A PROBABILISTIC FRAMEWORK FOR LEARNING TASK RELATIONSHIPS IN MULTI-TASK LEARNING

by

YU ZHANG

A Thesis Submitted to
The Hong Kong University of Science and Technology
in Partial Fulfillment of the Requirements for
the Degree of Doctor of Philosophy
in Computer Science and Engineering

August 2011, Hong Kong

Authorization

I hereby declare that I am the sole author of the thesis.

I authorize the Hong Kong University of Science and Technology to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize the Hong Kong University of Science and Technology to reproduce the thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

_____

YU ZHANG

A PROBABILISTIC FRAMEWORK FOR LEARNING TASK RELATIONSHIPS IN MULTI-TASK LEARNING

by

YU ZHANG

This is to certify that I have examined the above Ph.D. thesis
and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by
the thesis examination committee have been made.

PROF. DIT-YAN YEUNG, THESIS SUPERVISOR

PROF. MOUNIR HAMDI, HEAD OF DEPARTMENT

Department of Computer Science and Engineering
29 August 2011

ACKNOWLEDGMENTS

First and foremost, I would like to thank my supervisor, Prof. Dit-Yan Yeung, for his support throughout my PhD study at the HKUST. Without his guidance, encouragement, and kindness, this thesis would never have been possible.

I would like to thank my thesis committee members, Prof. Eric Xing (from Machine Learning Department, Carnegie Mellon University), Prof. Mike Ka Pui So (from Department of Information Systems, Business Statistics and Operations Management, HKUST), Prof. Nevin L. Zhang, Prof. Qiang Yang, and Prof. Dit-Yan Yeung, for their insightful comments and suggestions.

I would like to thank the members of Prof. Yeung's group for their friendship, encouragement, support and collaboration, such as Gang Wang, Guang Dai, Chris Kui Jia, Zhihua Zhang, Wu-Jun Li, Yang Ruan, Jingni Chen, Yi Zhen, Craig Yu, Tony Ho, Emprise Chan, Yan-Ming Zhang, Guoqiang Zhong, and Deming Zhai. My many friends at the HKUST have given me an unforgettable memory of both research and everyday life. They are: Bin Cao, Qian Xu, Nathan Nan Liu, Weizhu Chen, Wei Bi, Sinno Jialin Pan, Vincent Wenchen Zheng, Evan Wei Xiang, Weike Pan, Derek Hao Hu, Si Shen and Tengfei Liu.

I would also like to thank Prof. Eric Xing for his insightful discussions and guidance on research during my visit to Carnegie Mellon University. My many friends helped me in many ways to adapt to life in the USA. They include Jun Zhu, Li Nao, Yanlin Li, Yin Zhang, Pingzhong Tang, Yuandong Tian, Lei Li, Junming Yin, Haiyi Zhu, Cao Shen, and Yu Sheng.

Finally, I am deeply indebted to my family especially my mother whose love and support have been the source of my courage.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

A PROBABILISTIC FRAMEWORK FOR LEARNING TASK RELATIONSHIPS IN MULTI-TASK LEARNING

by

YU ZHANG

Department of Computer Science and Engineering

The Hong Kong University of Science and Technology

ABSTRACT

For many real-world machine learning applications, labeled data is costly because the data labeling process is laborious and time consuming. As a consequence, only limited labeled data is available for model training, leading to the so-called labeled data deficiency problem. In the machine learning research community, several directions have been pursued to address this problem. Among these efforts, a promising direction is *multi-task learning* which is a learning paradigm that seeks to boost the generalization performance of a model on a learning task with the help of some other related tasks. This learning paradigm has been inspired by human learning activities in that people often apply the knowledge gained from previous learning tasks to help learn a new task more efficiently and effectively. Of the several approaches proposed in previous research for multi-task learning, a relatively less studied yet very promising approach is based on automatically learning the relationships among tasks from data.

In this thesis, we first propose a powerful probabilistic framework for multi-task learning based on the task relationship learning approach. The main novelty of our framework lies in the use of a matrix variate prior with parameters that model task relationships. Based on this general multi-task learning framework, we then propose four specific methods, namely, *multi-task relationship learning* (MTRL), *multi-task generalized t process* (MTGTP), *multi-task high-order task relationship learning* (MTHOL), and *probabilistic multi-task feature selection* (PMTFS). By utilizing a matrix variate normal distribution as a prior on the model parameters of all tasks, MTRL can be formulated efficiently as a convex optimization problem. On the other hand, MTGTP is a Bayesian method that models the task covariance matrix as a random matrix with

an inverse-Wishart prior and integrates it out to achieve Bayesian model averaging to improve the generalization performance. With MTRL as a base, MTHOL provides a generalization that learns high-order task relationships and model parameters. Unlike MTRL, MTGTP and MTHOL which are for standard multi-task classification or regression problems, PMTFS addresses the feature selection problem under the multi-task setting by incorporating the learning of task relationships. Besides conducting experimental validation of the proposed methods on several data sets for multi-task learning, we also investigate in detail a collaborative filtering application under the multi-task setting. Through both theoretical and empirical studies on the several methods proposed, we show that task relationship learning is a very promising approach for multi-task learning and related learning problems.

CHAPTER 1

INTRODUCTION

In many real-world applications, the cost of acquiring labeled data is very high since it may require the effort of domain experts and is time consuming. Therefore in many applications the use of labeled data is limited and this is called the 'labeled data deficiency problem'. In the machine learning research community, many efforts have been devoted to alleviating this problem by developing methods that do not rely heavily on labeled data. For example, even though labeled data is limited, an abundant supply of unlabeled data is available at very low cost, for instance, on the Internet, there exist enormous unlabeled web pages and images. Two learning frameworks, active learning (Cohn et al., 1994, 1996) and semi-supervised learning (Chapelle et al., 2006), have been developed to utilize the information from the unlabeled data to reduce the dependency on labeled data. Even though these two frameworks can utilize unlabeled data, the ways they utilize it are totally different. Semi-supervised learning usually utilizes the geometric information contained in the unlabeled data to help improve the performance of some classification or regression tasks. The whole process is automatic and requires little human effort. On the other hand, unlike many conventional machine learning methods which wait passively for labeled data to be provided in order to start the learning process, active learning takes a more active approach by selecting unlabeled data points to query some oracle or domain expert.

Besides utilizing unlabeled data, there is another way to improve the performance when the labeled data is scarce, for example, using the labeled information in other related tasks. This is called multi-task learning (Caruana, 1997; Baxter, 1997; Thrun, 1995) which seeks to improve the performance of one learning task with the help of some other related tasks. The origins of multi-task learning date back to studies in psychology and cognitive science, and multi-task learning could be viewed as a way to mimic the human learning ability where people often learn a new task by re-using the knowledge gained from previous learning tasks.

In the following, we first introduce some basic concepts about multi-task learning. Then, we present our motivation and summarize the main contributions of this thesis. Finally, we outline the organization of the whole thesis.

1.1 Multi-Task Learning

Multi-task learning (Caruana, 1997; Baxter, 1997; Thrun, 1995) is a learning paradigm which seeks to improve the generalization performance of a learning task with the help of some other

related tasks. This learning paradigm has been inspired by human learning activities in that people often apply knowledge gained from previous learning tasks to help learn a new task. For example, a baby first learns to recognize human faces and later uses this knowledge to help it learn to recognize other objects. By learning tasks simultaneously, and meanwhile exploiting their intrinsic relatedness, informative domain knowledge is shared across the tasks and thus facilitates individual task learning. It is particularly desirable to share such knowledge across tasks when there are a number of related tasks but only limited training data available for each one.

Multi-task learning can be formulated under two different settings: *symmetric* and *asymmetric* (Xue et al., 2007). While symmetric multi-task learning seeks to improve the performance of all tasks simultaneously, the objective of asymmetric multi-task learning is to improve the performance of some target task using information from the source tasks, typically after the source tasks have been learned using some symmetric multi-task learning method. In this sense, asymmetric multi-task learning is related to *transfer learning* (Pan & Yang, 2010), but the major difference is that the source tasks are still learned simultaneously in asymmetric multi-task learning but they are learned independently in transfer learning.

1.2 Motivation

Since multi-task learning seeks to improve the performance of a task with the help of other related tasks, a central issue is to characterize the relationships between tasks accurately. Given the training data in multiple tasks, there are two important aspects that distinguish between different methods for characterizing the task relationships. The first aspect is on *what* task relationships can be represented by a method. Generally speaking there are three types of pairwise task relationships: positive task correlation, negative task correlation, and task unrelatedness (corresponding to outlier tasks). Positive task correlation is very useful for characterizing task relationships since similar tasks are likely to have similar model parameters. For negative task correlation, since the model parameters of two tasks with negative correlation are more likely to be dissimilar, knowing that two tasks are negatively correlated can help to reduce the search space of the model parameters. As for task unrelatedness, identifying outlier tasks can prevent them from impairing the performance of other tasks since outlier tasks are unrelated to other tasks. The second aspect is on *how* to obtain the relationships, either from the model assumption or automatically learned from data. Obviously, learning the task relationships from data automatically is the more favorable option because the model assumption adopted may be incorrect and, even worse, it is not easy to verify the correctness of the assumption from data. So a powerful multi-task learning method could learn the three task relationships from data automatically.

A powerful multi-task learning method based on the Gaussian process (GP) (Bonilla et al., 2007) provides an approach to model and learn task relationships in the form of a task covariance matrix. A task covariance matrix can model all three types of task relationships: positive task correlation, negative task correlation, and task unrelatedness. However, although this method provides a powerful way to model task relationships, learning about the task covariance matrix gives rise to a non-convex optimization problem which is sensitive to parameter initialization. When the number of tasks is large, the authors propose using low-rank approximation (Bonilla et al., 2007) which will then weaken the expressive power of the task covariance matrix. Moreover, since the method is based on the Gaussian process (Rasmussen & Williams, 2006), scaling it to large data sets poses a serious computational challenge.

Some limitations in (Bonilla et al., 2007) have motivated us to develop some new methods which can overcome them to some extent. Moreover, since multi-task feature selection can find common features for different tasks which is appealing for some applications such as bioinformatics, we hope to extend the idea of learning task relationships to multi-task feature selection. We also hope to find some real-world applications for multi-task learning in some areas, such as information retrieval.

1.3 Main Contributions

In this thesis, we propose a simple probabilistic framework to learn task relationship in multi-task learning:

$$y_j^i \sim p(r(\boldsymbol{\theta}_i, \mathbf{x}_j^i)), \text{ for } i = 1, \ldots, m, \ j = 1, \ldots, n_i \tag{1.1}$$

$$\boldsymbol{\Theta} \sim q(\boldsymbol{\Theta}; \boldsymbol{\Gamma}), \tag{1.2}$$

where $\mathbf{x}_j^i$ is the jth data point in the ith task with its output y_j^i, m is the number of learning tasks, n_i is the number of data points in the ith task, $\boldsymbol{\theta}_i$ is the model parameter of the ith task, $\boldsymbol{\Theta}$ is the union of $\boldsymbol{\theta}_i$ for $i = 1, \ldots, m$, and $q(\boldsymbol{\Theta}; \boldsymbol{\Gamma})$ defines some matrix variate distribution (Gupta & Nagar, 2000) for $\boldsymbol{\Theta}$ parameterized by $\boldsymbol{\Gamma}$. $\boldsymbol{\Gamma}$ measures the task relationships in some form such as covariance matrix. Eq. (1.2) defines the prior for $\boldsymbol{\Theta}$ and Eq. (1.1) describes the likelihood on $\mathbf{x}_j^i$.

By utilizing this probabilistic framework, we developed some new multi-task learning methods which are introduced in the following:

- Multi-task relationship learning (MTRL): We propose a probabilistic approach to learning the relationships between tasks in multi-task learning. This approach can be viewed as a novel generalization of the probabilistic formulation for single-task learning. Besides modeling positive task correlation, our approach, called *multi-task relationship learning* (MTRL), can also describe negative task correlation and identify outlier tasks based

on the same underlying principle. By utilizing a matrix variate normal distribution as a prior on the model parameters of all tasks, we can obtain a concrete method with a convex objective function. For efficiency, we use an alternating method to learn the optimal model parameters for each task as well as the relationships between tasks. We study MTRL in the symmetric multi-task learning setting and then generalize it to the asymmetric setting as well. We also discuss some variants of the probabilistic approach to demonstrate the use of other matrix variate priors for learning task relationships. Moreover, to gain more insight into our model, we also study the relationships between MTRL and some existing multi-task learning methods. Experiments conducted on a toy problem as well as several benchmark data sets demonstrate the effectiveness of MTRL as well as its high interpretability revealed by the task covariance matrix.

- Multi-task generalized t process (MTGTP): Since learning the task covariance matrix directly in (Bonilla et al., 2007) has both computational and representational drawbacks, here we propose a Bayesian extension by modeling the task covariance matrix as a random matrix with an inverse-Wishart prior and integrating it out to achieve Bayesian model averaging. To make the computation feasible, we first give an alternative weight-space view of the multi-task GP model in (Bonilla et al., 2007) and then integrate out the task covariance matrix in the model, leading to a multi-task generalized t process. For the likelihood, we use a generalized t noise model which, together with the generalized t process prior, brings about the robustness advantage as well as an analytical form for the marginal likelihood. In order to specify the inverse-Wishart prior, we use the maximum mean discrepancy (MMD) statistic to estimate the parameter matrix of the inverse-Wishart prior. Moreover, we investigate some theoretical properties of MTGTP, such as its asymptotic analysis and learning curve. Comparative experimental studies on two common multi-task learning applications show very promising results.

