Gibbs sampling and MAP inference. We conclude the section with an experimental evaluation of PMA for missing value prediction.

5.3.1 Gibbs Sampling

Let $\tilde{X}$ be a full matrix, where the missing values have been initialized to random values. For a given (K_1, K_2), the sampler updates the latent matrices and the missing entries in $\tilde{X}$. Since $X = F + G$, it is sufficient to sample only F or only G—we choose to sample G. If K_1 and/or K_2 is unknown, we alternate between sampling (G, X) and estimating K_1 and/or K_2.

Sampling G: Given K_1, K_2, and a full data matrix $\tilde{X}$, using Bayes rule we have

$$\begin{aligned}
&p(g_i(j)|G_{(-i,-j)}, \tilde{X}, K_1, K_2) \propto \\
&p(g_i(j)|G_{(-i,-j)}, \tilde{X}_{(-i,-j)}, K_1, K_2) p(\tilde{x}_{ij}|G, \tilde{X}_{(-i,-j)}, K_1, K_2) \\
&= p(g_i(j)|G_{(-i,-j)}, K_2) p(\tilde{x}_{ij} - g_i(j)|F_{(-i,-j)}, K_1) \ ,
\end{aligned}$$

due to conditional independence and the fact that $F_{(-i,-j)} = \tilde{X}_{(-i,-j)} - G_{(-i,-j)}$. Note that the individual distributions are univariate Gaussians as in (5.13) and (5.14). Since the product of two Gaussians is also a Gaussian, we have

$$p(g_i(j)|G_{-i,-j}, X, K_1, K_2) \propto N(g_i(j)|\mu_{ij}, \sigma_{ij}^2) \tag{5.17}$$

where

$$\mu_{ij} = \frac{m_i^g(j)s_j^f(i) + m_j^f(i)s_i^g(j)}{s_j^f(i) + s_i^g(j)} \ , \ \sigma_{ij}^2 = \frac{s_j^f(i)s_i^g(j)}{s_j^f(i) + s_i^g(j)} \ , \tag{5.18}$$

with m^g, m^f, s^g, s^f are from (5.14) with $F = X - G$.

The sampler involves several matrix inverses, but these can be computed efficiently from K_1^{-1} and K_2^{-1}. For computations involving F, instead of computing n inverses of $(n-1) \times (n-1)$ sub-matrices of K_1 (see (5.14)), we can obtain each such inverse from rank-2 modifications to K_1. Assuming that K_1^{-1} has been computed, consider the problem of computing $K^{-1}_{1,(-1,-1)} = K^{-1}_{1,(2:\,n,2:\,n)}$. According to the Sherman-Morrison formula, we have

$$(K_1 + uv^t)^{-1} = K_1^{-1} - \frac{K_1^{-1}uv^tK_1^{-1}}{1 + v^tK_1^{-1}u}, \tag{5.19}$$

where $1 + v^tK_1^{-1}u \neq 0$ and $u, v \in \mathbb{R}^n$. We construct rank-2 updates to zero out entries $K_{1,(2:\,n,1)}$ and $K_{1,(1,2:\,n)}$. This can be accomplished in two steps, first we obtain $A = K_1 + u_1v_1^T$ where $u_{1(1)} = 0$, $u_{1(2:n)} = -K_{1(2:n,1)}$, $v_{1(1)} = 1$, $v_{1(2:n)} = 0$. Then we obtain $B = A + u_2v_2^T$ where $u_{(1)} = 1$, $u_{(2:n)} = 0$, $v_{(1)} = 0$, $v_{(2:n)} = -K_{1(1,2:n)}$. Applying the Sherman-Morrison formula twice we compute $B^{-1} = (K_1 + u_1v_1^T + u_2v_2^T)^{-1}$. From basic properties of block matrices it follows: $K^{-1}_{1,(-1,-1)} = K^{-1}_{1,(2:\,n,2:\,n)} = B^{-1}_{(2:n,2:n)}$. We follow a similar computation for all the n submatrices $K_{1,(-i,-i)}$. Further, for computations involving G, we can efficiently compute the m inverses of $(m-1) \times (m-1)$ sub-matrices $K_{2,(-j,-j)}$ of K_2.

Sampling $\tilde{X}$: Missing values in X are sampled by extending the sampler and treating the missing $\tilde{x}_{ij}$ as latent variables. In particular, we sample $\tilde{x}_{ij}$ conditioned on $\tilde{X}_{(-i,-j)}$ and one of F and G. Conditioning on F, we have

$$p(\tilde{x}_{ij}|\tilde{X}_{(-i,-j)}, F, K_1, K_2) = N(x_{ij}|\bar{x}_{ij}, \zeta_{ij}) \tag{5.20}$$

where

$$\begin{aligned} \bar{x}_{ij} &= f_j(i) + K_{2,(i,-i)}K^{-1}_{2,(-i,-i)}(\tilde{x}_{-i,j} - f_j(-i)) \ , \\ \zeta_{ij} &= K_{2,(i,i)} - K_{2,(i,-i)}K^{-1}_{2,(-i,-i)}K_{2,(-i,i)} \ . \end{aligned} \tag{5.21}$$

Parameter Estimation: If K_1 and K_2 are unknown, we initialize $\hat{K}_1 \succ 0, \hat{K}_2 \succ 0$, and alternate between sampling $(G, \tilde{X})$ and estimating $(\hat{K}_1, \hat{K}_2)$. We have already outlined how to sample G and $\tilde{X}$. Let $F = \tilde{X} - G$. Then, we have $\hat{K}_1 = \frac{1}{m}\sum_{i=1}^{m} f_j f_j^T$ and $\hat{K}_2 = \frac{1}{n}\sum_{i=1}^{n} g_i g_i^T$.

5.3.2 MAP Inference

As before, we start with a full matrix $\tilde{X}$, where the missing values have been filled at random. For given gram matrices (K_1, K_2), we alternate between estimating F (or G) and $\tilde{X}$.

Estimating F: Given $\tilde{X}, K_1$, and K_2 the joint log-likelihood over (X, F) is:

$$\log p(\tilde{X}, F|K_1, K_2) = \log p(F|K_1) + \sum_{i=1}^{n} \log p(\tilde{x}_{i:}|f_{:}(i), K_2) \ .$$

For a given $\tilde{X}$, the MAP F can be obtained by maximizing the joint log-likelihood, or equivalently minimizing

$$\sum_{i=1}^{n} (\tilde{x}_i - F^T e_i^n)^T K_2^{-1} (\tilde{x}_i - F^T e_i^n) + \sum_{j=1}^{m} e_j^{mT} F^T K_1^{-1} F e_j^m \ ,$$

where $e_i^n \in \mathbb{R}^n, e_j^m \in \mathbb{R}^m$ are vectors of all zeros with the i^{th} and j^{th} position set to one respectively. A direct calculation shows that the solution has to satisfy the following Sylvester equation

$$FK_2 + K_1 F = K_1 \tilde{X} \ . \tag{5.22}$$

A solution to the Sylvester equation exists if and only if no eigenvalue of K_1 is equal to the negative of an eigenvalue of K_2 [85]. Since both K_1 and K_2 are positive definite, the condition is satisfied, and the solution can be obtained by standard methods [85, 86].

Estimating $\tilde{X}$: We iteratively update the originally missing entries $\tilde{x}_{ij}$ based on the mode of the distribution $p(\tilde{x}_{ij}|F, \tilde{X}_{(-i,-j)}, K_1, K_2)$, given by

$$\tilde{x}_{ij}^{new} = f_j(i) + \mathcal{K}_{2,(j,-j)} \mathcal{K}_{2,(-j,-j)}^{-1} (\tilde{x}_{-i,j} - f_j(-i)) \ . \tag{5.23}$$

Note that the expression is similar to (5.21), where we sample from the corresponding distribution.

Parameter Estimation: We initialize $\hat{K}_1 \succ 0$, $\hat{K}_2 \succ 0$, and the missing values of X randomly. Then, we alternate between updating $(\tilde{X}, F)$, and estimating $\hat{K}_1$, $\hat{K}_2$. We have already discussed updates for $(\tilde{X}, F)$. Let $G = \tilde{X} - F$. Then $\hat{K}_1 = \frac{1}{m}\sum_{j=1}^{m} f_j f_j^T$ and $\hat{K}_2 = \frac{1}{n}\sum_{i=1}^{n} g_i g_i^T$.

5.3.3 Experimental Evaluation

We report results from two sets of experiments for missing value prediction. In the first set, we compare PMA to Gaussian Process regression (GPR) on simulated datasets. In the second set, we compare PMA to other algorithms, including GPR, Probabilistic Matrix Factorization (PMF) [69], and intrinsic LMC (I-LMC) on benchmark datasets.

Datasets and Evaluation: For the first set, we use PMA to generate artificial datasets, of size 50×20 and 50×50. Evaluation is done using mean square error of the predicted values. For the second set, we use two multi-label classification datasets—Emotions [80] and Scene [79]. In multi-label classification, for n points and m classes, class memberships are represented as $n \times m$ binary matrix B. We consider a truncated log-odds matrix X, with $x_{ij} = c$ if $b_{ij} = 1$, and $x_{ij} = -c$ if $b_{ij} = 0$. For the experiments, certain entries x_{ij} are assumed to be missing. Evaluation is done using class membership prediction accuracy based on $\text{sign}(\hat{x}_{ij})$.

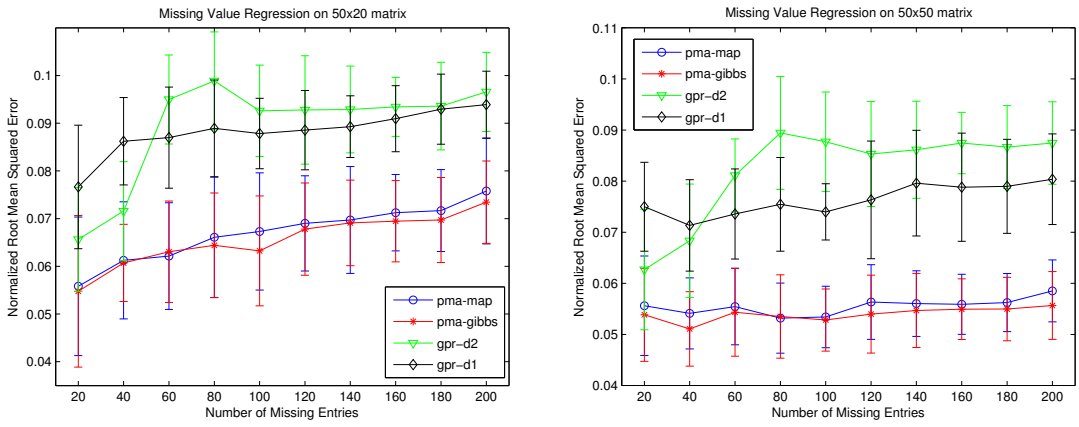
Methodology: All experiments were conducted using five-fold cross validation. For the first set, both K_1 and K_2 are assumed to be unknown, and are estimated from data. For the second set, K_1 is instantiated using the RBF kernel function $\mathcal{K}_1$ based on feature vectors of the data objects, and K_2 is estimated from data.

Algorithms: For the first set, we compare PMA to GPR. GPR-D1 treats rows as data points, while GPR-D2 treats columns as data points. We utilize one implementation of PMA based on MAP inferece (PMA-MAP) and one based on Gibbs sampling (PMA-GIBBS). For the second set, we compare six different algorithms: GPR, PMA-MAP, PMA-GIBBS, PMA-EXACT, PMF, I-LMC and BMR. PMA-EXACT performs exact prediction by flattening out the matrix X into a vector, assuming a covariance matrix of the form $\mathcal{K}_1 \oplus \mathcal{K}_2$. I-LMC corresponds to intrinsic LMC in the prediction step, whereby we utilize a covariance matrix of the form $\mathcal{K}_1 \otimes \mathcal{K}_2$. For both PMA-EXACT and I-LMC we use a provided kernel $\mathcal{K}_1$ and K_2 as obtained by Gibbs sampling. The purpose of comparing the latter two algorithms is to see whether PMA suffers by assuming a sparser dependency structure. BMR is the Bayesian Multivariate Regression approach proposed in the previous chapter.

Performance: The results for the first set involving simulated data is in Figure 5.3. PMA performs better than both GPR-D1 and GPR-D2, suggesting that there is a clear benefit in modeling both $\mathcal{K}_1$ and $\mathcal{K}_2$. While not all matrices will necessarily have relevant

correlation structure across both dimensions, when this is the case, PMA appears to do better. PMA-GIBBS and PMA-MAP perform similarly, with PMA-GIBBS appearing slightly better.

The results for the second set involving benchmark datasets is in Table 5.1. We make the following observations: (i) PMA clearly outperforms a GPR, illustrating the value in modeling correlations across both rows and columns; (ii) The differences in performance between PMA-GIBBS, PMA-EXACT and I-LMC are negligible. The results indicate that PMA does not suffer by assuming a sparser dependency structure. Further, PMA-GIBBS is fairly accurate when compared to PMA-EXACT, which can be computationally prohibitive for large datasets; (iii) PMA-GIBBS appears to perform slightly but consistently better than PMA-MAP; (iv) PMA is competitive compared to PMF and BMR. On Emotions, PMF and BMR appear slightly better. On Scene, PMA is significantly better, suggesting a clear advantage for certain datasets.



(a) 50x20 Matrix (b) 50x50 Matrix

Figure 5.2: Five fold cross validation on two artificially created data sets. PMA clearly benefits from modeling both covariance structures. The GPR both across rows (GPR-D1) and columns (GPR-D2) is weaker. Gibbs sampling appears to do slightly better compared to MAP, when it comes to inference in PMA.

	10%	15%	20%	25%	30%
Emotions					
GPR	32.3 ± 5.9	33.1 ± 4.7	32.6 ± 5.0	34.6 ± 2.3	35.3 ± 2.3
PMA-MAP	23.9 ± 6.5	25.3 ± 3.8	26.9 ± 4.4	29.7 ± 4.8	30.8 ± 4.7
PMA-GIBBS	23.3 ± 5.3	24.8 ± 3.2	$\mathbf{25.1 \pm 3.8}$	$\mathbf{27.2 \pm 3.9}$	28.0 ± 4.0
PMA-EXACT	$\mathbf{19.7 \pm 4.9}$	$\mathbf{23.6 \pm 6.9}$	25.8 ± 4.0	27.3 ± 5.4	27.9 ± 4.1
I-LMC	$\mathbf{20.3 \pm 4.6}$	25.1 ± 5.9	25.7 ± 3.7	27.6 ± 4.5	$\mathbf{27.8 \pm 3.8}$
PMF	21.8 ± 5.0	$\mathbf{22.6 \pm 2.4}$	$\mathbf{24.6 \pm 3.0}$	$\mathbf{26.3 \pm 1.6}$	$\mathbf{26.0 \pm 3.7}$
BMR	$\mathbf{21.0 \pm 5.3}$	$\mathbf{22.5 \pm 5.9}$	$\mathbf{24.5 \pm 2.1}$	$\mathbf{25.6 \pm 2.1}$	$\mathbf{25.6 \pm 3.5}$
Scene					
GPR	14.7 ± 1.7	34.5 ± 8.0	17.2 ± 2.1	17.4 ± 1.7	18.0 ± 2.1
PMA-MAP	11.9 ± 1.0	13.6 ± 2.5	13.8 ± 2.7	13.9 ± 3.2	14.8 ± 1.5
PMA-GIBBS	$\mathbf{10.3 \pm 1.4}$	$\mathbf{10.9 \pm 2.6}$	$\mathbf{11.1 \pm 1.8}$	$\mathbf{11.3 \pm 2.1}$	$\mathbf{12.3 \pm 1.2}$
PMA-EXACT	$\mathbf{10.4 \pm 1.0}$	$\mathbf{11.0 \pm 1.0}$	$\mathbf{11.6 \pm 1.8}$	$\mathbf{11.9 \pm 1.2}$	$\mathbf{12.5 \pm 2.3}$
I-LMC	10.4 ± 1.0	$\mathbf{10.9 \pm 1.0}$	$\mathbf{11.8 \pm 1.7}$	$\mathbf{11.8 \pm 1.2}$	$\mathbf{12.9 \pm 2.6}$
PMF	$\mathbf{9.2 \pm 2.2}$	13.8 ± 3.0	16.1 ± 3.4	18.5 ± 2.8	20.1 ± 3.0
BMR	12.9 ± 1.7	13.0 ± 0.9	13.3 ± 2.0	14.1 ± 1.6	15.6 ± 2.5

Table 5.1: Error rates for recovering missing labels obtained using five-fold cross validation on the Emotions and Scene data sets. Performance of GPR, PMA-MAP, PMA-GIBBS, PMA-EXACT, I-LMC, PMF and BMR is evaluated while an increasing percentage of labels are missing in the training data. Missing Labels are randomly selected. The error rates reflect the percentage of missing labels incorrectly recovered.

5.4 Predicting New Rows

We consider the problem of predicting a new row in the data matrix X assuming that $\mathcal{K}_1$ is a known kernel function. The motivation comes from multi-label classification, where a new row translates to all labels for a data point not encountered before. The methods developed can also be applied to predict new columns assuming $\mathcal{K}_2$ is a known kernel function. As before, we outline two inference approaches based on Gibbs sampling and MAP inference respectively. We evaluate PMA for new row prediction in the task of multi-label classification.

We first focus on initializing the new row $x_{(n+1)\cdot}$ of X. Since F and G do not have values for this new row, one needs to get suitable extensions for F and G. Since F has dependencies along columns, for each column j, we obtain the MAP estimate $f_j(n+1)$

using GP regression and $\mathcal{K}_1$ yielding the extended matrix $\tilde{F} \in \mathbb{R}^{(n+1)\times m}$. Since G does not have dependencies along columns, we sample a new row $g_{n+1}^T \sim GP(0, \mathcal{K}_2)$ yielding the extended $\tilde{G} \in \mathbb{R}^{(n+1)\times m}$.

For Gibbs Sampling, we obtain the initial extended matrix as $\tilde{X} = \tilde{F} + \tilde{G}$. Then we proceed as in Section 5.3.1, while treating the entire last row of $\tilde{X}$ as latent, in addition to $\tilde{G}$ and any other missing entries in $\tilde{X}$. For MAP inference, since g_{n+1}^T is zero mean, the $(n+1)^{st}$ row of $\tilde{F}$ serves directly as an estimate for the new row $x_{(n+1):}$. In either setting, if K_2 is unknown, we alternate between sampling/estimating $\tilde{X}$ and estimating K_2.

5.4.1 Experimental Evaluation

We compare the performance of PMA (PMA-GIBBS) to existing state-of-the-art methods for multi-label classification on a number of benchmark data sets. We use the Scene [79], Emotions [80],Image [88] and YeastProt [89] datasets for evaluation. In table 5.2 a summary is provided for all of the data sets used. As before, we use truncated log-odds during learning, and the sign of the predicted score for evaluation.

Algorithms and Methodology: We evaluate PMA-GIBBS against four multi-label classification algorithms. For PMA-GIBBS, we assume that $\mathcal{K}_1$ is an RBF Kernel over the points, where its parameters are estimated using cross validation, and K_2 is unknown and estimated from the data. The inputs into $\mathcal{K}_1$ are given by feature vectors of the data points. As a baseline, we consider one-vs-rest SVM as a multi-label classifier, which we refer to as MLSVM. We also consider two state-of-the-art approaches for multi-label learning: Multi Label K-nearest Neighbors (MLKNN) [42], a method which applies the k-nearest neighbor idea to the multi-label setting; and Instance Based Learning by Logistic Regression (IBLR) [74], where features are first transformed to incorporate label information from local neighborhoods prior to applying logistic regression. In all multi-label experiments, we utilize an RBF Kernel in PMA, where the parameter σ is chosen by cross-validation. We also compare performance to Bayesian Multivariate Regression (BMR), developed in the previous chapter.

We also consider the setting where the training set has partial labels, i.e., missing entries in X. While PMA and BMR can utilize partial labels, the other algorithms cannot. Hence, we construct a reduced training set discarding points with partial labels.

The corresponding models are called PMA-GIBBS-D, BMR-D MLKNN-D, IBLRML-D and MLSVM-D.

Evaluation Measures: We evaluated multi-label classification performance using three different measures: one error, precision, and ranking loss. Let $g(x, l)$ denote a real-valued function which assigns a score to label l for data point x, such that a larger score is considered better. Further, let L_x denote a set of true labels associated with x.

1) *One error* evaluates how frequently the top ranked predicted label is not among the true labels. If $\mathbb{1}[\cdot]$ denotes the indicator function, we have:

$$OneError(g) = \frac{1}{D}\sum_{d=1}^{D} \mathbb{1}\left[\text{argmax}_{l\in L}\, g(x_d, l) \notin L_{x_d}\right] \ .$$

2) For true labels $l \in L_x$, *average precision* evaluates the fraction of labels in L_x that rank at least as high as l according to the scoring rule g on average. For any data point x and any label $l \in L_x$, let $\mathcal{R}(x, l) = \{l' \in L_x | rank_g(x, l') \leq rank_g(x, l)\}$, where the ranking is among all possible labels. Then, average precision is:

$$AvePrec(g) = \frac{1}{D}\sum_{d=1}^{D} \frac{1}{|L_{x_d}|} \sum_{l\in L_{x_d}} \frac{|\mathcal{R}(x_d, l)|}{rank_g(x, l)} \ .$$

3) Ranking loss reflects the average number of labels that are reversely ordered for a given instance. Let $\mathcal{T}(x_d) = \{(l_1, l_2) \mid g(x_d, l_1) \leq g(x_d, l_2),\ (y_1, y_2) \in L_{x_d} \times \bar{L}_{x_d}\}$, where $\bar{L}_{x_d}$ denotes the complement of L_{x_d}. Ranking loss is defined as:

$$RankLoss(g) = \frac{1}{D}\sum_{d=1}^{D} \frac{|\mathcal{T}(x_d)|}{|\bar{L}_{x_d}||L_{x_d}|} \ .$$

Thus, for one error and ranking loss, lower is better; for average precision, higher is better. We evaluate all multi-label algorithms using all performance measures and five fold cross validation.

Performance: We evaluate performance on both multi-label datasets by considering an increasing number of labeled points. As seen in Figure 5.3, PMA outperforms the other four methods on all data sets and for all three performance measures, and the improvements are in almost all cases significant. Further, all four methods designed specifically for multi-label classification tend to outperform MLSVMs. Compared to

BMR, PMA is slightly better on the Emotions data set and it is significantly better on the remaining data sets.

We also tested the prediction performance when missing labels are present in the training data. As seen in Table 5.3, PMA-GIBBS outperforms the other models due to its ability to leverage partially labeled data. Among algorithms which discard points with partial labels, PMA-GIBBS-D outperforms the others.

Data Set	**Points**	**Features**	**Labels**
Scene	2407	294	6
Emotions	593	72	6
Image	2000	135	5
YeastProt	2000	215	138

Table 5.2: Data sets used in multi-label classification

	OneError	AvePrec	Coverage
PMA-GIBBS	$\mathbf{29.7 \pm 4.2}$	$\mathbf{82.3 \pm 2.7}$	$\mathbf{10.6 \pm 2.3}$
BMR	33.3 ± 4.9	79.9 ± 2.4	11.6 ± 2.0
PMA-GIBBS-D	51.1 ± 7.5	67.5 ± 4.8	22.4 ± 3.7
BMR-D	54.7 ± 6.7	64.5 ± 4.6	26.1 ± 3.4
MLKNN-D	70.5 ± 2.0	46.3 ± 4.3	23.7 ± 8.1
IBLRML-D	61.9 ± 8.9	36.9 ± 3.9	54.8 ± 3.9
MLSVM-D	87.9 ± 2.0	40.9 ± 1.6	83.1 ± 1.4

Table 5.3: Five fold cross validation on the Scene data set with 25% of label entries missing. PMA-GIBBS utilizes all available data while training. PMA-GIBBS-D, BMR-D, MLKNN-D,IBLRML-D and MLSVM-D discard data points with missing label entries in the training stage.

5.5 Related Work

In this section we review work previously done in extending Gaussian Processes to capture correlations across outputs. We also review work done in multi-label classification.

One of the most popular approaches to capture correlations among outputs of a GP

is to utilize convolution processes (CPs) [90, 84]. In CPs each output is represented as the convolution of a smoothing kernel and a latent function. The outputs of the convolutions are then modeled by a single Gaussian Process [91], whereby data points are still assumed to be drawn i.i.d. In our model the i.i.d. assumption is not made. From a convolution stand point our model also differs. Let $P_f(F)$ denote the PMA prior over matrices F, Let $P_g(G)$ denote a prior over matrices G. In PMA we have $P(X) = \int_F P_g(X - F)P_f(F)dF = (P_f * P_g)(X)$, where P_f and P_g are GP-based priors over matrices. Please note that in PMA this convolution view point does not hold for single rows or columns of X, since P_f is defined over columns and P_g over rows.

In [92] a model is proposed which is capable of incorporating relational side information in form of a graph, resulting in correlated outputs of a GP. Unlike our approach this method does not model correlations on two different levels explicitly. In [82] a model is proposed which assumes latent functions to be the sum of two random variables, one of which contains relational side information. The resulting model, unlike PMA, is represented by a single GP with a modified Kernel, from which points are drawn i.i.d. In [83] an approach is proposed that combines ideas from [92] and [82]. Latent variables are assumed to be a sum of multiple random variables which encode relational information, whereby the aggregate latent variables are representable as outputs of a single GP (see discussion after equation (11) in [83]). Further in [83] unlike in PMA links are modeled explicitly.

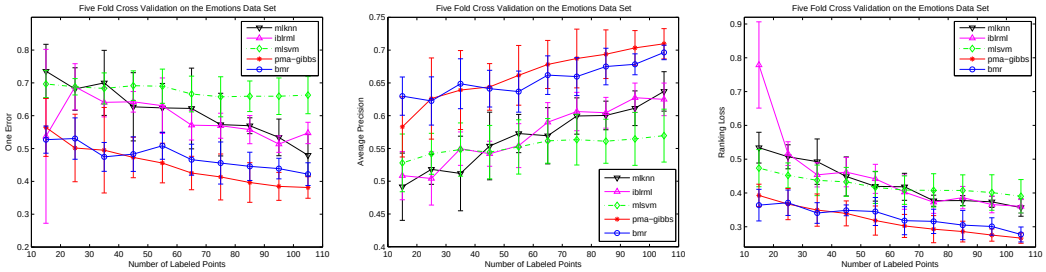
When it comes to multi-label classification the state-of-the art is comprised of methods which attempt to capture correlations among labels. In recent years several such methods have been proposed. In [70], a maximum margin approach is proposed which minimizes the ranking loss. While the ability to handle kernels is important in several domains, most existing approaches do not have a natural way of providing a direct uncertainty quantification. A number of probabilistic models have also been proposed for multi-label classification. In [68], a mixture model is proposed for text classification. More recently, in [72], a fully Bayesian model was proposed based on sparse and infinite canonical correlation analysis. Most of the probabilistic models do not have a good way of incorporating kernels, and if they do as in [73], they do not model correlations among labels explicitly.