- Multi-Task High-Order Task Relationship Learning (MTHOL): Nevertheless, the existing task relationship learning methods (Bonilla et al., 2007; Zhang & Yeung, 2010a; Zhang et al., 2010) only model and learn pairwise task relationships. Some relationships between tasks found in real-world applications may be more complicated than pairwise relationships can characterize. This motivates us to explore the use of high-order task relationships for multi-task learning. We analyze the MTRL method and propose an alternative formulation for the model based on a matrix variate probability distribution. The new formulation allows us to generalize, though in a nontrivial way, the use of pairwise task relationships to high-order task relationships, leading to a new prior for the model parameters of different tasks. We then propose a new model which we refer to as Multi-Task High-Order relationship Learning (MTHOL).

- Probabilistic Multi-Task Feature Selection (PMTFS): Multi-task feature selection aims

to find the common useful features for all learning tasks at hand by utilizing $l_{1,2}$ and $l_{1,\infty}$ norms under the regularization framework. However, an underlying assumption of multi-task feature selection is that all tasks are similar and share the same features. This assumption may not be correct in practice because outlier tasks may exist (that is, tasks that are not related to all other tasks) or tasks with negative correlation (that is, tasks that are negatively correlated with some other tasks). Motivated by this observation, we study the problem of learning the task relationships in the context of multi-task feature selection. Instead of choosing between specific choices such as the $l_{1,2}$ and $l_{1,\infty}$ norms, we consider a family of $l_{1,q}$ norms for multi-task feature selection where $1 < q \leq \infty$. Using the $l_{1,q}$ norm, we formulate the general multi-task feature selection problem and give it a probabilistic interpretation. Based on this probabilistic interpretation, we have developed a probabilistic formulation using a noninformative prior called the Jeffreys prior (Gelman et al., 2003), and devised an expectation-maximization (EM) algorithm to learn all model parameters, including q, automatically. Moreover, we propose to use a matrix variate generalized normal prior for the model parameters to learn the relationships between tasks.

- Multi-Domain Collaborative Filtering (MCF): Collaborative filtering (CF) is an effective recommendation approach based on the intuitive idea that the preference of a user can be predicted by exploiting the information about other users which share similar interests. In particular, CF techniques exploit users past activities, such as transaction histories or expressed product satisfaction ratings, to predict their future activities. Even though CF methods have achieved great successes in recommendation applications, some problems which limit their performance still exist. A big challenge and our focus here is the data sparsity problem (Su & Khoshgoftaar, 2009) which means that the rating matrix is extremely sparse. In particular, we consider a multi-domain CF (MCF) problem which jointly models a collection of rating prediction tasks arising from multiple domains. For example, different product or service categories such as books and electronics naturally constitute different domains. This is a multi-task learning problem where each task corresponds to a CF problem on a single domain. By exploiting the correlation between rating prediction problems in different domains, we can transfer the shared knowledge among similar domains to alleviate the data sparsity problem and therefore improve the rating prediction performance in all domains. Specifically, we propose a probabilistic model which uses probabilistic matrix factorization (PMF) (Salakhutdinov & Mnih, 2007) for the rating prediction problem in each domain and allows that knowledge to be adaptively transferred across different domains by automatically learning the correlation between them. We also introduce the link function for different domains to correct their biases. Experiments conducted on several real-world applications demonstrate the effectiveness of our method.

1.4 Thesis Outline

The rest of this thesis is organized as follows:

Chapter 2 introduces some background knowledge, including a brief literature review of multi-task learning and matrix variate distributions (Gupta & Nagar, 2000).

Chapter 3 gives details of the multi-task relationship learning (MTRL) model, including its model formulation, parameter learning and experimental results on some applications.

Chapter 4 presents the multi-task generalized t process (MTGTP) model and discusses its model formulation and parameter learning.

Then in Chapter 5 the multi-task high-order task relationship learning (MTHOL) model in which the model formulation as well as the parameter learning problem is discussed.

Chapter 6 focusses on the probabilistic multi-task feature selection (PMTFS) model by discussing the model formulation and parameter learning problem.

Chapter 7 then presents an application of multi-task learning on collaborative filtering, which is multi-domain collaborative filtering (MCF) problem.

Chapter 8 concludes the thesis and proposes several potential directions for future pursuit.

1.5 Notations

Some commonly used notations in this thesis are listed in Table 1.1.

Table 1.1: Notations

Notation	Physical Meaning
$\mathbb{R}$	real space
$\mathbb{R}^d$	d-dimensional Euclidean space
$\mathbf{M} \in \mathbb{R}^{m \times n}$	matrix of size $m \times n$ denoted by a boldface uppercase letter
$\mathbf{z} \in \mathbb{R}^n$	n-dimensional (column) vector denoted by a boldface lowercase letter
a	real scalar denoted by an italic lowercase letter
$\mathbf{M}^T$	transpose of a matrix $\mathbf{M}$
$\mathbf{M}^{-1}$	inverse of a square matrix $\mathbf{M}$
$\mathbf{0}$	zero vector or matrix with appropriate size
$\mathbf{0}_d$	$d \times 1$ zero vector
$\mathbf{0}_{a \times b}$	$a \times b$ zero matrix
$\mathbf{1}$	vector or matrix of all 1's with appropriate size
$\mathbf{1}_d$	$d \times 1$ vector of all 1's
$\mathbf{1}_{a \times b}$	$a \times b$ matrix of all 1's
$\mathbf{I}$	identity matrix with appropriate size
$\mathbf{I}_n$	identity matrix of size $n \times n$
$\text{tr}(\mathbf{M})$	trace of a square matrix $\mathbf{M}$
$\mathbf{M} \succeq \mathbf{0}$	$\mathbf{M}$ is a positive semidefinite (PSD) matrix
$\mathbf{M} \succ \mathbf{0}$	$\mathbf{M}$ is a positive definite (PD) matrix
$\mathbf{A} \succ (\succeq) \mathbf{B}$	$\mathbf{A} - \mathbf{B} \succ (\succeq) \mathbf{0}$
$\|\mathbf{z}\|_1$	l_1 norm of a vector $\mathbf{z}$
$\|\mathbf{z}\|_2$	l_2 norm of a vector $\mathbf{z}$
$\|\mathbf{z}\|_\infty$	l_∞ norm of a vector $\mathbf{z}$
$\|\mathbf{M}\|_F$	Frobenius norm of a matrix $\mathbf{M}$
$\mathcal{N}(\cdot)$	univariate or multivariate normal distributions
$\mathcal{MN}(\cdot)$	matrix variate normal distribution
$\mathbf{A} \otimes \mathbf{B}$	Kronecker product of $\mathbf{A}$ and $\mathbf{B}$
$\text{diag}(\mathbf{z})$	convert a vector $\mathbf{z}$ into a diagonal matrix
$\|\mathbf{M}\|$	determinant of a matrix $\mathbf{M}$
$\|a\|$	absolute value of a real scalar a
$\Gamma(\cdot)$	Gamma function
$\Gamma_n(\cdot)$	multivariate Gamma function

CHAPTER 2

BACKGROUND

In this chapter, we briefly introduce some background knowledge related to the whole thesis, including a brief survey of multi-task learning and matrix variate distributions (Gupta & Nagar, 2000).

2.1 A Brief Survey of Multi-Task Learning

The aim of multi-task learning is to improve the performance of each task with the help of other related tasks. One important issue is to find a way to share information across different tasks, for example, sharing via data representation or model representation. According to different modes of information sharing, we can divide all existing multi-task learning methods into several categories which are discussed in detail in the following section. Besides different models for multi-task learning, there are some theoretical studies to analyze some problems in multi-task learning, for example, sample complexity. Moreover, multi-task learning has applications in different areas, for example, computer vision (Heisele et al., 2001; Lapedriza et al., 2008; Quattoni et al., 2008; Ahmed et al., 2008; Kienzle & Chellapilla, 2006; Torralba et al., 2004; An et al., 2008; Zhang & Yeung, 2010c), information retrieval (Yu et al., 2004; Yu & Tresp, 2005; Cao et al., 2010; Zhang et al., 2010b; Raina et al., 2006), bioinformatics (Bickel et al., 2008; Xu et al., 2010; Liu et al., 2010; Zhang et al., 2010a; Puniyani et al., 2010; Lee et al., 2010) and finance (Ghosn & Bengio, 1996).

In the following, we briefly review some representative multi-task learning models, theoretical results and applications.

2.1.1 Models

In this section, we review some existing methods for multi-task learning, which can be classified into five categories: common representation approach, task regularization approach, task clustering approach, hierarchical Bayesian approach and task relationship learning approach.

Common Representation Approach

The "representation" here mostly refers to data representation. Neural networks are the earliest studied models in this category. Note that a multi-task neural network is just a conventional

multilayer feed-forward neural network that captures the commonality of the tasks. In a multi-task neural network, the hidden layer corresponds to common data representation after some linear or nonlinear transformation. Similar to conventional neural networks, the back propagation (BP) algorithm can be used to learn the model parameters. Note that this method not only solves the multi-output problems but also the general multi-task learning problems. Following this strategy, Liao and Carin (2005) extend the radial basis function networks for multi-task learning. In this work, the hidden layer is also treated as a common representation for each task. Since a radial basis function network has an analytical solution, it can use the data from multiple tasks to determine the form of the RBF function in the hidden layer via active learning. Different from multi-task neural networks, Silver et al. (2008) propose a context-sensitive neural network method for multi-task learning, in which each data point is first expanded with a task indicator vector and the expanded data is trained by a conventional neural network with a single output node. Surprisingly, the performance of context-sensitive neural networks is better than that of multi-task neural networks. However, the same strategy does not work for decision tree and support vector machine (SVM) models.

Argyriou et al. (Argyriou et al., 2006, 2008a) propose a multi-task feature learning method to learn the common representation for multi-task learning under a regularization framework:

$$\xi(\mathbf{U}, \mathbf{A}) = \sum_{i=1}^{k} \sum_{j=1}^{n_i} l(y_j^i, \mathbf{a}_i^T \mathbf{U}^T \mathbf{x}_j^i) + \gamma ||\mathbf{A}||_{2,1}^2 \tag{2.1}$$

where $l(\cdot, \cdot)$ denotes the loss function, $\mathbf{U}$ is the common transformation to find the common representation, $\mathbf{a}_i$ is the model parameters for task T_i after the transformation, $\mathbf{A} = (\mathbf{a}_1, \ldots, \mathbf{a}_k)$, $\mathbf{A}^j$ denotes jth row of $\mathbf{A}$, and $||\mathbf{A}||_{2,1} = \sum_j ||\mathbf{A}^j||_2$ denotes the $l_{1,2}$ norm of a matrix $\mathbf{A}$. The $l_{1,2}$ norm in regularization function will lead to row sparsity where all the elements in one whole rows of $\mathbf{A}$ are zero, which has the effect of feature selection on $\mathbf{U}^T \mathbf{x}_j^i$, due to the sparsity property of l_1 norm. An alternating optimization method is used to learn the model parameters. This alternating optimization method can be seen as a two-step learning method: in the first step, it learns each task individually and the second step learns the common representation for all tasks. Argyriou et al. (2007) extend this method to a more general formulation.

The common representation in multi-task neural networks and multi-task feature learning is derived from some transformation on the original data representation. Different from them, Obozinski et al. (2006) propose a feature selection method for multi-task learning, whose formulation is similar to Eq. (2.1) but without learning the transformation matrix $\mathbf{U}$. This can be viewed as a multi-task extension of LASSO (Tibshirani, 1996). Similar to (Obozinski et al., 2006), some probabilistic multi-task feature selection methods in (Xiong et al., 2007; Bi et al., 2008) are proposed for multi-task learning. These two works first learn a common representation for all tasks by using an l_1-norm based feature selection method and then learn the model parameters for each task independently. These methods can be seen as a probabilistic model of

(Obozinski et al., 2006) from the perspective of hierarchical Bayesian modeling. Jebara (2004) extends the maximum entropy discrimination (MED) (Jaakkola et al., 1999) method to multi-task settings. In (Jebara, 2004), feature selection and kernel selection problems are discussed under the framework of MED for multi-task learning, where the feature representation and the combination coefficients for the kernels are shared by all tasks.

Task Regularization Approach

The task regularization approach includes the multi-task learning methods formulated under the regularization framework, whose objective function usually comprises two terms: an empirical loss on the training data of each task and a regularization term to encode the relationships between the tasks.

Evgeniou and Pontil (2004) propose a multi-task extension of the SVM model, which minimizes the following objective function

$$\xi(\{\mathbf{w}_i\}) = \sum_{i=1}^{k}\sum_{j=1}^{n_i} l(y_j^i, \mathbf{w}_i^T\mathbf{x}_j^i) + \lambda_1 \sum_{i=1}^{k} ||\mathbf{w}_i||_2^2 + \lambda_2 \sum_{i=1}^{k} ||\mathbf{w}_i - \frac{1}{k}\sum_{j=1}^{k}\mathbf{w}_j||_2^2. \tag{2.2}$$

The first term of Eq. (2.2) denotes the empirical loss on the training data and second one is to penalize the complexity of the model parameters in each task, which are the same as those of single-task SVMs. Moreover, the third term is designed to penalize large deviation between each parameter vector of each task and the mean parameter vector of all tasks, which enforces the parameter vectors in all tasks to be similar to each other. Evgeniou and Pontil (2004) also show that its dual formulation is identical to that of conventional SVMs with the only modification on the definition of the kernel function used. Moreover, Parameswaran and Weinberger (2010) also propose a large-margin multi-task metric learning method by utilizing the same idea.