In [42], label statistics from neighborhoods are used to build a Bayesian classifier.

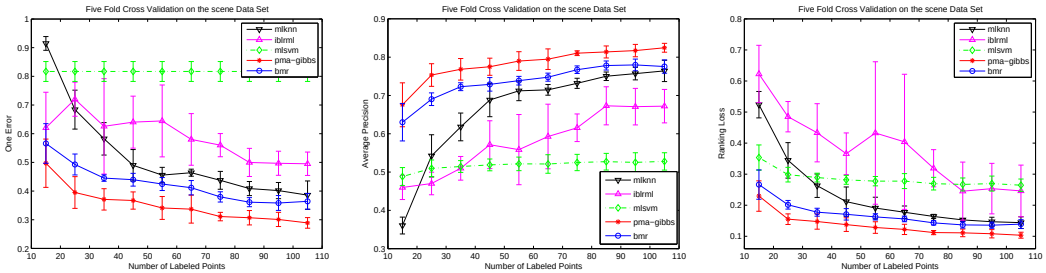
In [74], features are constructed based on label information from neighborhoods and subsequently used in logistic regression. Our model stands out as unique as it is capable of incorporating a kernel, providing direct uncertainty quantification and modeling correlation structure among labels explicitly at the same time.

5.6 Conclusions

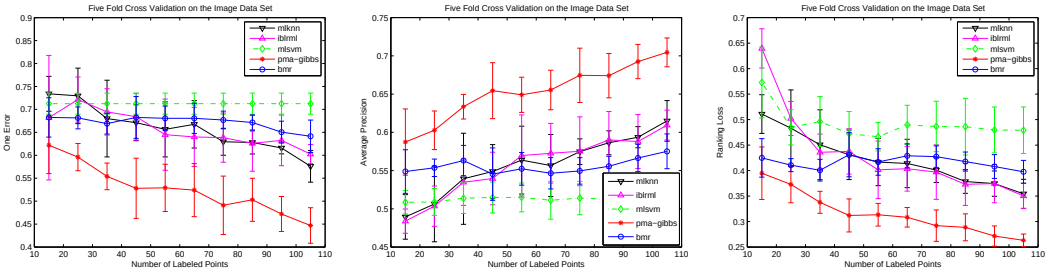
We have introduced a novel model for matrix data analysis capable of capturing correlations among rows and columns simultaneously. PMA has sparse statistical dependency structures yielding fast approximate inference algorithms. We have presented preliminary experiments demonstrating the advantage of PMA over single GPs for matrix analysis, as well as its ability to handle missing data. The ability of PMA to capture correlations along rows and columns simultaneously appears especially beneficial in domains such as multi-label classification. Our empirical evaluation shows that PMA can significantly outperform some of the existing multi-label classification algorithms. Further, PMA can readily be extended to higher order structures such as tensors which we plan to investigate in future work.



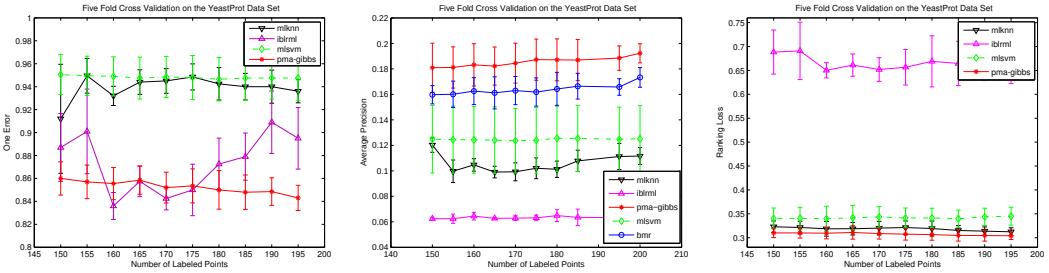
(a) Emotions - OneError (b) Emotions - AvePrec (c) Emotions - Ranking Loss



(d) Scene - OneError (e) Scene - AvePrec (f) Scene - Ranking Loss



(g) Image - OneError (h) Image - AvePrec (i) Image - Ranking Loss



(j) YeastProt - OneError (k) YeastProt - AvePrec (l) YeastProt - Ranking Loss

Figure 5.3: Five fold cross validation on the Emotions, Yeast and Image data sets using three evaluation measures. PMA consistently outperforms the other methods on all datasets according to all evaluation measures.

Chapter 6

Gaussian Process Topic Models

Having introduced PMA in the previous chapter, we propose a novel topic model capable of modeling the covariance across topics while incorporating a kernel among documents. This is accomplished by utilizing PMA as a prior.

6.1 Introduction

In recent years, significant progress has been made in analyzing text documents using topic models. Statistical topic models such as Latent Dirichlet Allocation (LDA) [93] and its variants have proven useful and effective. Such topic models allow mixed memberships of documents to several topics, where a topic is represented as a distribution over words.

In LDA, the topic proportions for each document are drawn from a Dirichlet distribution. As a consequence, LDA does not have the flexibility of modeling correlations among the topics. Correlated Topic Models (CTMs) [7] were proposed to address this issue. Instead of a Dirichlet prior, CTMs use a multi-variate normal distribution with a covariance parameter and map samples from the normal distribution to the topic simplex using a mean parameterization. The prior model assumes a fixed mean and covariance parameter for the entire corpus, and the corpus is used to learn these parameters. Correlations between topics are captured by the resulting covariance matrix.

Frequently one might have additional information about a text corpus, possibly in the form of additional features/structures, labels, one or more weighted graphs, etc. For

the purposes of this chapter, we assume that such additional information can be captured by a suitable kernel defined over the documents. While there has been recent work on incorporating link structure among documents [94], existing topic models, including CTM, are unable to leverage such information in form of a kernel. In this chapter, we propose Gaussian Process Topic Models (GPTMs) which can capture correlations among topics as well as leverage known similarities among documents using a kernel. GPTMs can be considered a generalization of CTMs using ideas from Gaussian Process (GP) embedding and regression. Given a kernel among documents, GPTM defines a Gaussian Process mapping from a suitable document space into the topic space. While topic proportions for all documents in CTM are generated from a single mean, the topic proportions in GPTM are generated from different means. The location of the means for any document is determined by the Gaussian Process mapping.

Gaussian Processes (GPs) define distributions over function spaces and have been successfully used for non-linear regression, classification and embedding [95, 96, 5]. The Gaussian Process Latent Variable Model (GPLVM) [5] is a probabilistic embedding method which utilizes a GP mapping from the embedded space to the data space. While GPLVM is powerful non-linear embedding method, current literature does not have effective models for combining kernel based non-linear embedding models such as GPLVM with probabilistic topic models such as CTM. One can obtain embeddings from either family of methods—from LDA/CTM based on the topic structure or from GPLVM using the kernel and observed features. The proposed GPTM can systematically leverage both types of information and obtain an embedding based on both the topic structure and the kernel.

We propose suitable approximate inference algorithms for learning GPTMs and making predictions on a test set. The proposed inference algorithm marginalizes the latent variables in the topic model and maximizes over the latent variables in the embedding, so we obtain one good embedding. During learning, GPTMs work with two different positive definite matrices—a topic covariance matrix over the topics and a document kernel matrix over the documents. Our analysis shows that the two matrices get integrated in an elegant manner to determine the final embedding. In particular, we obtain a Sylvester equation involving both matrices whose solution gives the final embedding. While Sylvester equations have been extensively studied in control theory, to the best

of our knowledge, their usage in the context of topic models is novel.

The rest of the chapter is organized as follows. We introduce GPTMs in Section 6.2 and discuss learning GPTMs in Section 6.3. We present experimental results in Section 6.4, provide a discussion on our model in Section 6.5 and conclude in Section 6.6.

6.2 The Model

Correlated Topic Models (CTMs) [7] are an important recent advance in the realm of topic models [93, 97]. CTMs have the ability to capture correlation among topics. However, CTMs were not designed to capture any additional information regarding the documents, possibly in the form of a kernel over the documents. In this section, we introduce Gaussian Process Topic Models (GPTMs) which are a systematic generalization of CTMs capable of incorporating knowledge from a kernel over the documents.

The key difference between CTM and the proposed GPTM is how the model samples mixed memberships over topics for each document. In CTMs, one samples $\eta \in \mathbb{R}^K$ from a multivariate Gaussian $N(\mu, \Sigma)$ and maps η to the topic simplex using a mean parameterization [7]. As a result, $E[\eta] = \mu$, i.e., apriori all documents have the same mixing proportions in expectation. In GPTMs, apriori all documents have different mixing proportions in expectation. The mixing proportions are derived from the kernel over the documents and, intuitively, similar documents according to the kernel have similar mixing proportions.

Given a kernel function $\mathcal{K}$ over documents, the corresponding GP defines a distribution over functions over all documents. For a set of D documents, we get a distribution over $f \in \mathbb{R}^D$ given by

$$p(f|\mathcal{K}) = \frac{1}{(2\pi)^{D/2}|\mathcal{K}|^{1/2}} \exp\left(-\frac{1}{2} f^T \mathcal{K}^{-1} f\right) . \tag{6.1}$$

Assuming there are K topics, we independently sample $f_1, \ldots, f_K \in \mathbb{R}^D$ from the above distribution, and construct a $K \times D$ matrix F, whose i^{th} row is f_i^T. Hence $p(F|\mathcal{K}) = \prod_{i=1}^K p(f_i|\mathcal{K})$. Now, for each document $d = 1, \ldots, D$ we generate $\eta_d \in \mathbb{R}^K$ following

$$p(\eta_d|\mu_d, \Sigma) \sim \mathcal{N}(\eta|\mu_d, \Sigma) \tag{6.2}$$

where Σ denotes a $K \times K$ topic covariance matrix and $\mu_d = Fe_d \in \mathbb{R}^K$, where $e_d \in \mathbb{R}^D$ represents the all zeros vector with only the d^{th} entry 1. Thus, $\mu_d \in \mathbb{R}^K$ is the d^{th} column

of F. Each η_d is then mapped to the topic simplex using the mean parameterization: $\theta(\eta_d) = \frac{\exp(\eta_d)}{\sum_i \exp(\eta_d(i))}$. Since the rows of F were drawn independently from the GP with kernel $\mathcal{K}$, similar documents according to the kernel will implicitly have similar μ_d, and hence similar topic proportions apriori. Thus, the GP prior captures global relationships between documents in determining the apriori mixed memberships.

The entire generative model (Figure 6.1) can be specified as follows:

1. Draw $F|\mathcal{K} \sim p(F|\mathcal{K}) = \prod_i \mathcal{N}(f_i|0, \mathcal{K})$.
2. For each document $d = 1, \ldots, D$:
   (a) $\eta_d|F, \Sigma \sim \mathcal{N}(\eta_d|Fe_d, \Sigma)$.
   (b) For each word $w_n, n = 1, \ldots, N_d$:
      i. Draw a topic $z_n|\eta_d \sim \text{Discrete}(\theta(\eta_d))$.
      ii. Draw a word $w_n|z_n, \beta_{1:K} \sim \text{Discrete}(\beta_{z_n})$.

The joint probability of all observed and latent variables in GPTM is given by:

$$p(w, z, \eta, F|\mathcal{K}, \Sigma, \beta) = \prod_{i=1}^{K} p(f_i|\mathcal{K}) \prod_{d=1}^{D} p(\eta_d|Fe_d, \Sigma) \prod_{n=1}^{N_d} p(z_n|\eta_d)p(w_n|z_n, \beta) \ . \tag{6.3}$$

The proposed GPTM is different from both CTM as well as GPLVM, while drawing from the strengths of both of these models. Unlike CTM, the apriori topic proportions of the documents are different and the difference is based on the kernel $\mathcal{K}$. Unlike GPLVM, GPTM takes topic structure into account. The final embedding will be based on both the kernel as well as the structure of the documents as determined by the topic model. As a result, GPTM leverages the strength of both topic models as well as kernel methods, in particular CTMs and GPs.

6.3 Learning GPTMs

Exact inference in GPTMs is computationally intractable. In this section, we first explain why some standard approaches to approximate inference may not be desirable

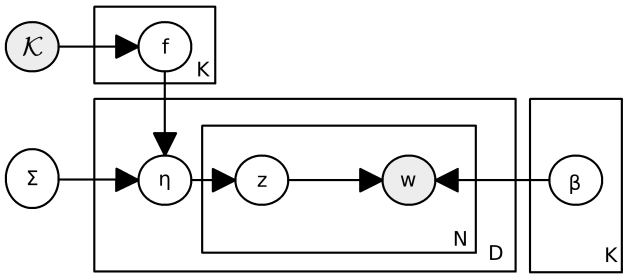


Figure 6.1: Gaussian Process Topic Model

and then outline a somewhat non-standard approach for doing approximate inference in GPTMs. The log-likelihood of the observed words w given $(\mathcal{K}, \Sigma, \beta)$ is given by:

$$\log p(w|\mathcal{K}, \Sigma, \beta) = \log E_{(F,\eta,z)}[p(w, z, \eta, F|\mathcal{K}, \Sigma, \beta)] \ , \tag{6.4}$$

where the expectation is over the distribution on the latent variables (F, η, z). In several GP-based models [96], one can integrate over all functions f which translates to the distribution over F in GPTM. Focussing on the terms involving F in the joint distribution, we have

$$\prod_{i=1}^{K} p(f_i|\mathcal{K}) \prod_{d=1}^{D} p(\eta_d|Fe_d, \Sigma) = \frac{1}{|\mathcal{K}|^{k/2}|\Sigma|^{D/2}} \times$$
$$\exp\left\{-\frac{1}{2}\big(\mathrm{Tr}(F\mathcal{K}^{-1}F^T) + \mathrm{Tr}(F^T\Sigma^{-1}F)\big) + \mathrm{Tr}(F^T\Sigma^{-1}\boldsymbol{\eta})\right\},$$

where $\eta = [\eta_1 \ \cdots \ \eta_D]$ is the $K \times D$ matrix of means variables η_d. The terms involving F have both row and column dependencies, one coming from $\mathcal{K}$ and the other from Σ. Hence, exact marginalization over F is difficult. Further, while the apriori marginal over each entry η_{id} of η are univariate Gaussians with zero mean and variance $\Sigma_{i,i} + \mathcal{K}_{d,d}$, the joint distribution over η will have both row and column dependencies, and hence exact marginalization of η is also difficult. While variational inference by assuming a fully factorized distribution over F, η will lead to a variational lower bound, such an approach to inference undermines a key property of GPTMs, viz dependencies along both rows and columns. Gibbs sampling based inference is possible for the model but could be computationally burdensome. It involves inverting $(K-1) \times (K-1)$ and $(D-1) \times (D-1)$ matrices for each entry in η in each sampling iteration.

In GPTMs, there are two sets of latent variables to consider: the matrix F, arising out of the GP, and variables (η, z), which are common in topic models [7]. In light of the previous discussion, we choose to maximize the log-likelihood over F and variationally marginalize it over (η, z). As we shall see shortly, the maximum aposteriori (MAP) inference over F can be done while maintaining the row and column dependencies. In particular, the first order conditions lead to a Sylvester equation [86, 98] in F involving both Σ and $\mathcal{K}$. Further, MAP inference over F leads to an embedding of the data points taking into account both the kernel, the covariance among topics, as well as observed words. While maximizing over F is unconventional in the context of GPs, related ideas have been explored in the recent literature in the context of probabilistic embedding using GPLVMs [5]. In Section 6.4, we compare the embedding performance of GPTMs to that of GPLVMs as well as CTMs.

6.3.1 Approximate Inference

Our goal in terms of learning is to choose (F, Σ, β) so as to maximize

$$\log p(w, F|\mathcal{K}, \Sigma, \beta) = \log E_{\eta,z}[p(w, z, \eta, F|\mathcal{K}, \Sigma, \beta)] \ . \tag{6.5}$$

In principle, one can also optimize over $\mathcal{K}$ using kernel learning methods, but we do not explore this aspect in this chapter. Since computing the expectation over the latent variables (η, z) is intractable, following [7], we propose a variational inference approach to lower bound the expectation over (η, z). In particular, for each document, we consider the family of fully factored variational distributions q as:

$$q(\eta_{1:K}, z_{1:N}|\lambda_{1:K}, \nu^2_{1:K}, \phi_{1:N}) = \prod_{i=1}^{K} q(\eta_i|\lambda_i, \nu_i^2) \prod_{n=1}^{N} q(z_n|\phi_n) \ , \tag{6.6}$$

where $q(\eta_i|\lambda_i, \nu_i^2)$ are univariate Gaussian distributions with mean λ_i and variance ν_i^2, and $q(z_n|\phi_n)$ are discrete distributions with parameter ϕ_n.

Using Jensen's inequality [7], for any F we have:

$$\begin{aligned} &\log p(w, F|\mathcal{K}, \Sigma, \beta) \\ &\geq \log p(F|\mathcal{K}) + \sum_{d=1}^{D} \Big\{ E_q[\log p(\eta_d|Fe_d, \Sigma)] \\ &+ E_q[\log p(z_d|\eta_d)] + E_q[\log p(w_d|z_d, \beta)] \Big\} + H(q) \end{aligned} \tag{6.7}$$

Table 6.1: Terms of the lower bound for expected loglikelihood

Term	Expression
$\log p(F\|\mathcal{K})$	$\frac{K}{2}\log\|\mathcal{K}^{-1}\| - \frac{KD}{2}\log 2\pi - \frac{1}{2}\mathrm{Tr}(F\mathcal{K}^{-1}F^T)$
$E_q[\log p(\eta_d\|Fe_d,\Sigma)]$	$\frac{1}{2}\log\|\Sigma^{-1}\| - \frac{K}{2}\log 2\pi - \frac{1}{2}\{\mathrm{Tr}(\mathrm{diag}(\nu^2)\Sigma^{-1}) + (\lambda - Fe_d)^T\Sigma^{-1}(\lambda - Fe_d)\}$
$E_q(\log p(z_n\|\eta))$	$\sum_{i=1}^K \lambda_i\phi_{n,i} - \zeta^{-1}\left(\sum_{i=1}^K \exp\{\lambda_i + \nu_i^2/2\}\right) + 1 - \log(\zeta)$
$E_q[\log p(w_n\|z_n,\beta)]$	$\sum_{i=1}^K \phi_{n,i}\log\beta_{i,w_n}$
$H(q)$	$\sum_{i=1}^K \frac{1}{2}(\log\nu_i^2 + \log 2\pi + 1) - \sum_{n=1}^N\sum_{i=1}^K \phi_{n,i}\log\phi_{n,i}$

We give the exact expressions for each term in Table 6.1. For the derivation of the last three terms we refer the reader to [7], since these terms are the same as in CTM. The first two terms are unique to our model and stem from the introduction of F. For the first term, by definition, we have

$$\begin{aligned}
&\log p(F|\mathcal{K}) \\
&= \sum_{i=1}^K \left\{ \frac{1}{2}\log|\mathcal{K}^{-1}| - \frac{D}{2}\log 2\pi - \frac{1}{2}f_i^T\mathcal{K}^{-1}f_i \right\} \\
&= \frac{K}{2}\log|\mathcal{K}^{-1}| - \frac{KD}{2}\log 2\pi - \frac{1}{2}\mathrm{Tr}(F\mathcal{K}^{-1}F^T) \ .
\end{aligned}$$

For the second term corresponding to each document we have

$$\begin{aligned}
E_q[\log p(\eta_d|Fe_d,\Sigma)] &= \frac{1}{2}\log|\Sigma^{-1}| - \frac{K}{2}\log 2\pi \\
&- \frac{1}{2}E_q[(\eta_d - Fe_d)^T\Sigma^{-1}(\eta_d - Fe_d)] \\
&= \frac{1}{2}\log|\Sigma^{-1}| - \frac{K}{2}\log 2\pi + \mathrm{Tr}(\mathrm{diag}(\nu_d^2)\Sigma^{-1}) \\
&\quad + (\lambda_d - Fe_d)^T\Sigma^{-1}(\lambda_d - Fe_d) \ ,
\end{aligned}$$

where $e_d \in \mathbb{R}^D$ is the all zeros vector with only the d^{th} entry as one. Further, note that the third term $E_q[\log p(z_n|\eta)]$ cannot be computed in closed form, and we obtain a variational lower bound (see Table 6.1) with parameter ζ following [7].

6.3.2 Parameter Updates

The variational lower bound is optimized by updating the variational parameters $(\lambda, \nu, \phi, \zeta)$ and the model parameters (F, Σ, β). Since parts of our objective function are similar to CTM, a number of updates remain the same [7]. In particular, for the parameters

corresponding to the topics, we have:

$$\beta_{i,j} \propto \sum_{d=1}^{D} \sum_{n=1}^{N_d} \phi_{dn,i} w_{dn}^{j}, \tag{6.8}$$

$$\phi_{dj,i} \propto \exp\{\lambda_i\} \beta_{i,j} \ , \tag{6.9}$$

where w_{dn}^{j} is an indicator that the n^{th} word in the d^{th} document is the j^{th} word in the vocabulary. Further, for each document, the update for ζ is given by:

$$\zeta = \sum_{i=1}^{K} \exp\{\lambda_i + \nu_i^2/2\} \tag{6.10}$$

A solution for λ_i and ν_i^2 cannot be obtained analytically For each document. So gradient descent is used with gradients

$$\begin{aligned} g_\lambda =& -\Sigma^{-1}(\lambda - Fe_d) + \Sigma_{n=1}^{N} \phi_{n,1:K} \\ & - (N/\zeta) \exp\{\lambda + \nu^2/2\} \\ g_{\nu_i^2} =& -\Sigma_{ii}^{-1}/2 - (N/2\zeta) \exp\{\lambda_i + \nu_i^2/2\} + 1/(2\nu_i^2) \ . \end{aligned}$$

We now focus on computation of parameters which are different from CTM. Since these are unique to our model, we present them in more detail.

Computation of Σ: Unlike in CTM, we have to compute one covariance matrix given multiple means. Starting with (6.7) we can pose the problem as:

$$\begin{aligned} \max_{\Sigma} \Bigg\{ & \frac{D}{2} \log |\Sigma^{-1}| - \frac{1}{2} \sum_{d=1}^{D} \mathrm{Tr}(\mathrm{diag}(\nu_d^2)\Sigma^{-1}) \\ & -\frac{1}{2} \mathrm{Tr}\big[(L-F)^T \Sigma^{-1} (L-F)\big] \Bigg\} \ , \end{aligned} \tag{6.11}$$

with $L = [\lambda_1 \ \cdots \ \lambda_D]$ where λ_d and ν_d denote the variational parameters associated with document d. Taking the derivative with respect to Σ, we get

$$\Sigma = \frac{1}{D} \left(\sum_{d=1}^{D} \mathrm{diag}(\nu_d^2) + \sum_{d=1}^{D} (\lambda_d - Fe_d)(\lambda_d - Fe_d)^T \right) \ . \tag{6.12}$$

Computation of F: The matrix $F \in \mathbb{R}^{K \times D}$ is entirely new in our model. From (6.7), the optimization problem over F can be posed as:

$$\min_{F} \Big\{ \mathrm{Tr}\big[(L-F)^T \Sigma^{-1} (L-F)\big] + \mathrm{Tr}\big[F \mathcal{K}^{-1} F^T\big] \Big\} \tag{6.13}$$

Taking derivative with respect to F and setting it to zero, we obtain the following equation:

$$\Sigma F + F\mathcal{K} = \sum_{d=1}^{D} \lambda_d e_d^T \mathcal{K} \ . \tag{6.14}$$

With $A = \Sigma$, $B = \mathcal{K}$ and $C = \sum_{d=1}^{D} \lambda_d e_d^T \mathcal{K}$, the equation is of the form: $AF + FB = C$. The equation is known as the Sylvester equation, and it is widely studied in control theory [85, 86]. A solution to the Sylvester equation exists if and only if no eigenvalue of A is equal to the negative of an eigenvalue of B. In our case, since A and B are both positive semi-definite, such a situation can arise only if A and B both have at least one zero eigen-value. For that to happen, both Σ and $\mathcal{K}$ have to be singular, implying Σ^{-1} and $\mathcal{K}^{-1}$ are not well defined. Since Σ and $\mathcal{K}$ both act as covariance matrix/function of a Gaussian distribution/process, we assume them to be full rank and positive definite.[1]

As a result, a solution to the Sylvester equation exists and can be obtained using standard methods [86, 98].

6.3.3 Inference On New Documents

In the learning phase, one obtains the parameters (β, Σ) as well as the best F for the training set. While applying the model on new documents, (β, Σ) will stay unchanged, and we do variational inference to obtain parameters $(\lambda, \nu, \phi, \zeta)$ on the test set. Further, using the fact that location of the mean $\mu_d = Fe_d$ is determined by a GP, we can use GP regression to obtain estimates of document means in the test set.