In (Evgeniou et al., 2005), they extend their own work in (Evgeniou & Pontil, 2004) and propose a multi-task kernel, by which the formulations of the multi-task kernel methods can be reduced to those of the single-task kernel methods. Moreover, this work extends the regularized method proposed in (Evgeniou & Pontil, 2004) and proposes two more regularized methods for multi-task learning with the objective functions as

$$\xi(\{\mathbf{w}_i\}) = \sum_{i=1}^{m}\sum_{j=1}^{n_i} l(y_j^i, \mathbf{w}_i^T\mathbf{x}_j^i) + \sum_{j=1}^{k}\sum_{i\in C_j}(\lambda_1||\mathbf{w}_i - \bar{\mathbf{w}}_j||_2^2 + \lambda||\bar{\mathbf{w}}_j||_2^2), \tag{2.3}$$

where C_j denotes the set of tasks belonging to the jth task cluster, k is the number of clusters and $\bar{\mathbf{w}}_j$ denotes the mean of the model parameters of all tasks in the jth cluster, and

$$\xi(\{\mathbf{w}_i\}) = \sum_{i=1}^{m}\sum_{j=1}^{n_i} l(y_j^i, \mathbf{w}_i^T\mathbf{x}_j^i) + \lambda \sum_{i=1}^{m}\sum_{j=1}^{m} A_{ij}||\mathbf{w}_i - \mathbf{w}_j||_2^2, \tag{2.4}$$

where A_{ij} denotes the relatedness between task T_i and T_j. The regularization term in Eq. (2.3) enforces that the model parameters of the tasks belonging to the same cluster to be similar when given the cluster structure $\{C_i\}$. However, how to determinate the cluster structure is still a problem. Moreover, when we are given the relatedness between different tasks, we can construct a similarity graph and use the Laplacian matrix as a regularization term just as Eq. (2.4) does. However, it is not easy to obtain the information about the task relatedness in most real-world applications.

Similar to (Evgeniou et al., 2005), Kato et al. (2007) propose a multi-task learning method to utilize an unweighted task network to encode the relatedness between tasks. The formulation can be written as

$$\begin{aligned} \min \quad & \xi(\{\mathbf{w}_i\}) = \sum_{i=1}^{k}\sum_{j=1}^{n_i} l(y_j^i, \mathbf{w}_i^T\mathbf{x}_j^i) + \lambda_1 \sum_{i=1}^{k} ||\mathbf{w}_i||_2^2 + \lambda_2\rho \\ s.t. \quad & ||\mathbf{w}_{i_k} - \mathbf{w}_{j_k}||_2^2 \leq \rho \text{ for } T_{i_k} \text{and } T_{j_k} \text{ are related,} \end{aligned}$$

which means the difference between the parameter vectors of any two related tasks is small.

Moreover, in (Ando & Zhang, 2005), the model parameters of different tasks are assumed to share a common subspace. Since the original optimization problem is non-convex, Chen et al. (2009) formulate it as a convex problem by changing the loss function and making some relaxation on the original formulation. Moreover, the works in (Chen et al., 2010; Jalali et al., 2010) extend this method by assuming that the shared subspace has some special structure, that is, sparsity.

Task Clustering Approach

The method in (Thrun & O'Sullivan, 1996) is a first task clustering method for multi-task learning. The main idea is to cluster all tasks into several clusters, where the tasks in one task cluster are assumed to share a similar data or model representation. The base learner in (Thrun & O'Sullivan, 1996) is a weighted k-Nearest-Neighbor (kNN) classifier, in which each feature is given a weight for computing the distance metric. This method first learns the distance metric for each task independently. The relatedness between two tasks is defined as the generalization performance of one task by using the other task's distance metric in the kNN classifier and then by using the task relatedness, all tasks are divided into several clusters, and finally the data points in the tasks corresponding to one task cluster are pooled together to learn an optimal distance metric for the kNN classifier.

Different from (Thrun & O'Sullivan, 1996), Bakker and Heskes (2003) propose a Bayesian multi-task neural network, whose structure is similar to that of a conventional multi-task neural network with the input-to-hidden-layer weights shared by all tasks. Different from the multi-task neural network, the hidden-layer-to-output-layer weights $\mathbf{A}_i$ for each task are different

from each other but have a common prior. So this model is a Bayesian model and BP algorithm is no longer suitable for parameter learning. Instead the maximal likelihood estimation (MLE) method is used to learn the parameters. A widely used common prior for $\mathbf{A}_i$ is a Gaussian prior:

$$\mathbf{A}_i \sim \mathcal{N}(\mathbf{m}, \mathbf{\Sigma}) \text{ for } i = 1, \ldots, k.$$

This prior has the effect of clustering all tasks into one cluster. This can be easily generalized to multiple clusters by using a mixture of Gaussians prior as

$$\mathbf{A}_i \sim \sum_{j=1}^{k} q_j \mathcal{N}(\mathbf{m}_j, \mathbf{\Sigma}_j)$$

where k is the number of components to be set by the users and q_j is the mixing proportion with $\sum_{j=1}^{k} q_j = 1$. Another generalization of this model is to use a task gating model in which the mixing proportions are task-dependent. Here any two tasks in one task cluster can have different model parameters but this is not the case in (Thrun & O'Sullivan, 1996).

The above discussed task clustering methods require that the number of clusters should be given as a priori, which is not easy to obtain for many real-world applications. To solve this problem, Xue et al. (2007) propose a multi-task learning method, which utilizes a nonparametric Bayesian model, Dirichlet Process (DP), as a basic mechanism to cluster all tasks without knowing the number of clusters in advance. For each task, the authors use the logit model to model the data likelihood as

$$p(y_j^i | \mathbf{x}_j^i, \mathbf{w}_i) = \sigma(y_j^i \mathbf{w}_i^T \mathbf{x}_j^i).$$

Then they add a DP prior on all $\mathbf{w}_i$'s as

$$\mathbf{w}_i \sim \text{DP}(\alpha_0, G_0)$$

where α_0 denotes the concentration parameter and G_0 denotes the base measure. An equivalent formulation is as follows

$$p(\mathbf{w}_i | \mathbf{w}_{-i}, \alpha_0, G_0) = \frac{\alpha_0}{k - 1 + \alpha_0} G_0 + \frac{1}{k - 1 + \alpha_0} \sum_{j=1, j \neq i}^{k} \delta_{\mathbf{w}_j}, \tag{2.5}$$

where $\mathbf{w}_{-i}$ denotes the set of all $\mathbf{w}_j$'s excluding $\mathbf{w}_i$ and $\delta_{\mathbf{w}_j}$ a distribution concentrated at the single point $\mathbf{w}_j$. Eq. (2.5) is also called Chinese Restaurant Processes (CRP). Eq. (2.5) can be described as a clustering process: given $\mathbf{w}_{-i}$, the ith task can choose to belong to one existing cluster, corresponding to the first term in the right hand of Eq. (2.5), or open a new cluster, corresponding to the second term. A variational Bayesian method (Bishop, 2006) is used to make inference.

Moreover, some regularized methods are proposed for the task clustering approach. For example, Jacob et al. (2008) propose a regularized method by defining a regularization term

inspired by the k-means clustering algorithm to learn the cluster structure. Similar to (Thrun & O'Sullivan, 1996; Bakker & Heskes, 2003), a limitation of this method is that the number of clusters should be given as a priori. Moreover, Argyriou et al. (2008b) extend the multi-task feature learning method to learn the cluster structure, in which the tasks in the same task cluster share the same representation.

Hierarchical Bayesian Approach

Hierarchical Bayesian models are well studied by the statistics research community and widely used in many applications. Heskes (1998) proposes a Bayesian multi-task neural network method for multi-task learning, in which the hidden-to-output weights for each task have a prior with the parameters shared by all tasks. This model is similar to that in (Bakker & Heskes, 2003). In (Heskes, 1998), an empirical Bayesian method is used to learn the parameters in this model. Heskes (2000) extends this method by assuming that the hidden-to-output weights for each task have a prior with a common covariance matrix shared by all tasks but different mean vectors for different tasks.

The method in (Minka & Picard, 1997) first utilizes GP for multi-task learning. Then Lawrence and Platt (2004) generalize the informative vector machine (IVM) in (Lawrence et al., 2002), a sparse extension of GP, to multi-task learning, where the kernel parameters are shared by all tasks. Thus, the formulation in multi-task IVM is similar to that of single-task IVM with the difference being that the covariance matrix in multi-task IVM is a block matrix, of which each block submatrix corresponds to the covariance matrix for one task.

Yu et al. (2005) propose a hierarchical Bayesian model for multi-task regression, which utilizes a GP for each task. The mean vector and the covariance matrix in the GP priors share a common conjugate prior. For parameter learning, an EM algorithm is used to learn the mean vector and covariance matrix. Since the learned mean vector and covariance matrix do not have a parametric form, it needs an approximate estimation of the kernel function when making predictions. Since the GP prior is not robust to the outlier tasks, the robust extensions by utilizing the t Process (TP) are proposed in (Yu et al., 2007; Zhang & Yeung, 2010b). Different from the above methods, which are mostly based on GP, Zhang et al. (2005) describe a latent variable model for multi-task learning. For each task T_i, the classifier or regressor is parameterized by the parameters $\boldsymbol{\theta}_i$. Then $\boldsymbol{\theta}_i$'s in different tasks are assumed to satisfy a latent variable model as

$$\begin{aligned}\boldsymbol{\theta}_i &= \boldsymbol{\Lambda}\mathbf{s}_i + \mathbf{e}_i \\ \mathbf{e}_i &\sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Psi}).\end{aligned}$$

In this model, the parameters $\boldsymbol{\Lambda}$ and $\boldsymbol{\Psi}$ are shared by all tasks. By utilizing different priors on $\mathbf{s}_i$, this model is flexible to describe many situations in multi-task learning, such as independent

tasks, noisy tasks, clusters of tasks, tasks having sparse representations, duplicated tasks and evolving tasks.

Task Relationship Learning Approach

In multi-task learning, a central issue is how to characteristic the task relationships between different tasks. Most existing methods address this problem by making an assumption on the task relationships, for example, all tasks are similar or share the same data representation, or utilize some prior knowledge in some specific domains. However, in most cases, the model assumption is not easy to verify. Moreover, in most applications, prior knowledge about task relationships does not exist. In these cases, we hope to learn the task relationships from the data automatically. The task clustering approach can be viewed as one way to learn task relationships. However, the task relationships learned by the task clustering approach are 'local' task relationships since the task clustering approach can cluster the positive correlated tasks into the same task cluster and identity the outlier tasks, but ignore the negative correlated tasks which may exist between the tasks in different task clusters. The multi-task GP model proposed in (Bonilla et al., 2007) is the first method to learn the global task relationships in the form of a task covariance matrix. In the following, we briefly introduce this method.

The multi-task GP model in (Bonilla et al., 2007) directly models a task covariance matrix $\mathbf{\Sigma}$ and incorporates it into the definition of a GP prior as follows:

$$\langle f_j^i, f_s^r \rangle = \Sigma_{ir} k(\mathbf{x}_j^i, \mathbf{x}_s^r),$$

where $\langle \cdot, \cdot \rangle$ denotes the covariance of two random variables, f_j^i is the latent function value for the jth data point $\mathbf{x}_j^i$ in the ith task, Σ_{ir} is the (i, r)th element of $\mathbf{\Sigma}$, and $k(\cdot, \cdot)$ is a kernel function. The output y_j^i given f_j^i is defined as

$$y_j^i | f_j^i \sim \mathcal{N}(f_j^i, \sigma_i^2),$$

which defines the likelihood for $\mathbf{x}_j^i$. Here y_j^i is the label for $\mathbf{x}_j^i$ and σ_i is the noise level of the ith task. For parameter learning, two methods are presented in (Bonilla et al., 2007) to maximize the marginal likelihood with one based on the expectation-maximization (EM) (Dempster et al., 1977) algorithm and another based on a gradient method. The EM algorithm is more suitable for multi-output regression problems since the latent function value of each data point in each task is treated as a hidden variable in EM. On the other hand, the gradient method is more suitable for general multi-task learning problems. For the marginal likelihood, one advantage of the formulation in (Bonilla et al., 2007) is its analytical form. This is similar to the GP model and hence inference can be made efficiently. However, the model does have some drawbacks. One drawback is that when the number of tasks is large, the low-rank approximation of the task covariance matrix used to reduce the computational cost may limit its expressive power. Another

is that since the log-likelihood is non-convex with respect to $\mathbf{\Sigma}$ or its low-rank approximation, the solution found by the parameter learning algorithm may be very sensitive to the initial value of $\mathbf{\Sigma}$ with no guarantee of an optimal solution.

To overcome the drawbacks of multi-task GP and also develop methods to learn the task relationships in other models, we have previously developed some methods, for example, (Zhang & Yeung, 2010a, 2010b, 2010d; Zhang et al., 2010, 2010b). The multi-task relationship learning (MTRL) method in (Zhang & Yeung, 2010a) learns the task relationship also in the form of a task covariance matrix under the regularization framework by utilizing a convex formulation. An application of the MTRL method to transfer metric learning problems is investigated in (Zhang & Yeung, 2010d). Different from the MTRL method which is a non-Bayesian model, the multi-task generalized t process in (Zhang & Yeung, 2010b) learns a nonparametric task covariance matrix in a Bayesian model, for example, the generalized t process. Moreover, in an ongoing project, we study the learning of the high-order task relationships in multi-task learning which is beyond the pairwise task relationships in the existing methods. In the scenario of multi-task feature selection, the probabilistic multi-task feature selection method in (Zhang et al., 2010) discusses how to learn task relationships for multi-task feature selection which is an problem untouched in previous studies. Moreover, the multi-domain collaborative filtering method in (Zhang et al., 2010b) investigates how to learn the relations among different domains for the collaborative filtering problem. More details are discussed in the following chapters and we omit them here.

2.1.2 Theoretical Analysis

The above sections demonstrate the effectiveness of multi-task learning according to the empirical evidence. In the following, we present some existing theoretical analysis results for multi-task learning.

Baxter (1997) analyzes multi-task learning from a Bayesian perspective, in which the learning model is a Bayesian neural network. In (Baxter, 1997), the analysis answers the question 'how much information does each task need in multi-task learning framework' and gets the following result

$$\bar{R}_{k,\pi^\star} = \frac{\dim_{P_\Pi}(\pi^\star)}{2}\frac{\ln k}{k} + H(P_{\Theta|\pi^\star}) + O(\frac{\ln k}{k}) \tag{2.6}$$

where $\pi^\star$ denotes the true prior for the parameters Θ in the Bayesian neural network, $\bar{R}_{k,\pi^\star} = \frac{-E_{\Theta|\pi^\star}[\ln p(\Theta)]}{k}$ is defined as the expected amount of information per task to learn from the k tasks, $H(P_{\Theta|\pi^\star})$ denotes the entropy for the true prior and $\dim_{P_\Pi}(\pi^\star)$ is the local metric dimension of $\pi^\star$. From Eq. (2.6), we can see that when the number of tasks k increases, the required information per task is decreasing, which verifies the effectiveness of multi-task learning.

Different from the analysis in (Baxter, 1997), Baxter (2000) analyzes multi-task learning from the probably approximately correct (PAC) learning theory. Moreover, different from the objective of (Baxter, 1997), Baxter (2000) wants to answer the two questions of 'how many tasks are needed' and 'how many examples each task needs' with the answer in the following.

Theorem 2.1 *Let $\mathbb{H} = \{\mathcal{H}\}$ be any permissible hypothesis space family. We assume that the number of examples in each task is the same, denoted by n. If the number of tasks k satisfies*

$$k \geq \max\{\frac{256}{\varepsilon^2} \ln \frac{8\mathcal{C}(\frac{\varepsilon}{32}, \mathbb{H}^\star)}{\delta}, \frac{64}{\varepsilon^2}\},$$

and the number of examples n of each task satisfies

$$n \geq \max\{\frac{256}{n\varepsilon^2} \ln \frac{8\mathcal{C}(\frac{\varepsilon}{32}, \mathbb{H}_l^k)}{\delta}, \frac{64}{\varepsilon^2}\},$$

then with probability at least $1 - \delta$, all $\mathcal{H} \in \mathbb{H}$ will satisfy

$$er_Q(\mathcal{H}) \leq er_{\mathbf{z}}(\mathcal{H}) + \varepsilon.$$

From Theorem 2.1, we can get that the generalization error for multi-task learning is bounded by the empirical error, which is similar to single-task learning. So if the empirical error is small, then the generalization performance of multi-task learning will be guaranteed. Moreover, the lower bound of n decreases when k increases, since $\ln \mathcal{C}(\frac{\varepsilon}{32}, \mathbb{H}^\star) \in O(n)$. Moreover, some works such as (Ben-David et al., 2002; Ben-David & Schuller, 2003; Maurer, 2006) generalize the analysis to a more general case.

2.1.3 Applications

Multi-task learning has many applications in many areas, that is, computer vision, information retrieval and Bioinformatics. We review some of these applications in the following.

- *Face Recognition*: Heisele et al. (2001) propose a multi-task learning method for face recognition. This method first detects the components of a face and then combines the features of all the components and the whole face for face recognition. Lapedriza et al. (2008) propose a multi-task feature extraction method for face recognition. In this method, face recognition is treated as a target task, while other face tasks such as facial expression recognition as the complementary tasks to help improve the performance of face recognition. This method works by maximizing the mutual information between the low-dimensional representations and the labels in the face recognition task while minimizing the mutual information between the low-dimensional representations and the labels in the complementary tasks.

- *Image Classification*: Quattoni et al. (2008) propose a method for image classification: the unlabeled data in the target task is used to learn the prototype representation, then the prototypes are selected by some classifiers learned in the source tasks, and finally the selected prototypes are used for the target task. Ahmed et al. (2008) propose a method for visual recognition via a multi-task neural network, in which the target task and the pseudo tasks as the source tasks share a common representation in the common hidden layer. This method also discusses how to generate the pseudo tasks for visual recognition tasks. Kienzle and Chellapilla (2006) propose a biased regularization method for personalized handwriting recognition, in which the parameters of the SVM model in the source tasks are provided as a bias for those in the target task.

- *Object Detection*: Torralba et al. (2004) propose a method for multi-class object detection. Different from the previous methods in object detection which train a classifier for an individual object, this method formulates it as a multi-class classification problem where all the objects share some common features, with a by-product to reduce the number of features required in object detection.

- *Image Segmentation*: An et al. (2008) utilize a Dirichlet process and a kernel stick-breaking process to segment multiple images simultaneously. The Dirichlet process is used as a prior for the base measure and the kernel stick-breaking process is used to incorporate the spatial information to help the segmentation. This work can be viewed as a way for multi-task clustering.

- *Age Estimation*: In (Zhang & Yeung, 2010c), Zhang and Yeung formulate the age estimation problem as a multi-task regression problem, where each task corresponds to estimating the ages for one person, and propose a multi-task extension of warped GP to solve this problem.

- *Compressive Sensing*: Qi et al. (2008) propose a multi-task learning method for compressive sensing, whose model is similar to that in (Xue et al., 2007).

- *Collaborative Filtering*: Yu et al. (2004) unify the content-based filtering and collaborative filtering (CF) in one framework by using a task clustering method, in which the parameters for each user profile share a common DP prior. Yu and Tresp (2005) propose a multi-task learning method to solve the CF problem which utilizes the low-rank matrix approximation. The methods in (Cao et al., 2010; Zhang et al., 2010b) utilize the useful information in multiple domains to improve the performance on each domain by learning the domain relations in the form of a covariance matrix.

- *Text Classification*: Raina et al. (2006) propose a transfer learning method for the binary text classification problem. This method places a Gaussian prior on the model parameters

of the logistic regression model for target task and learns the covariance matrix of the Gaussian prior from the source tasks. Do and Ng (2006) also propose a method based on the logistic regression model for multi-class text classification problem.

- *Bioinformatics*: Bickel et al. (2008) propose a multi-task learning method based on the distribution matching for the HIV therapy screening problem. Xu et al. (2010) use the multi-task SVM (Evgeniou & Pontil, 2004) and the multi-task feature learning method (Argyriou et al., 2006, 2008a) to solve the protein subcellular location prediction problem. Liu et al. (2010) use the multi-task feature learning method (Argyriou et al., 2006, 2008a) for the cross-platform siRNA efficacy prediction problem. Zhang et al. (2010a) identifies the common mechanisms of the responses to the therapeutic targets by proposing a sparse multi-task regression method. Puniyani et al. (2010) utilize the multi-task feature selection method on the multi-population GWA mapping problems. Lee et al. (2010) extend the multi-task feature selection method by learning the hyperparameters for the eQTL detection problem.

- *Finance*: Ghosn and Bengio (1996) apply a multi-task learning method to the stock selection problem. Different from the previous methods, which use one neural network to predict the return of one stock, the method in (Ghosn & Bengio, 1996) learns from several stocks in one neural network, in which the hidden layer is shared by all stocks and can be viewed as a common representation.

- *Robot Inverse Dynamics*: The methods in (Chai et al., 2008; Yeung & Zhang, 2009) utilize the multi-task GP model in (Bonilla et al., 2007) for the robot inverse dynamics problem to improve the performance over the existing methods.

2.2 Matrix Variate Distributions

The single-task learning methods usually use the vector variate distributions, that is, normal distribution, as the priors on the model parameters in one single task. If we want to jointly model the distribution of the model parameters of all tasks in the form of a matrix, we need to resort to matrix variate distributions (Gupta & Nagar, 2000). Here, we briefly review some matrix variate distributions on which our proposed methods are based. More detailed information can be found in (Gupta & Nagar, 2000).

Definition 2.1 *A random matrix $\mathbf{X} \in \mathbb{R}^{m \times n}$ is said to follow matrix variate normal distribution with mean matrix $\mathbf{M} \in \mathbb{R}^{m \times n}$, row covariance matrix $\mathbf{\Sigma} \in \mathbb{R}^{m \times m}$ and column covariance matrix $\mathbf{\Omega} \in \mathbb{R}^{n \times n}$ where $\mathbf{\Omega} \succ \mathbf{0}$ and $\mathbf{\Sigma} \succ \mathbf{0}$, written as $\mathbf{X} \sim \mathcal{MN}_{m \times n}(\mathbf{M}, \mathbf{\Sigma} \otimes \mathbf{\Omega})$, iff its*

probability density function (p.d.f.) is given by

$$\frac{\exp\Big(-\frac{1}{2}\mathrm{tr}\big(\mathbf{\Sigma}^{-1}(\mathbf{X}-\mathbf{M})\mathbf{\Omega}^{-1}(\mathbf{X}-\mathbf{M})^T\big)\Big)}{(2\pi)^{md/2}|\mathbf{\Omega}|^{m/2}|\mathbf{\Sigma}|^{n/2}}.$$

Definition 2.2 *A random matrix $\mathbf{X} \in \mathbb{R}^{m\times n}$ is said to follow matrix variate t distribution with the number of degrees of freedom ν, mean matrix $\mathbf{M} \in \mathbb{R}^{m\times n}$, row covariance matrix $\mathbf{\Sigma} \in \mathbb{R}^{m\times m}$ and column covariance matrix $\mathbf{\Omega} \in \mathbb{R}^{n\times n}$ where $\mathbf{\Omega} \succ \mathbf{0}$ and $\mathbf{\Sigma} \succ \mathbf{0}$, written as $\mathbf{X} \sim \mathcal{MT}_{m\times n}(\mathbf{M}, \mathbf{\Sigma}\otimes\mathbf{\Omega})$, if its probability density function (p.d.f.) is given by*

$$\frac{\Gamma_m(\nu'/2)\,|\mathbf{I}_m+\mathbf{\Sigma}^{-1}(\mathbf{X}-\mathbf{M})\mathbf{\Omega}^{-1}(\mathbf{X}-\mathbf{M})^T|^{-\nu'/2}}{\pi^{mn/2}\Gamma_m((\nu+m-1)/2)|\mathbf{\Sigma}|^{n/2}|\mathbf{\Omega}|^{m/2}},$$

where $\nu' = \nu+m+n-1$.

Definition 2.3 *A random matrix $\mathbf{X}\in\mathbb{R}^{m\times n}$ is said to follow matrix variate generalized normal random variable, written as $\mathbf{X}\sim\mathcal{MGN}_{m\times n}(\mathbf{M},\mathbf{\Sigma},\mathbf{\Omega},q)$, iff its p.d.f. is given as follows:*

$$\frac{1}{\big[2\Gamma(1+\frac{1}{q})\big]^{mn}|\mathbf{\Sigma}|^n|\mathbf{\Omega}|^m}\exp\left[-\sum_{i=1}^{m}\sum_{j=1}^{n}\left|\sum_{k=1}^{m}\sum_{l=1}^{n}(\Sigma^{-1})_{ik}(X_{kl}-M_{kl})(\Omega^{-1})_{lj}\right|^q\right],$$

where $\mathbf{\Sigma}\in\mathbb{R}^{s\times s}$ and $\mathbf{\Omega}\in\mathbb{R}^{t\times t}$ are nonsingular, A_{ij} is the (i,j)th element of matrix $\mathbf{A}$ and $(A^{-1})_{ij}$ is the (i,j)th element of the matrix inverse $\mathbf{A}^{-1}$.

When the parameter q in a matrix variate generalized normal random variable is set to 2, the matrix variate generalized normal distribution will degenerate to matrix variate normal distribution which is one of the reasons that it is called matrix variate *generalized* normal distribution.

Definition 2.4 *A $n\times n$ random symmetric positive definite matrix $\mathbf{K}$ is said to have a Wishart distribution with freedom parameter q and $n\times n$ scale matrix $\mathbf{\Sigma}\succ\mathbf{0}$, written as $\mathbf{K}\sim\mathcal{W}_n(q,\mathbf{\Sigma})$, iff its p.d.f. is given by*

$$\frac{|\mathbf{K}|^{(q-n-1)/2}}{(2)^{qn/2}\Gamma_n(q/2)|\mathbf{\Sigma}|^{q/2}}\exp\Big(-\frac{1}{2}\mathrm{tr}(\mathbf{\Sigma}^{-1}\mathbf{K})\Big),\ q\le n.$$

CHAPTER 3

MULTI-TASK RELATIONSHIP LEARNING

3.1 Introduction

Multi-layered feedforward neural networks and multi-task feature learning assume that all tasks share the same representation without actually learning the task relationships from data automatically. Moreover, they do not consider a negative task correlation and are not robust against outlier tasks. The regularization methods in (Evgeniou & Pontil, 2004; Evgeniou et al., 2005; Kato et al., 2007) assume that the task relationships are given and then utilize this prior knowledge to learn the model parameters. However, they only utilize positive task correlation and task unrelatedness and not negative task correlation. The task clustering methods in (Thrun & O'Sullivan, 1996; Bakker & Heskes, 2003; Xue et al., 2007; Jacob et al., 2008) may be viewed as a way to learn the task relationships from data. Similar tasks are grouped into the same task cluster and outlier tasks are grouped separately, making these methods more robust against outlier tasks. However, they are *local* methods in the sense that only similar tasks within the same task cluster can interact to help each other, thus ignoring negative task correlation which may exist between tasks residing in different clusters. On the other hand, a powerful multi-task learning method in (Bonilla et al., 2007) based on GP provides a *global* approach to model and learn task relationships in the form of a task covariance matrix. A task covariance matrix can model all three types of task relationships: positive task correlation, negative task correlation, and task unrelatedness. However, although this method provides a powerful way to model task relationships, learning of the task covariance matrix gives rise to a non-convex optimization problem which is sensitive to parameter initialization. When the number of tasks is large, the authors propose to use low-rank approximation (Bonilla et al., 2007) which will then weaken the expressive power of the task covariance matrix. Moreover, since the method is based on GP, scaling it to large data sets poses a serious computational challenge.