First, consider one new document, so that the corpus is of size $(D + 1)$. Let $\tilde{F} \in R^{K \times (D+1)}$ denote the matrix containing the means of the entire corpus so that $\tilde{F} = [F \ F_*]$, where F_* denotes the mean for the new document. Let $\tilde{f} = [f \ f_*]$ denote a row of $\tilde{F}$, where f corresponds to the first D documents and f_* corresponds to the new document. A kernel for the entire corpus can be expressed as follows:

$$\tilde{\mathcal{K}} = \begin{bmatrix} \mathcal{K}_{f,f} & \mathcal{K}_{f,*} \\ \mathcal{K}_{*,f} & \mathcal{K}_{*,*} \end{bmatrix},$$

[1] One can generalize the models using pseudo-inserves, but we do not consider such generalizations here.

where $\tilde{\mathcal{K}} \in R^{(D+1)\times(D+1)}$. From GP regression [96], we know that the posterior probability distribution $p(f_*|f)$ can be expressed as:

$$p(f_*|f) = \frac{1}{(2\pi)^{1/2}|\overline{\mathcal{K}}|^{1/2}} \exp\left(-\frac{(f_* - \overline{f})^2}{2\overline{\mathcal{K}}}\right) , \tag{6.15}$$

where $\overline{f} \in \mathbb{R}$ and $\overline{\mathcal{K}} \in \mathbb{R}_+$ is given by

$$\overline{\mathcal{K}} = \mathcal{K}_{*,*} - \mathcal{K}_{*,f}\mathcal{K}_{f,f}^{-1}\mathcal{K}_{f,*} \tag{6.16}$$

$$\overline{f} = \mathcal{K}_{*,f}\mathcal{K}_{f,f}^{-1} f \ . \tag{6.17}$$

Similarly we can obtain a posterior distribution for a collection of M new documents. Let $F_* \in R^{K\times M}$ denote the matrix containing the means of the new documents and $\tilde{F} = [F \ F_*]$. Following the same steps as above, we note that each row $f_{i,*}$ of F_* follows a multi-variate Gaussian distribution with mean equal to the row $\overline{f}_i$ of $\overline{F}$, where $\overline{F}^T = \mathcal{K}_{*,f}\mathcal{K}_{f,f}^{-1}F^T$, and covariance $\overline{\mathcal{K}} = \mathcal{K}_{*,*} - \mathcal{K}_{*,f}\mathcal{K}_{f,f}^{-1}\mathcal{K}_{f,*}$. Since the rows of F_* are independent, the probability of the entire matrix is given by

$$\begin{aligned} p(F^*|F,\tilde{\mathcal{K}}) = \prod_{i=1}^{K} p(f_{i,*}|F,\tilde{\mathcal{K}}) = \frac{1}{(2\pi)^{K/2}|\overline{\mathcal{K}}|^{K/2}} \times \\ \exp\left\{-\frac{1}{2}\mathrm{Tr}\big[(F_* - \overline{F})\overline{\mathcal{K}}^{-1}(F_* - \overline{F})^T\big]\right\} \ . \end{aligned} \tag{6.18}$$

With the above prior distribution on F_* on the test set conditioned on the F from the training set, the rest of the generative model for the test set remains the same as GPTM. Introducing variational distributions as before, inference on new documents boils down to optimizing the variational parameters and F_* over the test set. The optimization over the variational parameters are same as in the training set. Focussing on the terms involving F_*, we get the following problem:

$$\begin{aligned} \min_{F_*} \Bigg\{ & \frac{1}{2}\mathrm{Tr}\big[(L - F_*)^T\Sigma^{-1}(L - F_*)\big] \\ & + \frac{1}{2}\mathrm{Tr}\big[(F_* - \overline{F})\overline{\mathcal{K}}_j^{-1}(F_* - \overline{F})^T\big]\Bigg\} \ , \end{aligned} \tag{6.19}$$

where $L = [\lambda_1 \ \cdots \ \lambda_M]$ is the $K \times M$ matrix of variational parameters λ_d on the test set. The first order conditions lead to the following matrix equation:

$$\Sigma F_* + F_*\overline{\mathcal{K}} = \left(\sum_{d=1}^{D} \lambda_d e_d^T + \overline{F}\right)\overline{\mathcal{K}} \ , \tag{6.20}$$

which is again a Sylvester equation. Note that the right hand side is affected by both observed words in the test set in the form of λ_d and by the estimated mean $\overline{F}$ obtained from training set documents.

6.4 Experimental Evaluation

In this section we evaluate GPTM both as a topic model and as an embedding method, respectively in comparison to CTM and GPLVM.

Datasets: Our experiments are performed on 6 text data sets. `Dif100`, `Sim100`, and `Same100` are subsets of the 20Newsgroup data set, each having 300 data points from 3 categories; CMU100 is a larger subset which contains 1000 documents from 10 categories. We also report experiments on NASA's Aviation Safety Reporting System (ASRS) dataset: `ASRS` is a subset of 1000 reports from 25 categories, and `ASRS3` is a subset of 788 documents from 3 categories.

Kernel Functions: We consider two kernels for our experiments: an unsupervised nearest-neighbor kernel derived from the document vectors and a semi-supervised must-link kernel derived from must-link constraints on some pairs of documents.

Let $\mathcal{X} = \{x_1, \ldots, x_D\}$ denote a set of feature vectors represented by the word counts in a given document. Let $G = (V, E)$ be a k-nearest neighbor graph which is symmetrized by making sure that $(x_i, x_j) \in E$ whenever $(x_j, x_i) \in E$. The graph neighbors are determined using cosine similarity among the document vectors. The nearest neighbor (NN) kernel is defined as:

$$\mathcal{K}_{NN}(x_i, x_j) = \begin{cases} \gamma \exp\left(\frac{-d(x_i,x_j)}{2\sigma^2}\right) & \text{if } (x_i, x_j) \in E \\ c & \text{if } i = j \\ 0 & \text{otherwise ,} \end{cases} \tag{6.21}$$

where $d(x_i, x_j) = 1 - \frac{x_i^T x_j}{\|x_i\| \|x_j\|}$, and σ^2, γ, c are constant parameters chosen to ensure $\mathcal{K}_{NN}$ is positive definite. The parameters are tuned for each data set using cross validation.

While $\mathcal{K}_{NN}$ can be used in practice, constructing a kernel based on document vectors may not be conceptually desirable from a generative model perspective. Note that the experiments using $\mathcal{K}_{NN}$ demonstrate the ability of our model to incorporate

neighborhood information, which may come from supplemental information regarding the documents. Further, GPTMs are not tied to the use of $\mathcal{K}_{NN}$. To illustrate the effectiveness of GPTMs, we consider another kernel constructed purely based on semi-supervised information, viz must-link constraints between certain pairs of documents, without utilizing the documents themselves.

In particular, we consider a semi-supervised setting with a set $\mathcal{C}$ of must link constraints, i.e., if $(x_i, x_j) \in \mathcal{C}$, then the documents are assumed to have the same label. Using such a must-link constraint set, we define the must-link (ML) kernel as:

$$\mathcal{K}_{ML}(x_i, x_j) = \begin{cases} \gamma & \text{if } (x_i, x_j) \in \mathcal{C} \\ c & \text{if } i = j \\ 0 & \text{otherwise ,} \end{cases} \tag{6.22}$$

where $\gamma > 0$ and c is chosen to ensure positive definiteness of $\mathcal{K}_{ML}$. The parameters are tuned per data set using cross validation.

Perplexity Computation: In our experiments we use test-set perplexity to evaluate variants of GPTM as well as compare GPTM to CTM. When comparing variants of GPTM we compute perplexity as:

$$Perplexity_{w,F_*} = \exp\left(-\frac{\sum_{i=1}^{M} \log p(w_i, F_*)}{\sum_{i=1}^{M} N_i}\right) , \tag{6.23}$$

based on the joint likelihood of the test documents w_i with MAP estimate F_*.

To compare our model to CTM, we compute conditional perplexity for $GPTM$ as follows:

$$Perplexity_{w|F_*} = \exp\left(-\frac{\sum_{i=1}^{M} \log p(w_i|F_*)}{\sum_{i=1}^{M} N_i}\right) \tag{6.24}$$

Since the conditional distribution $p(w|F*)$ is a distribution over the space of documents, which is the same for CTM, the perplexity comparison is meaningful and fair. We also consider additional measures, including topics inferred, document embeddings generated, and classification performance using the embeddings, to get a better understanding of their comparative performance.

Table 6.2: Perplexity on hold out test set.

	CTM	GPTM $\mathcal{K}_{ML}$	GPTM $\mathcal{K}_{KNN}$
Dif100	1231 ± 32	1195 ± 49	$\mathbf{1183 \pm 36}$
Sim100	1720 ± 19	1706 ± 22	$\mathbf{1684 \pm 21}$
Same100	761 ± 6	758 ± 6	$\mathbf{755 \pm 12}$
ASRS	491 ± 8	488 ± 8	$\mathbf{483 \pm 3}$
News100	2944 ± 83	2943 ± 66	$\mathbf{2936 \pm 82}$

6.4.1 GPTM vs. CTM

We report perplexity results comparing CTMs with GPTMs using both the ML-kernel and the NN-kernel with neighborhood $k = 10$ in Table 6.2. Perplexity was evaluated on a held out test-set using five-fold cross validation and the number of topics set to three. Compared to CTMs, we observe mild to moderate improvements in perplexity across all datasets for GPTMs using both kernels. Thus, in terms of perplexity on the test-set, GPTMs are better or at least as good as CTMs. For GPTMs, the NN-kernel appears to perform mildly better compared to the ML-kernel.

Table 6.3: Topics extracted by CTM from 20Newsgroup data

Topic 1	**Topic 2**	**Topic 3**	**Topic 4**	**Topic 5**	**Topic 6**
car	**plant**	echo	pittsburgh	system	god
oil	**court**	list	period	mac	existence
brake	scsi	motif	lemieux	files	exist
fluids	**disk**	xterm	stevens	disk	islam
tires	**cement**	set	play	file	standard
dot	data	mailing	power	comp	science
convention	ram	host	njd	software	atheism
abs	card	mail	scorer	ftp	religion
braking	property	sun	pgh	sys	religion
cars	**atlantic**	school	islanders	macintosh	laws

We also qualitatively examined the topics obtained by both models. In Tables 6.3,6.4 and 6.5 we list the 10 most likely words for some of the topics obtained from the `News100` dataset. As is evident, GPTMs returns topics which appear as interpretable as those obtained by CTM. Taking a closer look, we can see that some of the topics

Table 6.4: Topics extracted by GPTM using $\mathcal{K} = \mathcal{K}_{NN}$ and the 20Newsgroup data

Topic 1	**Topic 2**	**Topic 3**	**Topic 4**	**Topic 5**	**Topic 6**
car	**convention**	drive	play	system	god
oil	don	**disk**	power	mac	good
brake	party	hard	period	files	islam
fluids	people	drives	pittsburgh	disk	exist
tires	price	bios	islanders	comp	pain
dot	business	controller	hockey	file	laws
abs	hess	floppy	scorer	software	thing
cars	karl	card	pts	macintosh	time
braking	institute	rom	jersey	sys	existence
system	libertarian	scsi	good	ftp	faith

appear more coherent in GPTMs. For example, Topic 1 in CTM contains car part related words and the word `convention`. In GPTM with NN-kernel, we have a topic on car parts (Topic 1), and a topic about the liberterian party convention held at the Karl Hess business institute (Topic 2). Topic 2 in CTM appears to be memory related, along with words such as `plant`, `court`, `cement`, and `atlantic`. While in GPTM with $\mathcal{K}_{NN}$ the corresponding topic (Topic 3) appears only memory related. The two versions of GPTM produce rather similar topics. With $\mathcal{K}_{ML}$ the hockey topic appears more generic, possibly because there are multiple hockey related documents within the same class. While these are only anecdotal examples, the bottom line is that both CTMs and GPTMs produce high quality interpretable topics.

6.4.2 Variants Of GPTMs

We compare four variants of GPTMs to understand the value of the topic covariance matrix Σ and the kernel matrix $\mathcal{K}$. In particular, we consider: (i) GPTM-SI-KI, where both Σ and $\mathcal{K}$ are identity matrices. Since correlation among topics and similarity among documents are not considered, this model is closest to LDA in spirit; (ii) GPTM-KI, where $\mathcal{K}$ is identity but Σ is learnt from the data. This model is closest in spirit to CTM; (iii) GPTM-SI, where Σ is identity and $\mathcal{K}$ is set to either $\mathcal{K}_{NN}$ or $\mathcal{K}_{ML}$. This model is similar to LDA with a kernel over documents; and (iv) GPTM, where Σ is learned from data and $\mathcal{K}$ is either $\mathcal{K}_{NN}$ or $\mathcal{K}_{ML}$.

Table 6.5: Topics extracted by GPTM using $\mathcal{K} = \mathcal{K}_{ML}$ and the 20Newsgroup data

Topic 1	**Topic 2**	**Topic 3**	**Topic 4**	**Topic 5**	**Topic 6**
car	liberation	drive	team	system	god
brake	committee	tape	hockey	mac	morality
oil	institute	scsi	game	files	atheism
fluids	president	problem	players	ftp	moral
dot	hess	dos	games	file	existence
abs	karl	windows	play	disk	exist
tires	business	system	year	comp	standard
braking	national	floppy	nhl	software	faith
system	defense	cable	player	sys	good
cars	college	computer	playoffs	macintosh	true

Table 6.6: Different Variants of GPTM. Performance improves as more informative kernel and covariance matrices are considered.

	GP-SIKI	GP-KI	GP-SI $\mathcal{K}_{ML}$	GP-SI $\mathcal{K}_{KNN}$	GPTM $\mathcal{K}_{ML}$	GPTM $\mathcal{K}_{KNN}$
Dif100	1264 ± 47	1232 ± 29	1198 ± 46	1138 ± 42	1121 ± 40	$\mathbf{1095 \pm 53}$
Sim100	1778 ± 51	1741 ± 51	1720 ± 46	1656 ± 41	1639 ± 46	$\mathbf{1610 \pm 37}$
Same100	829 ± 21	792 ± 11	745 ± 28	633 ± 14	638 ± 12	$\mathbf{608 \pm 10}$
ASRS	514 ± 11	508 ± 7	472 ± 7	485 ± 9	$\mathbf{466 \pm 9}$	467 ± 7
News100	3093 ± 89	3047 ± 56	2911 ± 65	2877 ± 79	2818 ± 51	$\mathbf{2769 \pm 86}$

Table 6.2 shows the perplexity numbers on a held out test set using five fold cross-validation. We observe a consistent ordering in terms performance. GPTM-SI-KI performs worst, followed by GPTM-SI, GPTM-KI, and GPTM performs the best. The fact that GPTM-KI performs better than GPTM-SI-KI is consistent with the observation that CTM outperforms LDA. The comparison between GPTM-KI and GPTM-SI shows that GPTM-SI has a consistent better performance possibly implying the kernel adds more value in terms of perplexity than the topic covariance matrix. Finally, the full GPTM outperforms all the special cases. The results clearly illustrate the value in having a suitable kernel over the documents.

Comparing the two different Kernels in GPTM, $\mathcal{K}_{NN}$ fairly consistently results in better perplexity numbers. Interestingly, as far as perplexity is concerned, the nearest neighbor information appears more valuable compared to must-link constraints among

the documents.