Our goal in this chapter is to inherit the advantages of (Bonilla et al., 2007) while overcoming its disadvantages. Specifically, we propose a probabilistic approach, called *multi-task relationship learning* (MTRL), which also models the relationships between tasks in a non-parametric manner as a task covariance matrix. By utilizing a matrix variate normal distribution (Gupta & Nagar, 2000) as a prior on the model parameters of all tasks, we obtain a convex objective function which allows us to learn the model parameters and the task relationships simultaneously under the regularization framework. This model can be viewed as a generalization of the regularization framework for single-task learning to the multi-task setting. For efficiency,

we use an alternating optimization method in which each subproblem is a convex problem. We study MTRL in a symmetric multi-task learning setting and then generalize it to the asymmetric setting as well. We discuss some variants of the probabilistic approach to demonstrate the use of other priors for learning task relationships. Moreover, to gain more insight into our model, we also study the relationships between MTRL and some existing multi-task learning methods, such as (Evgeniou & Pontil, 2004; Evgeniou et al., 2005; Kato et al., 2007; Jacob et al., 2008; Bonilla et al., 2007), showing that the regularized methods in (Evgeniou & Pontil, 2004; Evgeniou et al., 2005; Kato et al., 2007; Jacob et al., 2008) can be viewed as special cases of MTRL and the multi-task GP model in (Bonilla et al., 2007) is related to our model. Moreover, we also apply this idea to transfer metric learning where the objective is to learn an accurate distance metric in the target task.

3.2 Multi-Task Relationship Learning

Suppose we are given m learning tasks $\{T_i\}_{i=1}^m$. For the ith task T_i, the training set $\mathcal{D}_i$ consists of n_i data points $(\mathbf{x}_j^i, y_j^i)$, $j = 1, \ldots, n_i$, with $\mathbf{x}_j^i \in \mathbb{R}^d$ and its corresponding output $y_j^i \in \mathbb{R}$ if it is a regression problem and $y_j^i \in \{-1, 1\}$ if it is a binary classification problem. The linear function for T_i is defined as $f_i(\mathbf{x}) = \mathbf{w}_i^T \mathbf{x} + b_i$.

3.2.1 Probabilistic Framework

The likelihood for y_j^i given $\mathbf{x}_j^i$, $\mathbf{w}_i$, b_i and ε_i is:

$$y_j^i \mid \mathbf{x}_j^i, \mathbf{w}_i, b_i, \varepsilon_i \sim \mathcal{N}(\mathbf{w}_i^T \mathbf{x}_j^i + b_i, \varepsilon_i^2). \tag{3.1}$$

The prior on $\mathbf{W} = (\mathbf{w}_1, \ldots, \mathbf{w}_m)$ is defined as

$$\mathbf{W} \mid \epsilon_i \sim \left(\prod_{i=1}^m \mathcal{N}(\mathbf{w}_i \mid \mathbf{0}_d, \epsilon_i^2 \mathbf{I}_d) \right) q(\mathbf{W}). \tag{3.2}$$

The first term of the prior on $\mathbf{W}$ is to penalize the complexity of each column of $\mathbf{W}$ separately and the second term is to model the structure of $\mathbf{W}$. Since $\mathbf{W}$ is a matrix variable, it is natural to use a matrix variate distribution (Gupta & Nagar, 2000) to model it. Here we use the matrix variate normal distribution for $q(\mathbf{W})$. More specifically,

$$q(\mathbf{W}) = \mathcal{MN}_{d \times m}(\mathbf{W} \mid \mathbf{0}_{d \times m}, \mathbf{I}_d \otimes \mathbf{\Omega}) \tag{3.3}$$

where $\mathbf{0}_{d \times m}$ is the $d \times m$ zero matrix. For the prior in Eq. (3.3), the row covariance matrix $\mathbf{I}_d$ models the relationships between features and the column covariance matrix $\mathbf{\Omega}$ models the relationships between different $\mathbf{w}_i$'s. In other words, $\mathbf{\Omega}$ models the relationships between tasks.

When there is only one task and $\mathbf{\Omega}$ is given as a positive scalar value, our model will degenerate to the probabilistic model for regularized least-squares regression and least-squares SVM (Gestel et al., 2004). So our probabilistic model can be viewed as a generalization of the probabilistic model for single-task learning. However, unlike single-task learning, $\mathbf{\Omega}$ cannot be given *a priori* for most multi-task learning applications and so we seek to estimate it from data automatically.

It is easy to see that this model is a concrete case of our proposed framework in section 1.3 with the prior defined in Eq. (3.2) and the likelihood in Eq. (3.1).

It follows that the posterior distribution for $\mathbf{W}$, which is proportional to the product of the prior and the likelihood function (Bishop, 2006), is given by:

$$p(\mathbf{W} \mid \mathbf{X}, \mathbf{y}, \mathbf{b}, \boldsymbol{\varepsilon}, \boldsymbol{\epsilon}, \mathbf{\Omega}) \propto p(\mathbf{y} \mid \mathbf{X}, \mathbf{W}, \mathbf{b}, \boldsymbol{\varepsilon})\, p(\mathbf{W} \mid \boldsymbol{\epsilon}, \mathbf{\Omega}), \tag{3.4}$$

where $\mathbf{y} = (y_1^1, \ldots, y_{n_1}^1, \ldots, y_1^m, \ldots, y_{n_m}^m)^T$, $\mathbf{X}$ denotes the data matrix of all data points in all tasks, and $\mathbf{b} = (b_1, \ldots, b_m)^T$. Taking the negative logarithm of Eq. (3.4) and combining with Eqs. (3.1)–(3.3), we obtain the *maximum a posteriori* (MAP) estimation of $\mathbf{W}$ and the MLE of $\mathbf{b}$ and $\mathbf{\Omega}$ by solving the following problem:

$$\min_{\mathbf{W},\mathbf{b},\mathbf{\Omega}\succeq 0} \sum_{i=1}^{m} \frac{1}{\varepsilon_i^2} \sum_{j=1}^{n_i} (y_j^i - \mathbf{w}_i^T \mathbf{x}_j^i - b_i)^2 + \sum_{i=1}^{m} \frac{1}{\epsilon_i^2} \mathbf{w}_i^T \mathbf{w}_i + \operatorname{tr}(\mathbf{W}\mathbf{\Omega}^{-1}\mathbf{W}^T) + d \ln |\mathbf{\Omega}|. \tag{3.5}$$

Here the PSD constraint on $\mathbf{\Omega}$ holds due to the fact that $\mathbf{\Omega}$ is defined as a task covariance matrix. For simplicity of discussion, we assume that $\varepsilon = \varepsilon_i$ and $\epsilon = \epsilon_i, \forall i = 1, \ldots, m$. The effect of the last term in problem (3.5) is to penalize the complexity of $\mathbf{\Omega}$. However, as we will see later, the first three terms in problem (3.5) are jointly convex with respect to all variables but the last term is concave since $-\ln |\mathbf{\Omega}|$ is a convex function with respect to $\mathbf{\Omega}$, according to (Boyd & Vandenberghe, 2004). Moreover, for kernel extension, we have no idea about d which may even be infinite after feature mapping, making problem (3.5) difficult to optimize. In the following, we first present a useful lemma which will be used later.

Lemma 3.1 *For any $m \times m$ PSD matrix $\mathbf{\Omega}$, $\ln |\mathbf{\Omega}| \leq \operatorname{tr}(\mathbf{\Omega}) - m$.*

Proof:
We denote the eigenvalues of $\mathbf{\Omega}$ by $e_1, \ldots, e_m$. Then $\ln |\mathbf{\Omega}| = \sum_{i=1}^m \ln e_i$ and $\operatorname{tr}(\mathbf{\Omega}) = \sum_{i=1}^m e_i$. Due to the concavity of the logarithm function, we can obtain

$$\ln x \leq \ln 1 + \frac{1}{1}(x - 1) = x - 1$$

by applying the first-order condition. Then

$$\ln |\mathbf{\Omega}| = \sum_{i=1}^{m} \ln e_i \leq \sum_{i=1}^{m} e_i - m = \operatorname{tr}(\mathbf{\Omega}) - m.$$

This proves the lemma. □

Based on Lemma 3.1, we can relax the optimization problem (3.5) as

$$\min_{\mathbf{W},\mathbf{b},\mathbf{\Omega}\succeq 0} \sum_{i=1}^{m} \frac{1}{\varepsilon^2} \sum_{j=1}^{n_i} (y_j^i - \mathbf{w}_i^T \mathbf{x}_j^i - b_i)^2 + \sum_{i=1}^{m} \frac{1}{\epsilon^2} \mathbf{w}_i^T \mathbf{w}_i + \mathrm{tr}(\mathbf{W}\mathbf{\Omega}^{-1}\mathbf{W}^T) + d\,\mathrm{tr}(\mathbf{\Omega}). \quad (3.6)$$

However, the last term in problem (3.6) is still related to the data dimensionality d which usually cannot be estimated accurately in kernel methods. So we incorporate the last term into the constraint, leading to the following problem

$$\begin{aligned} \min_{\mathbf{W},\mathbf{b},\mathbf{\Omega}} \quad & \sum_{i=1}^{m} \sum_{j=1}^{n_i} (y_j^i - \mathbf{w}_i^T \mathbf{x}_j^i - b_i)^2 + \frac{\lambda_1}{2} \mathrm{tr}(\mathbf{W}\mathbf{W}^T) + \frac{\lambda_2}{2} \mathrm{tr}(\mathbf{W}\mathbf{\Omega}^{-1}\mathbf{W}^T) \\ \text{s.t.} \quad & \mathbf{\Omega} \succeq 0 \\ & \mathrm{tr}(\mathbf{\Omega}) \leq c, \end{aligned} \quad (3.7)$$

where $\lambda_1 = \frac{2\varepsilon^2}{\epsilon^2}$ and $\lambda_2 = 2\varepsilon^2$ are regularization parameters. By using the method of Lagrange multipliers, it is easy to show that problems (3.6) and (3.7) are equivalent. Here we simply set $c = 1$.

The first term in (3.7) measures the empirical loss on the training data, the second term penalizes the complexity of $\mathbf{W}$, and the third term measures the relationships between all tasks based on $\mathbf{W}$ and $\mathbf{\Omega}$.

To avoid the task imbalance problem in which one task has so many data points that it dominates the empirical loss, we modify problem (3.7) as

$$\begin{aligned} \min_{\mathbf{W},\mathbf{b},\mathbf{\Omega}} \quad & \sum_{i=1}^{m} \frac{1}{n_i} \sum_{j=1}^{n_i} (y_j^i - \mathbf{w}_i^T \mathbf{x}_j^i - b_i)^2 + \frac{\lambda_1}{2} \mathrm{tr}(\mathbf{W}\mathbf{W}^T) + \frac{\lambda_2}{2} \mathrm{tr}(\mathbf{W}\mathbf{\Omega}^{-1}\mathbf{W}^T) \\ \text{s.t.} \quad & \mathbf{\Omega} \succeq 0 \\ & \mathrm{tr}(\mathbf{\Omega}) \leq 1. \end{aligned} \quad (3.8)$$

Note that (3.8) is a semi-definite programming (SDP) problem which is computationally demanding. In what follows, we will present an efficient algorithm for solving it.

3.2.2 Optimization Procedure

We first prove the convexity of problem (3.8) with respect to all variables.

Theorem 3.1 *Problem (3.8) is convex with respect to* $\mathbf{W}$, b *and* $\mathbf{\Omega}$.

Proof:

It is easy to see that the first two terms in the objective function of problem (3.8) are convex

with respect to all variables and the constraints in (3.8) are also convex. We rewrite the third term in the objective function as

$$\mathrm{tr}(\mathbf{W}\mathbf{\Omega}^{-1}\mathbf{W}^T) = \sum_t \mathbf{W}(t,:)\mathbf{\Omega}^{-1}\mathbf{W}(t,:)^T,$$

where $\mathbf{W}(t,:)$ denotes the tth row of $\mathbf{W}$. $\mathbf{W}(t,:)\mathbf{\Omega}^{-1}\mathbf{W}(t,:)^T$ is called a matrix fractional function in Example 3.4 (page 76) of (Boyd & Vandenberghe, 2004) and it is proved to be a convex function with respect to $\mathbf{W}(t,:)$ and $\mathbf{\Omega}$ there when $\mathbf{\Omega}$ is a positive semidefinite matrix (which is satisfied by the first constraint of (3.8)). Since $\mathbf{W}(t,:)$ is a row of $\mathbf{W}$, $\mathbf{W}(t,:)\mathbf{\Omega}^{-1}\mathbf{W}(t,:)^T$ is also convex with respect to $\mathbf{W}$ and $\mathbf{\Omega}$. Because the summation operation can preserve convexity according to the analysis on page 79 of (Boyd & Vandenberghe, 2004), $\mathrm{tr}(\mathbf{W}\mathbf{\Omega}^{-1}\mathbf{W}^T) = \sum_t \mathbf{W}(t,:)\mathbf{\Omega}^{-1}\mathbf{W}(t,:)^T$ is convex with respect to $\mathbf{W}$, $\mathbf{b}$ and $\mathbf{\Omega}$. So the objective function and the constraints in problem (3.8) are convex with respect to all variables and hence problem (3.8) is jointly convex. □

Even though the optimization problem (3.8) is convex with respect to $\mathbf{W}$, $\mathbf{b}$ and $\mathbf{\Omega}$ jointly, it is not easy to optimize the objective function with respect to all the variables simultaneously. Here we propose an alternating method to solve the problem more efficiently. Specifically, we first optimize the objective function with respect to $\mathbf{W}$ and $\mathbf{b}$ when $\mathbf{\Omega}$ is fixed, and then optimize it with respect to $\mathbf{\Omega}$ when $\mathbf{W}$ and $\mathbf{b}$ are fixed. This procedure is repeated until convergence. In what follows, we will present the two subproblems separately.