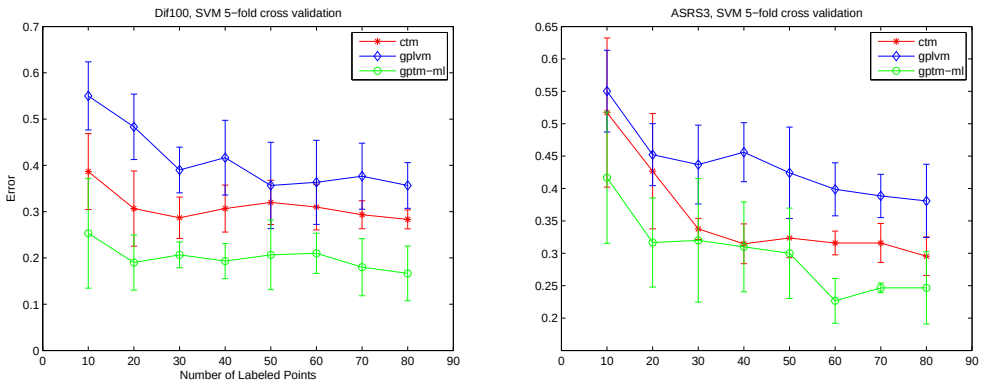
6.4.3 Embeddings

We investigate the embeddings obtained by CTM, GPLVM and GPTM using the kNN kernel. With GPLVM a separate derivation of updates is required for different Kernels. We used an existing implementation based on the RBF Kernel. Due to space constraints, we only show results on `Dif100` and `ASRS3` (Figure 6.3). The number of topics is set to the correct number of classes for all models and datasets. For CTM we plot λ, for GPLVM we plot the embedded points and for GPTM we plot F. Similar to CTM, for K topics, the degrees of freedom in η for GPTMs is $K-1$ since it eventually gets mapped to the topic simplex. Thus, following CTM, we display the embedding in $(K-1)$ dimensions. The data points are colored based on their true class label. Note that the CTM embeddings are based on the document structure as provided by the topic model, while GPLVM embeddings are based on the kernel and observed features. GPTM leverages both the topic structure of words as well as the provided kernel.

From Figure 6.3, we note that for each dataset GPTM produces an embedding where the classes are most cleanly separated. The kernel appears to be helping in preserving the neighborhood structure of the documents which is coherent with the class labels. The better embeddings also help explain why the perplexity goes down when the kNN-Kernel is used.

In Figure 6.4, we illustrate that the kernel can indeed provide control over document embeddings using the semi-supervised ML kernel defined in (6.22). We show results on the final embedding using this kernel on `Dif100` and `ASRS3` using 10, 100, and fully labeled points. Labels are converted to constraints using transitive closure. If we know the class labels (partially) upfront, the question is can we incorporate this knowledge into the topic model? As shown in Figure 6.4, in each data set, the classes become increasingly separated with additional labeled points while maintaining the structure in each class. The effect is especially evident in Figure 6.4(f), where the red and blue classes remain somewhat intertwined even with the fully labeled data.

Finally, we apply support vector machines on the embeddings generated by CTM, GPLVM, and GPTM using the must-link kernel. For CTMs and GPLVMs, the partially labeled data is used only for training the SVM. For GPTMs, they are used to determine



(a) Dif100 (b) ASRS3

Figure 6.2: The SVM algorithm is applied to the respective outputs of CTM, GPTM and GPLVM.

the ML kernel in GPTM as well as for training the SVM. Due to space constraints, we show results only on `Dif100` and `ASRS3` in Figure 6.2. The SVM is trained with an RBF Kernel, and the parameters are tuned using 5-fold cross validation. The classification results show that GPTM produces embeddings with the best class separability and lowest error rates.

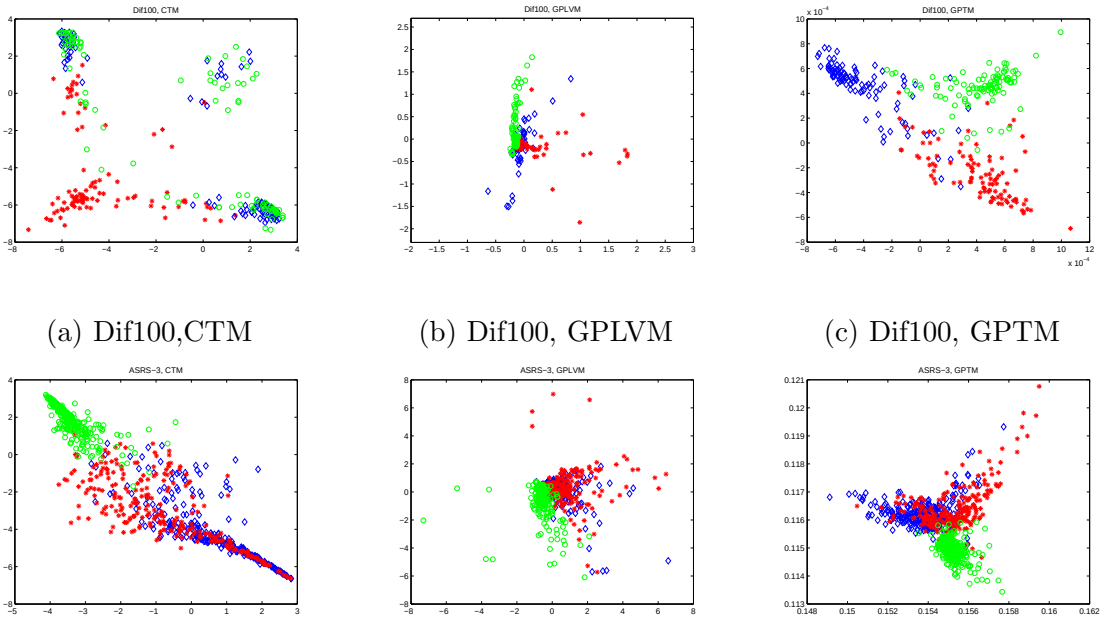
6.5 Discussion

In this section we briefly discuss computational aspects in GPTMs and how GPTMs compare to CTMs in terms of capturing topic correlations.

6.5.1 Computational Aspects

There are two major new computational aspects in our model. One pertains to the inversion of the $D \times D$ kernel matrix during GP regression on the test set, and the second is the solution of a Sylvester equation.

The inversion of the kernel matrix is something that most GP based models have to perform [96]. In recent years, progress has been made on making the computation scalable. For example, several sparsity-based approaches [99, 100] have been developed,



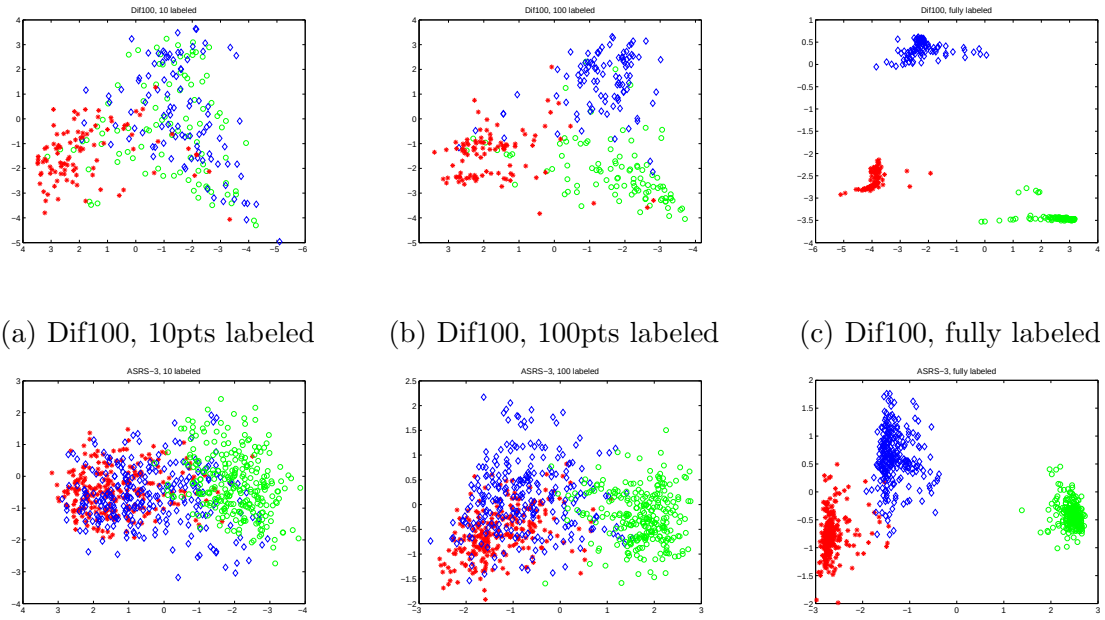
(a) Dif100,CTM (b) Dif100, GPLVM (c) Dif100, GPTM

(d) ASRS3, CTM (e) ASRS3, GPLVM (f) ASRS3, GPTM

Figure 6.3: Embeddings obtained from CTM, GPLVM, and GPTM without using class label information. GPTMs separate the classes better than CTM and GPLVM (Best viewed in color).

which can be readily leveraged in GPTMs. We will explore such approaches in future work.

Solving the Sylvester equation repeatedly is an important computational step in GPTMs. The equation has two matrices: the $K \times K$ topic covariance matrix Σ and the $D \times D$ kernel matrix $\mathcal{K}$, where $D \gg K$. During learning, the smaller matrix Σ gets updated iteratively whereas the bigger matrix $\mathcal{K}$ is fixed. Using the fact that $\mathcal{K}$ does not change during training, computations involved in solving the Sylvester equation repeatedly can be speeded up significantly. In the test set, since both Σ and $\overline{\mathcal{K}}$ are fixed and only the variational parameters λ_d on the test set get iteratively updated, repeated solution of the Sylvester equation can also be done efficiently.



(a) Dif100, 10pts labeled (b) Dif100, 100pts labeled (c) Dif100, fully labeled

(d) ASRS3, 10pts labeled (e) ASRS3, 100pts labeled (f) ASRS3, fully labeled

Figure 6.4: Semi-supervised embeddings from GPTM using ML kernel. As more labeled points are considered, GPTM separates the classes better while preserving topic structure. (Best Viewed in Color)

6.5.2 Topic Correlations

Throughout this chapter we have made comparisons between GPTMs and CTMs. While both models have a topic covariance matrix Σ, the type of information captured in Σ is somewhat different in GPTMs, as a result the covariance matrices in these two models cannot be directly compared.

Consider an example where we model only two topics. A level set of the prior distribution for CTMs is an ellipse over the topic space in $\mathbb{R}^2$. In GPTMs, each document has its own mean in $\mathbb{R}^2$ drawn from the GP. As a result, a level set of the prior for GPTMs involve D ellipses (possibly overlapping/merged) with the same axes but different centroids. In other words, GPTMs consider D different Gaussians with different means but the same covariance matrix. In principle, one can consider different covariance matrices

in different parts of the topic space. Such extensions will be considered in future work.

The topic covariance Σ in CTMs is inferred only based on the observed words. On the other hand, in GPTMs, the estimated covariance matrix Σ also has a dependency on F which in turn depends on the kernel $\mathcal{K}$. While Σ in GPTMs may not capture the exact same information as that in CTMs, as shown in Table 6.6, the performance in terms of perplexity improves when a non-identity covariance matrix is considered.

6.6 Conclusions

We have introduced a novel family of topic models called GPTMs which can take advantage of both the topic structure of documents and a given kernel among documents. GPTMs can be viewed as a systematic generalization of CTMs which leverages a kernel over documents. The kernel is used to define a GP prior over the topic mixing proportions of documents ensuring that similar documents according to the kernel have similar mixing proportions apriori. The final topic proportions for each document depend both on the kernel as well as the observed words. As our experiments illustrate, with a suitable kernel choice our model can provide good results both in terms of extracted topics as well as the resulting embedding. In particular the kernel allows semi-supervised information to be incorporated into the model, and we illustrate that increased semi-supervision leads to better class separability in the topic space.

Chapter 7

Object-Oriented Machine Learning Toolbox for MATLAB (MALT)

In this chapter we describe MALT, a powerful object-oriented Machine Learning toolbox, which was developed as a framework for conducting all empirical evaluations in this thesis.

7.1 Motivation

Most currently available machine learning code is written in terms of non-object oriented functions. When utilizing other people's code, several time-consuming challenges may arise. The code in many instances may not be well-documented or easy to use. Further the assumed data formats may be very different from algorithm implementation to implementation.

For instance, if we are interested in running experiments which utilize existing code for Support Vector Machines (SVMs) and Boosting, we may find ourselves confused as to what parameters one can use and how to pass them, unless the code is thoroughly documented. Further we may find that the SVM and Boosting implementations assume different data formats. As a result utilizing provided code can be very costly. Our

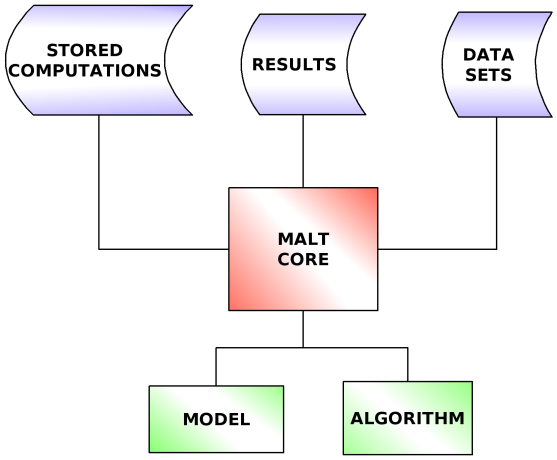


Figure 7.1: MALT framework. Allows algorithms to be plugged in. Provides unified parameter passing, repositories, and generic cross validation.

goal was to provide a framework which unifies how parameters are passed, thereby minimizing the need to study usage. Further our objective was to introduce a uniform data format for all algorithms to make comparisons easier.