Optimizing w.r.t. $\mathbf{W}$ and $\mathbf{b}$ when $\mathbf{\Omega}$ is fixed

When $\mathbf{\Omega}$ is given and fixed, the optimization problem for finding $\mathbf{W}$ and $\mathbf{b}$ is an unconstrained convex optimization problem. The optimization problem can be stated as:

$$\min_{\mathbf{W},\mathbf{b}} \sum_{i=1}^{m} \frac{1}{n_i} \sum_{j=1}^{n_i} (y_j^i - \mathbf{w}_i^T \mathbf{x}_j^i - b_i)^2 + \frac{\lambda_1}{2}\mathrm{tr}(\mathbf{W}\mathbf{W}^T) + \frac{\lambda_2}{2}\mathrm{tr}(\mathbf{W}\mathbf{\Omega}^{-1}\mathbf{W}^T). \tag{3.9}$$

To facilitate a kernel extension to be given later for the general nonlinear case, we reformulate the optimization problem into a dual form by first expressing problem (3.9) as a constrained optimization problem:

$$\begin{aligned} \min_{\mathbf{W},\mathbf{b},\{\varepsilon_j^i\}} \quad & \sum_{i=1}^{m} \frac{1}{n_i} \sum_{j=1}^{n_i} (\varepsilon_j^i)^2 + \frac{\lambda_1}{2}\mathrm{tr}(\mathbf{W}\mathbf{W}^T) + \frac{\lambda_2}{2}\mathrm{tr}(\mathbf{W}\mathbf{\Omega}^{-1}\mathbf{W}^T) \\ \text{s.t.} \quad & y_j^i - (\mathbf{w}_i^T \mathbf{x}_j^i + b_i) = \varepsilon_j^i \quad \forall i, j. \end{aligned} \tag{3.10}$$

The Lagrangian of problem (3.10) is given by

$$G = \sum_{i=1}^{m} \frac{1}{n_i} \sum_{j=1}^{n_i} (\varepsilon_j^i)^2 + \frac{\lambda_1}{2} \mathrm{tr}(\mathbf{W}\mathbf{W}^T) + \frac{\lambda_2}{2} \mathrm{tr}(\mathbf{W}\boldsymbol{\Omega}^{-1}\mathbf{W}^T)$$
$$+ \sum_{i=1}^{m} \sum_{j=1}^{n_i} \alpha_j^i \left[y_j^i - (\mathbf{w}_i^T \mathbf{x}_j^i + b_i) - \varepsilon_j^i \right]. \tag{3.11}$$

We calculate the gradients of G with respect to $\mathbf{W}$, b_i and ε_j^i and set them to 0 to obtain

$$\frac{\partial G}{\partial \mathbf{W}} = \mathbf{W}(\lambda_1 \mathbf{I}_m + \lambda_2 \boldsymbol{\Omega}^{-1}) - \sum_{i=1}^{m} \sum_{j=1}^{n_i} \alpha_j^i \mathbf{x}_j^i \mathbf{e}_i^T = 0 \Rightarrow \mathbf{W} = \sum_{i=1}^{m} \sum_{j=1}^{n_i} \alpha_j^i \mathbf{x}_j^i \mathbf{e}_i^T \boldsymbol{\Omega}(\lambda_1 \boldsymbol{\Omega} + \lambda_2 \mathbf{I}_m)^{-1}$$

$$\frac{\partial G}{\partial b_i} = -\sum_{j=1}^{n_i} \alpha_j^i = 0$$

$$\frac{\partial G}{\partial \varepsilon_j^i} = \frac{2}{n_i} \varepsilon_j^i - \alpha_j^i = 0,$$

where $\mathbf{e}_i$ is the ith column vector of $\mathbf{I}_m$. Combining the above equations, we obtain the following linear system:

$$\begin{pmatrix} \mathbf{K} + \frac{1}{2}\boldsymbol{\Lambda} & \mathbf{M}_{12} \\ \mathbf{M}_{21} & \mathbf{0}_{m \times m} \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha} \\ \mathbf{b} \end{pmatrix} = \begin{pmatrix} \mathbf{y} \\ \mathbf{0}_{m \times 1} \end{pmatrix}, \tag{3.12}$$

where $k_{MT}(\mathbf{x}_{j_1}^{i_1}, \mathbf{x}_{j_2}^{i_2}) = \mathbf{e}_{i_1}^T \boldsymbol{\Omega}(\lambda_1 \boldsymbol{\Omega} + \lambda_2 \mathbf{I}_m)^{-1} \mathbf{e}_{i_2} (\mathbf{x}_{j_1}^{i_1})^T \mathbf{x}_{j_2}^{i_2}$ is the linear multi-task kernel, $\mathbf{K}$ is the kernel matrix defined on all data points for all tasks using the linear multi-task kernel, $\boldsymbol{\alpha} = (\alpha_1^1, \ldots, \alpha_{n_m}^m)^T$, $\mathbf{0}_{p \times q}$ is the $p \times q$ zero matrix or vector, $\boldsymbol{\Lambda}$ is a diagonal matrix whose diagonal element is equal to n_i if the corresponding data point belongs to the ith task, $N_i = \sum_{j=1}^{i} n_j$, and $\mathbf{M}_{12} = \mathbf{M}_{21}^T = (\mathbf{e}_{N_0+1}^{N_1}, \mathbf{e}_{N_1+1}^{N_2}, \ldots, \mathbf{e}_{N_{m-1}+1}^{N_m})$ where $\mathbf{e}_q^p$ is a zero vector with only the elements whose indices are in $[q, p]$ being equal to 1.

When the total number of data points for all tasks is very large, the computational cost required to solve the linear system (3.12) directly will be very high. In this situation, we can use another optimization method to solve it. It is easy to show that the dual form of problem (3.10) can be formulated as:

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & h(\boldsymbol{\alpha}) = \frac{1}{2} \boldsymbol{\alpha}^T \tilde{\mathbf{K}} \boldsymbol{\alpha} - \sum_{i,j} \alpha_j^i y_j^i \\ \text{s.t.} \quad & \sum_j \alpha_j^i = 0 \quad \forall i, \end{aligned} \tag{3.13}$$

where $\tilde{\mathbf{K}} = \mathbf{K} + \frac{1}{2}\boldsymbol{\Lambda}$. Note that it is similar to the dual form of least-squares SVM (Gestel et al., 2004) except that there is only one constraint in least-squares SVM but here there are m constraints with each constraint corresponding to one task. Here we use an SMO algorithm

similar to that for least-squares SVM (Keerthi & Shevade, 2003). The detailed SMO algorithm is given in Appendix A.1.

Optimizing w.r.t. $\mathbf{\Omega}$ when $\mathbf{W}$ and b are fixed

When $\mathbf{W}$ and $\mathbf{b}$ are fixed, the optimization problem for finding $\mathbf{\Omega}$ becomes

$$\begin{aligned} \min_{\mathbf{\Omega}} \quad & \mathrm{tr}(\mathbf{\Omega}^{-1}\mathbf{W}^T\mathbf{W}) \\ \text{s.t.} \quad & \mathbf{\Omega} \succeq 0 \\ & \mathrm{tr}(\mathbf{\Omega}) \leq 1. \end{aligned} \tag{3.14}$$

Then we have

$$\begin{aligned} \mathrm{tr}(\mathbf{\Omega}^{-1}\mathbf{A}) &\geq \mathrm{tr}(\mathbf{\Omega}^{-1}\mathbf{A})\mathrm{tr}(\mathbf{\Omega}) \\ &= \mathrm{tr}((\mathbf{\Omega}^{-\frac{1}{2}}\mathbf{A}^{\frac{1}{2}})(\mathbf{A}^{\frac{1}{2}}\mathbf{\Omega}^{-\frac{1}{2}}))\mathrm{tr}(\mathbf{\Omega}^{\frac{1}{2}}\mathbf{\Omega}^{\frac{1}{2}}) \\ &\geq (\mathrm{tr}(\mathbf{\Omega}^{-\frac{1}{2}}\mathbf{A}^{\frac{1}{2}}\mathbf{\Omega}^{\frac{1}{2}}))^2 = (\mathrm{tr}(\mathbf{A}^{\frac{1}{2}}))^2, \end{aligned}$$

where $\mathbf{A} = \mathbf{W}^T\mathbf{W}$. The first inequality holds because of the last constraint in problem (3.14), and the last inequality holds because of the Cauchy-Schwarz inequality for the Frobenius norm. Moreover, $\mathrm{tr}(\mathbf{\Omega}^{-1}\mathbf{A})$ attains its minimum value $(\mathrm{tr}(\mathbf{A}^{\frac{1}{2}}))^2$ if and only if $\mathbf{\Omega}^{-\frac{1}{2}}\mathbf{A}^{\frac{1}{2}} = a\mathbf{\Omega}^{\frac{1}{2}}$ for some constant a and $\mathrm{tr}(\mathbf{\Omega}) = 1$. So we can get the analytical solution $\mathbf{\Omega} = \frac{(\mathbf{W}^T\mathbf{W})^{\frac{1}{2}}}{\mathrm{tr}((\mathbf{W}^T\mathbf{W})^{\frac{1}{2}})}$.

We set the initial value of $\mathbf{\Omega}$ to $\frac{1}{m}\mathbf{I}_m$ which corresponds to the assumption that all tasks are unrelated initially.

After learning the optimal values of $\mathbf{W}$, $\mathbf{b}$ and $\mathbf{\Omega}$, we can make prediction for a new data point. Given a test data point $\mathbf{x}_\star^i$ for task T_i, the predictive output $y_\star^i$ is given by $y_\star^i = \sum_{p=1}^{m}\sum_{q=1}^{n_p}\alpha_q^p k_{MT}(\mathbf{x}_q^p, \mathbf{x}_\star^i) + b_i$.

3.2.3 Incorporation of New Tasks

The method described above can only learn from multiple tasks simultaneously which is the setting for symmetric multi-task learning. In asymmetric multi-task learning, when a new task arrives, we could add the data for this new task to the training set and then train a new model from scratch for the $m+1$ tasks using the above method. However, it is undesirable to incorporate new tasks this way due to the high computational cost incurred. Here we introduce an algorithm for asymmetric multi-task learning which is more efficient.

For notational simplicity, let $\tilde{m}$ denote $m+1$. We denote the new task by $T_{\tilde{m}}$ and its training set by $\mathcal{D}_{\tilde{m}} = \{(\mathbf{x}_j^{\tilde{m}}, y_j^{\tilde{m}})\}_{j=1}^{n_{\tilde{m}}}$. The task covariances between $T_{\tilde{m}}$ and the m existing tasks are

represented by the vector $\boldsymbol{\omega}_{\tilde{m}} = (\omega_{\tilde{m},1}, \ldots, \omega_{\tilde{m},m})^T$ and the task variance for $T_{\tilde{m}}$ is defined as σ. Thus the augmented task covariance matrix for the $m+1$ tasks is:

$$\tilde{\boldsymbol{\Omega}} = \begin{pmatrix} (1-\sigma)\boldsymbol{\Omega} & \boldsymbol{\omega}_{\tilde{m}} \\ \boldsymbol{\omega}_{\tilde{m}}^T & \sigma \end{pmatrix},$$

where $\boldsymbol{\Omega}$ is scaled by $(1-\sigma)$ to make $\tilde{\boldsymbol{\Omega}}$ satisfy the constraint $\mathrm{tr}(\tilde{\boldsymbol{\Omega}}) = 1$.[1] The linear function for task T_{m+1} is defined as $f_{m+1}(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$.

With $\mathbf{W}_m = (\mathbf{w}_1, \ldots, \mathbf{w}_m)$ and $\boldsymbol{\Omega}$ at hand, the optimization problem can be formulated as follows:

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\omega}_{\tilde{m}},\sigma} \quad & \frac{1}{n_{\tilde{m}}}\sum_{j=1}^{n_{\tilde{m}}}(y_j^{\tilde{m}} - \mathbf{w}^T\mathbf{x}_j^{\tilde{m}} - b)^2 + \frac{\lambda_1}{2}\|\mathbf{w}\|_2^2 + \frac{\lambda_2}{2}\mathrm{tr}(\mathbf{W}_{\tilde{m}}\tilde{\boldsymbol{\Omega}}^{-1}\mathbf{W}_{\tilde{m}}^T) \\ \text{s.t.} \quad & \tilde{\boldsymbol{\Omega}} \succeq 0, \end{aligned} \tag{3.15}$$

where $\mathbf{W}_{\tilde{m}} = (\mathbf{W}_m, \mathbf{w})$. Problem (3.15) is an SDP problem. Here we assume $\boldsymbol{\Omega}$ is positive definite.[2] So if the constraint in (3.15) holds, then according to the Schur complement (Boyd & Vandenberghe, 2004), this constraint is equivalent to $\boldsymbol{\omega}_{\tilde{m}}^T\boldsymbol{\Omega}^{-1}\boldsymbol{\omega}_{\tilde{m}} \leq \sigma - \sigma^2$. Thus problem (3.15) becomes

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\omega}_{\tilde{m}},\sigma} \quad & \frac{1}{n_{\tilde{m}}}\sum_{j=1}^{n_{\tilde{m}}}(y_j^{\tilde{m}} - \mathbf{w}^T\mathbf{x}_j^{\tilde{m}} - b)^2 + \frac{\lambda_1}{2}\|\mathbf{w}\|_2^2 + \frac{\lambda_2}{2}\mathrm{tr}(\mathbf{W}_{\tilde{m}}\tilde{\boldsymbol{\Omega}}^{-1}\mathbf{W}_{\tilde{m}}^T) \\ \text{s.t.} \quad & \boldsymbol{\omega}_{\tilde{m}}^T\boldsymbol{\Omega}^{-1}\boldsymbol{\omega}_{\tilde{m}} \leq \sigma - \sigma^2. \end{aligned} \tag{3.16}$$

This is a convex problem and thus we can also use an alternating method to solve it.