Challenges can also arise when experiments on self-developed code are conducted. Every algorithm utilizes different parameters. As a result cross-validation has to be written over and over again, taking into consideration different usages of different algorithms. Another issue may arise when experiments are conducted on shared resources. After running cross-validation for 2 weeks, someone may shut down a machine. Our framework was designed to provide a cross-validation method in a generic sense, meaning that it works with all algorithms, irrespective of what parameters they accept. The cross-validation method in MALT was also built to withstand interruptions. If computations are interrupted, they can be resumed.

MALT is an object-oriented MATLAB toolbox, created with the intention to address challenges that arise with empirical evaluations in Machine Learning and Data Mining. It allows a researcher to conduct experiments very efficiently and obtained auto-generated graphs with little effort.

Currently there is no other Machine Learning package which can accomplish the same. WEKA is on of the most well known machine learning packages. However, it

is written in JAVA. While it works well, setting up experiments, adding self-developed code and figuring out how to use functions, can take up a lot of time.

Spider is another object-oriented package for MATLAB. It is a very useful package which illustrates the power of object-oriented design. However, the framework does not follow object-oriented principles as strictly as MALT. It also does not allow incremental cross-validation, or the ability to use repositories and arbitrary data sets.

7.2 MALT Features

Rather than describing the design of MALT we will highlight some of its features.

7.2.1 Unified Passing of Parameters

Each algorithm in the MALT framework is invoked in the same fashion and it returns results in the same way. Parameters are passed within an object, which means the user does not have to memorize the order of parameters. This unified interface is powerful because the time one has to spend with reading documentation is minimal.

7.2.2 Generic Data Sets

MALT has its own data set format. Unlike spider however, it allows for arbitrary data sets to be stored. In text modeling problems features might be represented by word counts. One might even be interested in knowing what the words are. MALT allows the user to store arbitrary components of a data set.

7.2.3 Generic Cross Validation

The MALT framework provides a generic cross validation algorithm. The user can plug in any algorithm into the framework and will be able to produce cross validation results.

7.2.4 Incremental Cross Validation and Reruns

Unlike any existing package, MALT provides the ability to run cross validation incrementally. In other words existing cross validation results can be extended by adding one algorithm, without having to rerun anything else. Interruptions of experiments, or

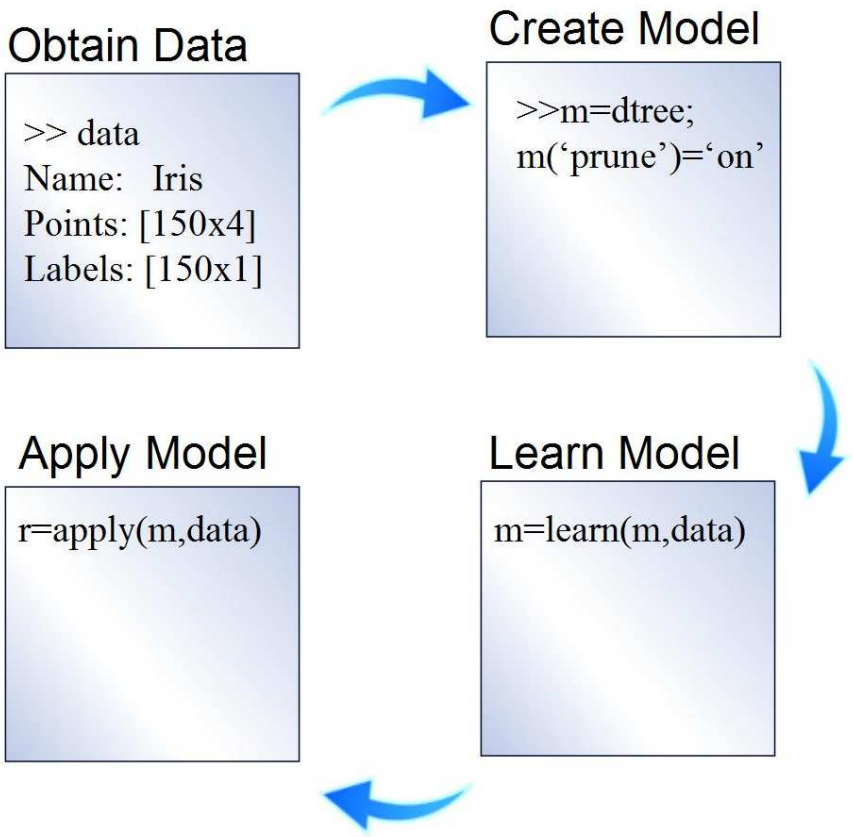


Figure 7.2: Unified use of algorithms.

the desire to add additional methods to existing results do not pose a problem with MALT. Another very useful feature is the ability to rerun only one algorithm, given an existing cross validation result.

7.2.5 Built-in Repositories

Repositories in MALT are objects which represent primitive databases. They allow data set, result sets, or anything else to be stored. For instance one can create a repository of datasets used in a particular experiment, and use them conveniently. Repositories can also be used to speed up algorithms. For example if k-nearest neighbors have to be

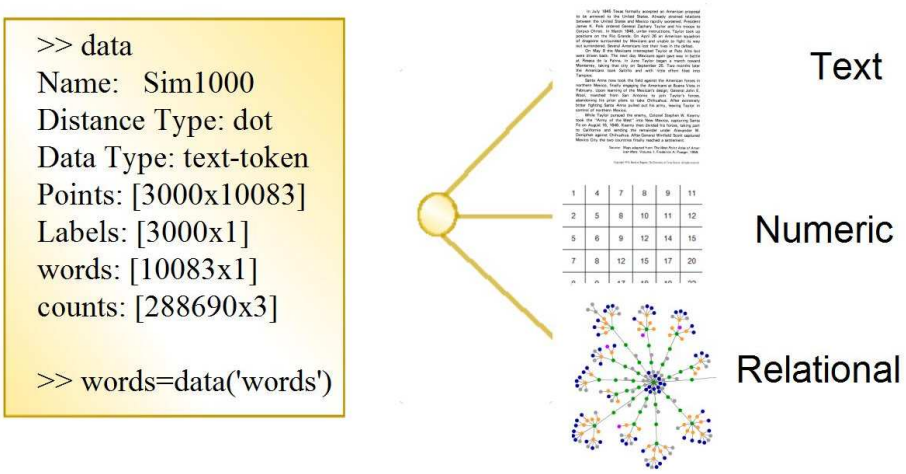


Figure 7.3: Data sets represented by a collection of components.

repeatedly computed in cross validation, one can precompute them and deposit them in a repository. This is another useful feature which can only be found in MALT.

7.2.6 Auto-Generated Plots

Results produced by cross validation are objects which can be plotted right away. The result set object fully labels the graph, making it publication ready.

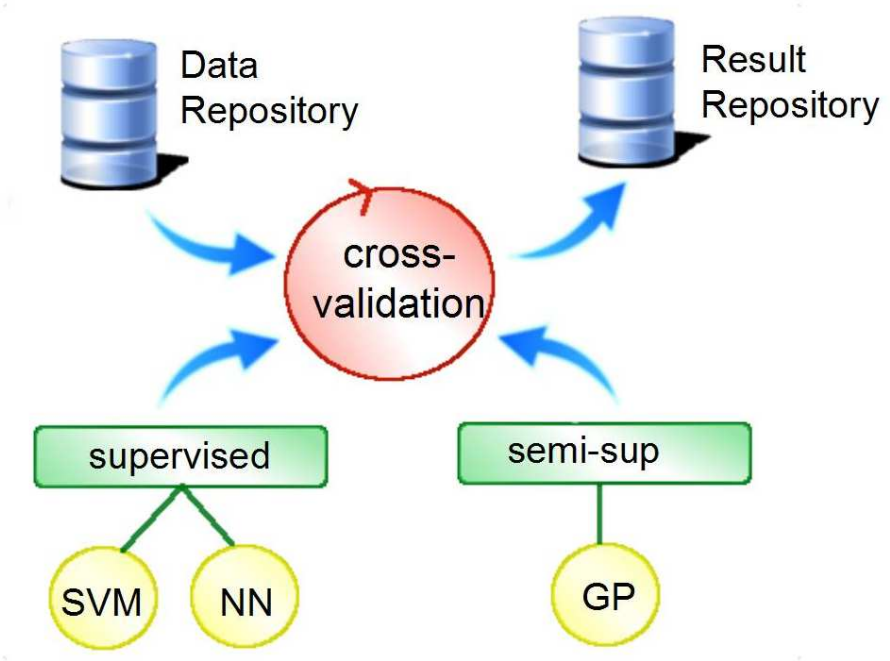


Figure 7.4: Generic Cross Validation

Figure 7.5: Repositories represent primitive and easy to use storage containers.

Chapter 8

Conclusions

In this thesis we have illustrated how the concept of dimensionality reduction can play a very important role beyond preprocessing and visualizing of data. In the first part of the thesis we have utilized semi-supervised dimensionality reduction to gain a better understanding of an entire family of semi-supervised predictive methods. As a result we have introduced novel label propagation algorithms based on exiting manifold embedding methods. We have also empirically illustrated that the proposed approaches are competitive in performance with the state-of-the-art methods in label propagation. The implication is that potential advances in non-probabilistic dimensionality reduction methods would translate to label propagation.

When it comes to probabilistic dimensionality reduction methods we attack a very important problem in statistical machine learning, namely modeling of matrices while capturing covariance dependencies across each dimension. We first introduced BMR, a very simple and scalable approach for mutli-label classification. The empirical evaluations indicates a number of favorable properties, including a competitive performance and high scalability.

With Probabilistic Matrix Addition we propose a model for arbitrary size real-valued matrices. Our model is closely related to the Linear Model of Corregionalization, which has a rich background in the field of Geostatistics. We illustrate through experimental evaluation that the PMA is approach is very competitive with the state of the art in multi-label classification. Further we illustrate competitive performance compared to Probabilistic Matrix Factorization when it comes to missing value prediction.

To illustrate the power of PMA and its use as part of a bigger model, we apply PMA as the prior in a topic model. With it we introduce the first topic model capable of capturing the covariance among topics while modeling the covariance among documents. The experiments illustrate the ability of the model to incorporate semi-supervised information without sacrificing the interpretability of topics.

In this thesis we have illustrated how intricate dependencies can be modeled effectively while dealing with high dimensional data. In particular we have introduced a new family of matrix priors, which move away from the i.i.d assumption and are applicable to a wide variety of applications. Our work is grounded in its relationship to the research from the field of Geo-statistics, in particular the Linear Model of Corregionalization. Furthermore we have illustrated the effectiveness of the idea when applied to Topic Modeling.

A possible extension of the work presented in this thesis would be to consider higher dimensional data structures such as tensors. In the case of tensors, for each dimension one would sample a tensor. The final output would be the sum of all the generated tensors. Due to the sparse dependency structure in PMA, a tensor extension would not be problematic.

References

[1] A. Agovic, A. Banerjee, A. Ganguly, and V. Protopopescu. Anomaly detection using manifold embedding and its applications in transportation corridors. *Intell. Data Anal.*, 13:435–455, August 2009.

[2] S. Kadoury and M.D. Levine. Face detection in gray scale images using locally linear embeddings. *Comput. Vis. Image Underst.*, 105:1–20, January 2007.

[3] X. Liu, H. Lu, and D. Zhang. Head pose estimation based on manifold embedding and distance metric learning. In *ACCV 09: Proceedings of the 9th Asian Conference on Computer Vision*, pages 61–70, 2009.

[4] G. Alvina and V. René. Clustering and dimensionality reduction on riemannian manifolds. In *CVPR 08: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2008*, 2008.

[5] N. Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. In *NIPS 03: Proceedings of the 16th Annual Conference on Neural Information Processing Systems*, 2003.

[6] E. Hörster, R. Lienhart, and M. Slaney. Image retrieval on large-scale image databases. In *Proceedings of the 6th ACM international conference on Image and video retrieval*, CIVR '07, pages 17–24, New York, NY, USA, 2007. ACM.

[7] D. Blei and J. Lafferty. Correlated topic models. *NIPS 06: Proceedings of the 19th Annual Conference on Neural Information Processing Systems*, 2006.

[8] H. Wackernagel. *Multivariate Geostatistics: An Introduction With Applications.* Springer-Verlag Berlin, 2003.

[9] A. E. Gelfand and S. Banerjee. Multivariate spatial process models. In A. E. Gelfand, P. Diggle, P. Guttorp, and M. Fuentes, editors, *Handbook of Spatial Statistics*. CRC Press, 2010.

[10] A. Agovic and A. Banerjee. A unified view of graph-based semi-supervised learning: Label propagation, graph-cuts, and embeddings. Technical Report 09-012, University of Minnesota - Department of Computer Science and Engineering, May 2009.

[11] A. Agovic, H. Shan, and A. Banerjee. Analyzing aviation safety reports: From topic modeling to scalable multi-label classification. In *Proceedings of the 2010 Conference on Intelligent Data Understanding, CIDU 2010*, pages 83–97, 2010.

[12] A. Agovic and A. Banerjee. Probabilistic matrix addition. In *Proceedings of the Twenty-Eighth International Conference on Machine Learning*, 2011.

[13] A. Agovic and A. Banerjee. Gaussian process topic models. In *Proceedings of the Twenty-Sixth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-10)*, pages 10–19, Corvallis, Oregon, 2010. AUAI Press.