When using the alternating method to optimize with respect to $\mathbf{w}$ and b, from the block matrix inversion formula, we can get

$$\tilde{\boldsymbol{\Omega}}^{-1} = \begin{pmatrix} \left((1-\sigma)\boldsymbol{\Omega} - \frac{1}{\sigma}\boldsymbol{\omega}_{\tilde{m}}\boldsymbol{\omega}_{\tilde{m}}^T\right)^{-1} & -\frac{1}{(1-\sigma)\sigma'}\boldsymbol{\Omega}^{-1}\boldsymbol{\omega}_{\tilde{m}} \\ -\frac{1}{(1-\sigma)\sigma'}\boldsymbol{\omega}_{\tilde{m}}^T\boldsymbol{\Omega}^{-1} & \frac{1}{\sigma'} \end{pmatrix},$$

where $\sigma' = \sigma - \frac{1}{1-\sigma}\boldsymbol{\omega}_{\tilde{m}}^T\boldsymbol{\Omega}^{-1}\boldsymbol{\omega}_{\tilde{m}}$. Then the optimization problem is formulated as

$$\min_{\mathbf{w},b} \frac{1}{n_{\tilde{m}}}\sum_{j=1}^{n_{\tilde{m}}}(y_j^{\tilde{m}} - \mathbf{w}^T\mathbf{x}_j^{\tilde{m}} - b)^2 + \frac{\lambda_1'}{2}\|\mathbf{w}\|_2^2 - \lambda_2'\mathbf{u}^T\mathbf{w}, \tag{3.17}$$

where $\lambda_1' = \lambda_1 + \frac{\lambda_2}{\sigma'}$, $\lambda_2' = \frac{\lambda_2}{(1-\sigma)\sigma'}$ and $\mathbf{u} = \mathbf{W}_m\boldsymbol{\Omega}^{-1}\boldsymbol{\omega}_{\tilde{m}}$. We first investigate the physical meaning of problem (3.17) before giving the detailed optimization procedure. We rewrite problem

[1] Due to the analysis in the above section, we find that the optimal solution of $\boldsymbol{\Omega}$ satisfies $\mathrm{tr}(\boldsymbol{\Omega}) = 1$. So here we directly apply this optimality condition.

[2] When $\boldsymbol{\Omega}$ is positive semi-definite, the optimization procedure is similar.

(3.17) as

$$\min_{\mathbf{w},b} \frac{1}{n_{\tilde{m}}} \sum_{j=1}^{n_{\tilde{m}}} (y_j^{\tilde{m}} - \mathbf{w}^T \mathbf{x}_j^{\tilde{m}} - b)^2 + \frac{\lambda_1'}{2} \|\mathbf{w} - \frac{\lambda_2'}{\lambda_1'} \mathbf{u}\|_2^2,$$

which enforces $\mathbf{w}$ to approach the scaled $\mathbf{u}$ as *a priori* information. This problem is similar to that of (Wu & Dietterich, 2004), but there exist crucial differences between them. For example, the model in (Wu & Dietterich, 2004) can only handle the situation that $m = 1$ but our method can handle the situations $m = 1$ and $m > 1$ in a unified framework. Moreover, $\mathbf{u}$ also has explicit physical meaning. Considering a special case when $\mathbf{\Omega} \propto \mathbf{I}_m$ which means the existing tasks are uncorrelated. We can show that $\mathbf{u}$ is proportional to a weighted combination of the model parameters learned from the existing tasks where each combination weight is the task covariance between an existing task and the new task. This is in line with our intuition that a positively correlated existing task has a large weight on the prior of $\mathbf{w}$, an outlier task has negligible contribution and a negatively correlated task even has opposite effect. We reformulate the problem (3.17) as a constrained optimization problem:

$$\begin{aligned} \min_{\mathbf{w},\mathbf{b},\{\varepsilon_j\}} \quad & \frac{1}{n_{\tilde{m}}} \sum_{j=1}^{n_{\tilde{m}}} \varepsilon_j^2 + \frac{\lambda_1'}{2} \|\mathbf{w}\|_2^2 - \lambda_2' \mathbf{u}^T \mathbf{w} \\ \text{s.t.} \quad & y_j^{\tilde{m}} - \mathbf{w}^T \mathbf{x}_j^{\tilde{m}} - b = \varepsilon_j \quad \forall j. \end{aligned}$$

The Lagrangian is given by

$$G' = \frac{1}{n_{\tilde{m}}} \sum_{j=1}^{n_{\tilde{m}}} \varepsilon_j^2 + \frac{\lambda_1'}{2} \|\mathbf{w}\|_2^2 - \lambda_2' \mathbf{u}^T \mathbf{w} + \sum_{j=1}^{n_{\tilde{m}}} \beta_j \left[y_j^{\tilde{m}} - \mathbf{w}^T \mathbf{x}_j^{\tilde{m}} - b - \varepsilon_j \right].$$

We calculate the gradients of G' with respect to $\mathbf{w}$, b and ε_j and set them to 0 to obtain

$$\begin{aligned} \frac{\partial G'}{\partial \mathbf{w}} &= \lambda_1' \mathbf{w} - \lambda_2' \mathbf{u} - \sum_{j=1}^{n_{\tilde{m}}} \beta_j \mathbf{x}_j^{\tilde{m}} = 0 \qquad (3.18) \\ \frac{\partial G'}{\partial b} &= -\sum_{j=1}^{n_{\tilde{m}}} \beta_j = 0 \\ \frac{\partial G'}{\partial \varepsilon_j} &= \frac{2}{n_{\tilde{m}}} \varepsilon_j - \beta_j = 0. \end{aligned}$$

Combining the above equations, we obtain the following linear system:

$$\begin{pmatrix} \frac{1}{\lambda_1'} \mathbf{K}' + \frac{n_{\tilde{m}}}{2} \mathbf{I}_{n_{\tilde{m}}} & \mathbf{1}_{n_{\tilde{m}}} \\ \mathbf{1}_{n_{\tilde{m}}}^T & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{\beta} \\ b \end{pmatrix} = \begin{pmatrix} \mathbf{y}' - \frac{\lambda_2'}{\lambda_1'} (\mathbf{X}')^T \mathbf{u} \\ 0 \end{pmatrix}, \qquad (3.19)$$

where $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_{n_{\tilde{m}}})^T$, $\mathbf{X}' = (\mathbf{x}_1^{\tilde{m}}, \ldots, \mathbf{x}_{n_{\tilde{m}}}^{\tilde{m}})$, $\mathbf{y}' = (\mathbf{x}_1^{\tilde{m}}, \ldots, \mathbf{x}_{n_{\tilde{m}}}^{\tilde{m}})^T$ and $\mathbf{K}' = (\mathbf{X}')^T \mathbf{X}'$ is the linear kernel matrix on the training set of the new task.

When optimizing with respect to $\boldsymbol{\omega}_{m+1}$ and σ, the optimization problem is formulated as

$$\begin{aligned} \min_{\boldsymbol{\omega}_{\tilde{m}},\sigma,\tilde{\boldsymbol{\Omega}}} \quad & \mathrm{tr}(\mathbf{W}_{\tilde{m}}\tilde{\boldsymbol{\Omega}}^{-1}\mathbf{W}_{\tilde{m}}^T) \\ \text{s.t.} \quad & \boldsymbol{\omega}_{\tilde{m}}^T\boldsymbol{\Omega}^{-1}\boldsymbol{\omega}_{\tilde{m}} \leq \sigma - \sigma^2 \\ & \tilde{\boldsymbol{\Omega}} = \begin{pmatrix} (1-\sigma)\boldsymbol{\Omega} & \boldsymbol{\omega}_{\tilde{m}} \\ \boldsymbol{\omega}_{\tilde{m}}^T & \sigma \end{pmatrix}. \end{aligned} \tag{3.20}$$

We impose a constraint as $\mathbf{W}_{\tilde{m}}\tilde{\boldsymbol{\Omega}}^{-1}\mathbf{W}_{\tilde{m}}^T \preceq \frac{1}{t}\mathbf{I}_d$ and the objective function becomes $\frac{1}{t}$ which is equivalent to a minimization optimization problem $\min -t$ since $t > 0$. Using the Schur complement, we can get

$$\mathbf{W}_{\tilde{m}}\tilde{\boldsymbol{\Omega}}^{-1}\mathbf{W}_{\tilde{m}}^T \preceq \frac{1}{t}\mathbf{I}_d \Longleftrightarrow \begin{pmatrix} \tilde{\boldsymbol{\Omega}} & \mathbf{W}_{\tilde{m}}^T \\ \mathbf{W}_{\tilde{m}} & \frac{1}{t}\mathbf{I}_d \end{pmatrix} \succeq \mathbf{0}.$$

By using the Schur complement again, we get

$$\begin{pmatrix} \tilde{\boldsymbol{\Omega}} & \mathbf{W}_{\tilde{m}}^T \\ \mathbf{W}_{\tilde{m}} & \frac{1}{t}\mathbf{I}_d \end{pmatrix} \succeq \mathbf{0} \Longleftrightarrow \tilde{\boldsymbol{\Omega}} - t\mathbf{W}_{\tilde{m}}^T\mathbf{W}_{\tilde{m}} \succeq \mathbf{0}.$$

So problem (3.20) can be formulated as

$$\begin{aligned} \min_{\boldsymbol{\omega}_{\tilde{m}},\sigma,\tilde{\boldsymbol{\Omega}},t} \quad & -t \\ \text{s.t.} \quad & \boldsymbol{\omega}_{\tilde{m}}^T\boldsymbol{\Omega}^{-1}\boldsymbol{\omega}_{\tilde{m}} \leq \sigma - \sigma^2 \\ & \tilde{\boldsymbol{\Omega}} = \begin{pmatrix} (1-\sigma)\boldsymbol{\Omega} & \boldsymbol{\omega}_{\tilde{m}} \\ \boldsymbol{\omega}_{\tilde{m}}^T & \sigma \end{pmatrix}. \\ & \tilde{\boldsymbol{\Omega}} - t\mathbf{W}_{\tilde{m}}^T\mathbf{W}_{\tilde{m}} \succeq \mathbf{0}, \end{aligned} \tag{3.21}$$

which is an SDP problem. In real applications, the number of tasks m is usually not very large and we can use a standard SDP solver to solve problem (3.21). Moreover, we may also reformulate problem (3.21) as a second-order cone programming (SOCP) problem (Lobo et al., 1998) which is more efficient than SDP when m is large. We will present the procedure in Appendix A.2.

In case two or more new tasks arrive together, the above formulation only needs to be modified slightly to accommodate all the new tasks simultaneously.

3.2.4 Kernel Extension

So far we have only considered the linear case for MTRL. In this section, we will apply the kernel trick to provide a nonlinear extension of the algorithm presented above.

The optimization problem for the kernel extension is essentially the same as that for the linear case, with the only difference being that the data point $\mathbf{x}_j^i$ is mapped to $\Phi(\mathbf{x}_j^i)$ in some reproducing kernel Hilbert space where $\Phi(\cdot)$ denotes the feature map. Then the corresponding kernel function $k(\cdot,\cdot)$ satisfies $k(\mathbf{x}_1,\mathbf{x}_2)=\Phi(\mathbf{x}_1)^T\Phi(\mathbf{x}_2)$.

For symmetric multi-task learning, we can also use an alternating method to solve the optimization problem. In the first step of the alternating method, we use the nonlinear multi-task kernel

$$k_{MT}(\mathbf{x}_{j_1}^{i_1},\mathbf{x}_{j_2}^{i_2})=\mathbf{e}_{i_1}^T\mathbf{\Omega}(\lambda_1\mathbf{\Omega}+\lambda_2\mathbf{I}_m)^{-1}\mathbf{e}_{i_2}k(\mathbf{x}_{j_1}^{i_1},\mathbf{x}_{j_2}^{i_2}).$$

The rest is the same as the linear case. For the second step, the change needed is in the calculation of $\mathbf{W}^T\mathbf{W}$. Since

$$\mathbf{W}=\sum_{i=1}^{m}\sum_{j=1}^{n_i}\alpha_j^i\Phi(\mathbf{x}_j^i)\mathbf{e}_i^T\mathbf{\Omega}(\lambda_1\mathbf{\Omega}+\lambda_2\mathbf{I}_m)^{-1}$$

which is similar to the representer theorem in single-task learning, we have

$$\mathbf{W}^T\mathbf{W}=\sum_{i,j}\sum_{p,q}\alpha_j^i\alpha_q^pk(\mathbf{x}_j^i,\mathbf{x}_q^p)(\lambda_1\mathbf{\Omega}+\lambda_2\mathbf{I}_m)^{-1}\mathbf{\Omega}\mathbf{e}_i\mathbf{e}_p^T\mathbf{\Omega}(\lambda_1\mathbf{\Omega}+\lambda_2\mathbf{I}_m)^{-1}. \tag{3.22}$$

In the asymmetric setting, when a new task arrives, we still use the alternating method to solve the problem. In the first step of the alternating method, the analytical solution (3.19) needs to calculate $(\Phi(\mathbf{X}'))^T\mathbf{u}$ where $\Phi(\mathbf{X}')=(\Phi(\mathbf{x}_1^{\tilde{m}}),\ldots,\Phi(\mathbf{x}_{n_{\tilde{m}}}^{\tilde{m}}))$ denotes the data matrix of the new task after feature mapping and $\mathbf{u}=\mathbf{W}_m\mathbf{\Omega}^{-1}\boldsymbol{\omega}_{\tilde{m}}$. Since $\mathbf{W}_m$ is derived from symmetric multi-task learning, we get

$$\mathbf{W}_m=\sum_{i=1}^{m}\sum_{j=1}^{n_i}\alpha_j^i\Phi(\mathbf{x}_j^i)\mathbf{e}_i^T\mathbf{\Omega}(\lambda_1\mathbf{\Omega}+\lambda_2\mathbf{I}_m)^{-1}$$