[14] I. Joliffe. *Principal Component Analysis*. Springer-Verlag, 1996.

[15] D. de Ridder and R. Duin. Locally linear embedding for classification. Technical Report PH-2002-01, Delft University of Technology, 2002.

[16] Zhenyue Z. and Hongyuan Z. Principal manifolds and nonlinear dimensionality reduction via tangent space alignment. *SIAM J. Sci. Comput.*, 26(1):313–338, 2005.

[17] H. Zha and Z. Zhang. Isometric embedding and continuum ISOMAP. In *Proceedings of the 20th International Conference on Machine Learning*, 2003.

[18] M. Collins, S. Dasgupta, and R. Schapire. A generalization of principal component analysis to the exponential family. In *Proc. of the 14th Annual Conference on Neural Information Processing Systems (NIPS)*, 2001.

[19] J. Forster and M. K. Warmuth. Relative expected instantaneous loss bounds. In *Proc. of the 13th Annual Conference on Computational Learing Theory (COLT)*, pages 90–99, 2000.

[20] A. Banerjee, S. Merugu, I. Dhillon, and J. Ghosh. Clustering with Bregman divergences. *Journal of Machine Learning Research*, 6:1705–1749, 2005.

[21] I. Borg and P. Groenen. *Modern Multi-dimensioanl Scaling*. Springer, 1996.

[22] A. M. Andrew. Statistical pattern recognition. *Robotica*, 18(2):219–223, 2000.

[23] Introduction to Statistical Pattern Recognition. *K. Fukunaga*. Academic Press, 1990.

[24] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

[25] T. Tangkuampien and T.-J. Chin. Locally linear embedding for markerless human motion capture using multiple cameras. In *Proccedings of Digital Image Computing: Techniques and Applications*, 2005.

[26] X. Yang, H. Fu, H. Zha, and J. Barlow. Semi-supervised nonlinear dimensionality reduction. In *Proceedings of the 23rd International Conference on Machine Learning*, 2006.

[27] D. Donoho and C. Grimes. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedinsg of the National Academy of Science*, 100(10), 2003.

[28] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, June 2003.

[29] J. Tanenbaum, V. de Silva, and J. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2323, 2000.

[30] M. Bernstein, V. de Silva, J. Langford, and J. Tenenbaum. Graph approximations to geodesics on embedded manifolds. Technical report, Stanford University, 2000.

[31] D. Kulpinski. Lle and isomap analysis of spectral and color images. Master's thesis, Simon Fraser University, 2002.

[32] R. Pless. Image spaces and video trajectories: Using Isomap to explore video sequences. In *Proceedings of IEEE International Conference on Computer Vision*, 2003.

[33] O. C. Jenkins and M. J. Mataric. A spatio-temporal extension to Isomap nonlinear dimensionality reduction. In *Proceedings of the 21st International Conference on Machine Learning*, 2004.

[34] B. Ferris, D. Fox, and N. Lawrence. Wifi-slam using gaussian process latent variable models. In *IJCAI 07: Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 2480–2485, 2007.

[35] C. Ek, P. Torr, and N. Lawrence. Gaussian process latent variable models for human pose estimation. pages 132–143. 2008.

[36] L. Eciolaza, M. Alkarouri, N.D. Lawrence, V. Kadirkamanathan, and P.J. Fleming. Gaussian process latent variable models for fault detection. pages 287–292, 2007.

[37] X. Wang, X. Gao, Y. Yuan, D. Tao, and J. Li. Semi-supervised gaussian process latent variable model with pairwise constraints. *Neurocomput.*, 73:2186–2195, June 2010.

[38] R. Urtasun and T. Darrell. Discriminative gaussian process latent variable model for classification. In *Proceedings of the 24th international conference on Machine learning*, ICML '07, pages 927–934, New York, NY, USA, 2007. ACM.

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

[40] K. V. Mardia and C. R. Goodall. Spatial-temporal analysis of multivariate environmental monitoring data. In *Multivariate Environmental Statistics*, pages 347–386. Elsevier Science Publishers B.V., 1993.

[41] A. Gelfand, A. Schmidt, S. Banerjee, and C. Sirmans. Nonstationary multivariate process modeling through spatially varying coregionalization. *An Official Journal of the Spanish Society of Statistics and Operations Research*, 13(2):263–312, 2004.

[42] M. Zhang and Z. Zhou. ML-KNN: A lazy learning approach to multi-label learning. *Pattern Recognition*, 40(7):2038–2048, 2007.

[43] E. Bonilla, K.M. Chai, and C. Williams. Multi-task Gaussian process prediction. In *NIPS 08: Proceedings of the 21st Annual Conference on Neural Information Processing Systems*. 2008.

[44] Y. W. Teh and M. Seeger. Semiparametric latent factor models. In *Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics (AISTATS 2005)*, 2005.

[45] O. Chapelle, B. Scholkopf, and A. Zien, editors. *Semi-Supervised Learning*. MIT Press, 2006.

[46] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In *Proceedings of the 20th International Conference on Machine Learning*, 2003.

[47] D. Zhou, O. Bousquet, T. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *NIPS 03: Proceedings of the 16th Annual Conference on Neural Information Processing Systems*, 2003.

[48] F. Wang and C. Zhang. Label propagation through linear neighborhoods. In *Proceedings of the 23rd International Conference on Machine Learning*, 2006.

[49] X. Zhu and A. B. Goldberg. *Introduction to Semi-Supervised Learning*. Morgan & Claypool Publishers, 2009.

[50] M. Szummer and T. Jaakkola. Partially labeled classification with markov random walks. In *NIPS 01: Proceedings of the 14th Annual Conference on Neural Information Processing Systems*, 2001.

[51] U. von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

[52] F. R. K. Chung. *Spectral Graph Theory.* American Mathematical Society, 1997.

[53] B. Mohar. The laplacian spectrum of graphs. In *Graph Theory, Combinatorics, and Applications*, pages 871–898. Wiley, 1991.

[54] P. G. Doyle and L. J. Snell. Random walks and electric networks, Jan 2000.

[55] M. Belkin, I. Matveeva, and P. Niyogi. Regularization and semi-supervised learning on large graphs. In *Proceedings of the 18th Annual Conference on Learning Theory*, 2004.

[56] O. Chapelle, J. Weston, and B. Schölkopf. Cluster kernels for semi-supervised learning. In *NIPS 03: Proceedings of the 16th Annual Conference on Neural Information Processing Systems*, volume 15 of *NIPS*, pages 585–592, 2003.

[57] C. Ding, H. D. Simon, R. Jin, and T. Li. A learning framework using green's function and kernel regularization with application to recommender system. In *KDD '07: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 260–269. ACM, 2007.

[58] L. Hagen and A.B. Kahng. New spectral methods for ratio cut partitioning and clustering. In *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, pages 1074–1085, 1992.

[59] J. Shi and J. Malik. Normalized cuts and image segmentation. In *CVPR '97: Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)*, page 731, Washington, DC, USA, 1997. IEEE Computer Society.

[60] A. Blum and S. Chawla. Learning from labeled and unlabeled data using graph mincuts. In *Proc. 18th International Conf. on Machine Learning*, pages 19–26. Morgan Kaufmann, San Francisco, CA, 2001.

[61] A. Blum, J. Lafferty, M. R. Rwebangira, and R. Reddy. Semi-supervised learning using randomized mincuts. In *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, page 13. ACM, 2004.

[62] T. Joachims. Transductive learning via spectral graph partitioning. In *Proceedings of the 20th International Conference on Machine Learning*, 2003.

[63] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, December 2000.

[64] X. Yang, H. Fu, H. Zha, and J. Barlow. Semi-supervised nonlinear dimensionality reduction. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pages 1065–1072. ACM Press, 2006.

[65] A. Banerjee, I. S. Dhillon, J. Ghosh, and S. Sra. Clustering on the unit hypersphere using von mises-fisher distributions. *J. Mach. Learn. Res.*, 6:1345–1382, 2005.

[66] A. Skabar, D. Wollersheim, and T. Whitfort. Multi-label classification of gene function using mlps. In *IJCNN*, 2006.

[67] M. Wang, X. Zhou, and T. Chua. Automatic image annotation via local multi-label classification. In *CIVR*, 2008.

[68] A. McCallum. Multi-label text classification with a mixture model trained by EM. In *AAAI Workshop on Text Learning*, 1999.

[69] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *NIPS 07: Proceedings of the 20th Annual Conference on Neural Information Processing Systems*, 2007.

[70] A. Elisseeff and J. Weston. A kernel method for multi-labelled classification. In *ACM SIGIR*, 2005.

[71] L. Tang, J. Chen, and J. Ye. On multiple kernel learning with multiple labels. In *IJCAI 09: Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, 2009.

[72] P. Rai and H. Daume. Multi-label prediction via sparse infinite CCA. In *NIPS 09: Proceedings of the 22nd Annual Conference on Neural Information Processing Systems*, 2009.

[73] Y. Song, L. Zhang, and L. Giles. A sparse gaussian processes classification framework for fast tag suggestions. In *CIKM*, 2008.

[74] W. Cheng and E. Hüllermeier. Combining instance-based learning and logistic regression for multilabel classification. *Mach. Learn.*, 76(2-3):211–225, 2009.

[75] Y. Zhang and Z. Zhou. Multi-label dimensionality reduction via dependence maximization. In *AAAI*, 2008.

[76] S. Ji and J. Ye. Linear dimensionality reduction for multi-label classification. In *IJCAI 09: Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, 2009.

[77] G. Chen, Y. Song, F. Wang, and C. Zhang. Semi-supervised multi-label learning by solving a sylvester equation. In *SDM*, 2008.

[78] C. G. M. Snoek, M. Worring, J. C. Van Gemert, J. Geusebroek, and A. W. M. Smeulders. The challenge problem for automated detection of 101 semantic concepts in multimedia. In *In Proceedings of the ACM International Conference on Multimedia*, pages 421–430. ACM Press, 2006.

[79] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown. Learning multi-label scene classification. *Pattern Recognition*, 37(9):1757–1771, 2004.

[80] K. Trohidis, G. Tsoumakas, G. Kalliris, and I. Vlahavas. Multilabel classification of music into emotions. In *ISMIR*, 2008.

[81] J. Read, B. Pfahringer, G. Holmes, and E. Frank. Classifier chains for multi-label classification. In *Proc 13th European Conference on Principles and Practice of Knowledge Discovery in Databases and 20th European Conference on Machine Learning*, 2009.

[82] R. Silva, W. Chu, and Z. Ghahramani. Hidden common cause relations in relational learning. In *NIPS 07: Proceedings of the 20th Annual Conference on Neural Information Processing Systems*, 2007.

[83] Z. Xu, K. Kersting, and V. Tresp. Multi-relational learning with gaussian processes. In *IJCAI 09: Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, 2009.

[84] D. M. Higdon. Space and space-time modelling using process convolutions. In *Quantitative methods for current environmental issues*, pages 37–56. Springer-Verlag, 2002.

[85] G. Golub et al. A Hessenberg-Schur method for the problem AX + XB= C. *IEEE Transactions on Automatic Control*, 24(6), 1979.

[86] E. L. Wachspress. Iterative solution of the Lyapunov matrix equation. *Applied Mathematics Letters*, 1(1):87–90, 1988.

[87] A. J. Laub. *Matrix Analysis for Scientists and Engineers*. SIAM, 2005.

[88] Z. Zhou and L. Zhang. Multi-instance multi-label learning with application to scene classification. In *NIPS 06: Proceedings of the 19th Annual Conference on Neural Information Processing Systems*, 2006.

[89] G. Pandey, C. Myers, and V. Kumar. Incorporating functional inter-relationships into protein function prediction algorithms. *BMC Bioinformatics*, 10(1):142+, 2009.

[90] P. Boyle and M. Frean. Dependent Gaussian processes. In *NIPS 05: Proceedings of the 18th Annual Conference on Neural Information Processing Systems*, volume 17, pages 217–224. MIT Press, 2005.

[91] M. Alvarez and N. D. Lawrence. Sparse convolved gaussian processes for multi-output regression. In *NIPS 08: Proceedings of the 21st Annual Conference on Neural Information Processing Systems*, pages 57–64, 2008.

[92] W. Chu, V. Sindhwani, Z. Ghahramani, and S. Sathiya Keerthi. Relational learning with Gaussian processes. In *NIPS 07: Proceedings of the 20th Annual Conference on Neural Information Processing Systems*. 2007.

[93] D. M. Blei et al. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, January 2003.

[94] J. Chang and D. Blei. Relational topic models for document networks. In *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, 2009.

[95] J. Quiñonero Candela and C. E. Rasmussen. A unifying view of sparse approximate gaussian process regression. *Journal of Machine Learning Research*, 6:1939–1959, 2005.

[96] C. Rasmussen and C. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2005.

[97] T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proc Natl Acad Sci U S A*, 101:5228–5235, 2004.

[98] R. H. Bartels and G. W. Stewart. Solution of the matrix equation AX + XB = C. *Commun. ACM*, 15(9):820–826, 1972.

[99] N.D. Lawrence, M. Seeger, and R. Herbrich. Fast sparse Gaussian process methods: The informative vector machine. In *NIPS 02: Proceedings of the 15th Annual Conference on Neural Information Processing Systems*, pages 609–616. MIT Press, 2002.

[100] L. Foster, A. Waagen, N. Aijaz, M. Hurley, A. Luis, J. Rinsky, C. Satyavolu, M.J. Way, P. Gazis, and A. Srivastava. Stable and efficient gaussian process calculations. *J. Mach. Learn. Res.*, 10:857–882, 2009.