Then

$$(\Phi(\mathbf{X}'))^T\mathbf{u}=(\Phi(\mathbf{X}'))^T\mathbf{W}_m\mathbf{\Omega}^{-1}\boldsymbol{\omega}_{\tilde{m}}=\mathbf{M}\mathbf{\Omega}^{-1}\boldsymbol{\omega}_{\tilde{m}}$$

where

$$\begin{aligned}\mathbf{M} &= (\Phi(\mathbf{X}'))^T\mathbf{W}_m\\ &= (\Phi(\mathbf{X}'))^T\sum_{i=1}^{m}\sum_{j=1}^{n_i}\alpha_j^i\Phi(\mathbf{x}_j^i)\mathbf{e}_i^T\mathbf{\Omega}(\lambda_1\mathbf{\Omega}+\lambda_2\mathbf{I}_m)^{-1}\\ &= \sum_{i=1}^{m}\sum_{j=1}^{n_i}\alpha_j^i\tilde{\mathbf{k}}_j^i\mathbf{e}_i^T\mathbf{\Omega}(\lambda_1\mathbf{\Omega}+\lambda_2\mathbf{I}_m)^{-1}\end{aligned}$$

and $\tilde{\mathbf{k}}_j^i=\left(k(\mathbf{x}_j^i,\mathbf{x}_1^{\tilde{m}}),\ldots,k(\mathbf{x}_j^i,\mathbf{x}_{n_{\tilde{m}}}^{\tilde{m}})\right)^T$. In the second step of the alternating method, we need to calculate $\mathbf{W}_{\tilde{m}}^T\mathbf{W}_{\tilde{m}}$ where $\mathbf{W}_{\tilde{m}}=(\mathbf{W}_m,\mathbf{w})$. Following the notations in Appendix A.2,

we denote $\mathbf{W}_{\tilde{m}}^T\mathbf{W}_{\tilde{m}}$ as $\mathbf{W}_{\tilde{m}}^T\mathbf{W}_{\tilde{m}} = \begin{pmatrix} \mathbf{\Psi}_{11} & \mathbf{\Psi}_{12} \\ \mathbf{\Psi}_{12}^T & \Psi_{22} \end{pmatrix}$ where $\mathbf{\Psi}_{11} \in \mathbb{R}^{m\times m}$, $\mathbf{\Psi}_{12} \in \mathbb{R}^{m\times 1}$ and $\Psi_{22} \in \mathbb{R}$. Then $\mathbf{\Psi}_{11} = \mathbf{W}_m^T\mathbf{W}_m$, $\mathbf{\Psi}_{12} = \mathbf{W}_m^T\mathbf{w}$ and $\Psi_{22} = \mathbf{w}^T\mathbf{w}$. It is easy to show that $\mathbf{\Psi}_{11}$ can be calculated as in Eq. (3.22) which only needs to be computed once. Recall that

$$\mathbf{w} = \frac{\lambda_2'}{\lambda_1'}\mathbf{u} + \frac{1}{\lambda_1'}\Phi(\mathbf{X}')\boldsymbol{\beta} = \frac{\lambda_2'}{\lambda_1'}\mathbf{W}_m\mathbf{\Omega}^{-1}\boldsymbol{\omega}_{\tilde{m}} + \frac{1}{\lambda_1'}\Phi(\mathbf{X}')\boldsymbol{\beta}$$

from Eq. (3.18). So we can get

$$\begin{aligned} \mathbf{\Psi}_{12} &= \mathbf{W}_m^T\Big[\frac{\lambda_2'}{\lambda_1'}\mathbf{W}_m\mathbf{\Omega}^{-1}\boldsymbol{\omega}_{\tilde{m}} + \frac{1}{\lambda_1'}\Phi(\mathbf{X}')\boldsymbol{\beta}\Big] \\ &= \frac{\lambda_2'}{\lambda_1'}\mathbf{\Psi}_{11}\mathbf{\Omega}^{-1}\boldsymbol{\omega}_{\tilde{m}} + \frac{1}{\lambda_1'}\mathbf{M}^T\boldsymbol{\beta} \end{aligned}$$

and

$$\begin{aligned} \Psi_{22} =& \Big\|\frac{\lambda_2'}{\lambda_1'}\mathbf{W}_m\mathbf{\Omega}^{-1}\boldsymbol{\omega}_{\tilde{m}} + \frac{1}{\lambda_1'}\Phi(\mathbf{X}')\boldsymbol{\beta}\Big\|_2^2 \\ =& \frac{(\lambda_2')^2}{(\lambda_1')^2}\boldsymbol{\omega}_{\tilde{m}}^T\mathbf{\Omega}^{-1}\mathbf{W}_m^T\mathbf{W}_m\mathbf{\Omega}^{-1}\boldsymbol{\omega}_{\tilde{m}} + \frac{1}{(\lambda_1')^2}\boldsymbol{\beta}^T\Phi(\mathbf{X}')^T\Phi(\mathbf{X}')\boldsymbol{\beta} + \frac{2\lambda_2'}{(\lambda_1')^2}\boldsymbol{\omega}_{\tilde{m}}^T\mathbf{\Omega}^{-1}\mathbf{W}_m^T\Phi(\mathbf{X}')\boldsymbol{\beta} \\ =& \frac{(\lambda_2')^2}{(\lambda_1')^2}\boldsymbol{\omega}_{\tilde{m}}^T\mathbf{\Omega}^{-1}\mathbf{\Psi}_{11}\mathbf{\Omega}^{-1}\boldsymbol{\omega}_{\tilde{m}} + \frac{1}{(\lambda_1')^2}\boldsymbol{\beta}^T\mathbf{K}'\boldsymbol{\beta} + \frac{2\lambda_2'}{(\lambda_1')^2}\boldsymbol{\omega}_{\tilde{m}}^T\mathbf{\Omega}^{-1}\mathbf{M}^T\boldsymbol{\beta} \end{aligned}$$

where $\mathbf{K}'$ is the kernel matrix. In the testing phrase, when given a test data point $\mathbf{x}_\star$, the output can be calculated as

$$\begin{aligned} y_\star &= \mathbf{w}^T\Phi(\mathbf{x}_\star) + b \\ &= \Big(\frac{\lambda_2'}{\lambda_1'}\mathbf{W}_m\mathbf{\Omega}^{-1}\boldsymbol{\omega}_{\tilde{m}} + \frac{1}{\lambda_1'}\Phi(\mathbf{X}')\boldsymbol{\beta}\Big)^T\Phi(\mathbf{x}_\star) + b \\ &= \frac{\lambda_2'}{\lambda_1'}\boldsymbol{\omega}_{\tilde{m}}^T\mathbf{\Omega}^{-1}\mathbf{W}_m^T\Phi(\mathbf{x}_\star) + \frac{1}{\lambda_1'}\boldsymbol{\beta}^T\mathbf{k}_\star + b \\ &= \frac{\lambda_2'}{\lambda_1'}\boldsymbol{\omega}_{\tilde{m}}^T\mathbf{\Omega}^{-1}(\lambda_1\mathbf{\Omega} + \lambda_2\mathbf{I}_m)^{-1}\mathbf{\Omega}\sum_{i=1}^{m}\sum_{j=1}^{n_i}\alpha_j^i k(\mathbf{x}_j^i, \mathbf{x}_\star)\mathbf{e}_i + \frac{1}{\lambda_1'}\boldsymbol{\beta}^T\mathbf{k}_\star + b \\ &= \frac{\lambda_2'}{\lambda_1'}\boldsymbol{\omega}_{\tilde{m}}^T(\lambda_1\mathbf{\Omega} + \lambda_2\mathbf{I}_m)^{-1}\sum_{i=1}^{m}\sum_{j=1}^{n_i}\alpha_j^i k(\mathbf{x}_j^i, \mathbf{x}_\star)\mathbf{e}_i + \frac{1}{\lambda_1'}\boldsymbol{\beta}^T\mathbf{k}_\star + b, \end{aligned}$$

where $\mathbf{k}_\star = \big(k(\mathbf{x}_\star, \mathbf{x}_1^{\tilde{m}}), \ldots, k(\mathbf{x}_\star, \mathbf{x}_{n_{\tilde{m}}}^{\tilde{m}})\big)^T$.

3.2.5 Discussions

In some applications, there may exist prior knowledge about the relationships between some tasks, e.g., two tasks are more similar than two other tasks, some tasks are from the same task

cluster, etc. It is easy to incorporate the prior knowledge by introducing additional constraints into problem (3.8). For example, if tasks T_i and T_j are more similar than tasks T_p and T_q, then the corresponding constraint can be represented as $\Omega_{ij} > \Omega_{pq}$; if we know that some tasks are from the same cluster, then we can enforce the covariances between these tasks very large while their covariances with other tasks very close to 0.

3.2.6 Some Variants

In our probabilistic model, the prior on $\mathbf{W}$ given in Eq. (3.2) is very general. Here we discuss some different choices for $q(\mathbf{W})$.

Utilizing Other Matrix Variate Normal Distributions

When we choose another matrix variate normal distribution for $q(\mathbf{W})$, such as

$$q(\mathbf{W}) = \mathcal{MN}_{d\times m}(\mathbf{W} \mid \mathbf{0}_{d\times m}, \mathbf{\Sigma} \otimes \mathbf{I}_m),$$

it leads to a formulation similar to multi-task feature learning (Argyriou et al., 2008a, 2008):

$$\begin{aligned} \min_{\mathbf{W},\mathbf{b},\mathbf{\Sigma}} & \sum_{i=1}^{m} \frac{1}{n_i} \sum_{j=1}^{n_i} (y_j^i - \mathbf{w}_i^T \mathbf{x}_j^i - b_i)^2 + \frac{\lambda_1}{2} \mathrm{tr}(\mathbf{W}\mathbf{W}^T) + \frac{\lambda_2}{2} \mathrm{tr}(\mathbf{W}^T \mathbf{\Sigma}^{-1} \mathbf{W}) \\ \text{s.t. } & \mathbf{\Sigma} \succeq 0 \\ & \mathrm{tr}(\mathbf{\Sigma}) \leq 1. \end{aligned}$$

From this aspect, we can understand the difference between our method and multi-task feature learning. Multi-task feature learning is to learn the covariance structure on the model parameters and the parameters of different tasks are independent given the covariance structure. However, the task relationship is not very clear in this method in that we do not know which task is helpful. In our formulation (3.8), the relationships between tasks are described explicitly in the task covariance matrix $\mathbf{\Omega}$. Another advantage of formulation (3.8) is that kernel extension is very natural as that in single-task learning. For multi-task feature learning, however, Gram-Schmidt orthogonalization on the kernel matrix is needed and hence it will incur additional computational cost.

The above choices for $q(\mathbf{W})$ either assume the tasks are correlated but the data features are independent, or the data features are correlated but the tasks are independent. Here we can generalize them to the case which assumes that the tasks and the data features are both correlated by defining $q(\mathbf{W})$ as $q(\mathbf{W}) = \mathcal{MN}_{d\times m}(\mathbf{W} \mid \mathbf{0}_{d\times m}, \mathbf{\Sigma} \otimes \mathbf{\Omega})$ where $\mathbf{\Sigma}$ describes the correlations between data features and $\mathbf{\Omega}$ models the correlations between tasks. Then the

corresponding optimization problem becomes

$$\begin{aligned}\min_{\mathbf{W},\mathbf{b},\mathbf{\Sigma},\mathbf{\Omega}} \quad & \sum_{i=1}^{m}\frac{1}{n_i}\sum_{j=1}^{n_i}(y_j^i-\mathbf{w}_i^T\mathbf{x}_j^i-b_i)^2+\frac{\lambda_1}{2}\mathrm{tr}(\mathbf{W}\mathbf{W}^T)+\frac{\lambda_2}{2}\mathrm{tr}(\mathbf{W}^T\mathbf{\Sigma}^{-1}\mathbf{W}\mathbf{\Omega}^{-1}) \\ \text{s.t.} \quad & \mathbf{\Sigma}\succeq 0,\ \mathrm{tr}(\mathbf{\Sigma})\leq 1 \\ & \mathbf{\Omega}\succeq 0,\ \mathrm{tr}(\mathbf{\Omega})\leq 1. \end{aligned} \tag{3.23}$$

Unfortunately this optimization problem is non-convex due to the third term in the objective function which makes the performance of this model sensitive to the initial values of the model parameters. But we can also use the alternating method to obtain a locally optimal solution. Moreover, the kernel extension of this method is not very easy to derive since we cannot estimate the covariance matrix $\mathbf{\Sigma}$ for feature correlation in an infinite-dimensional kernel space. But we can also get an approximation by assuming that the primal space of $\mathbf{\Sigma}$ is spanned by the training data points in the kernel space, which is similar to the representer theorem in (Argyriou et al., 2008a). Compared with this problem, problem (3.8) is convex and its kernel extension is very natural. Moreover, for problem (3.8), the feature correlations can be considered in the construction of the kernel function by using the following linear and RBF kernels:

$$\begin{aligned} k_{linear}(\mathbf{x}_1,\mathbf{x}_2) &= \mathbf{x}_1^T\mathbf{\Sigma}^{-1}\mathbf{x}_2 \\ k_{rbf}(\mathbf{x}_1,\mathbf{x}_2) &= \exp\Big(-(\mathbf{x}_1-\mathbf{x}_2)^T\mathbf{\Sigma}^{-1}(\mathbf{x}_1-\mathbf{x}_2)/2\Big). \end{aligned}$$

Utilizing Matrix Variate t Distribution

It is well known that the t distribution has heavy-tail behavior which makes it more robust against outliers than the corresponding normal distribution. This also holds for the matrix variate normal distribution and the matrix variate t distribution (Gupta & Nagar, 2000). So we can use the matrix variate t distribution for $q(\mathbf{W})$ to make the model more robust.

We assign the matrix variate t distribution to $q(\mathbf{W})$:

$$q(\mathbf{W})=\mathcal{MT}_{d\times m}(\nu,\mathbf{0}_{d\times m},\mathbf{I}_d\otimes\mathbf{\Omega}).$$

The p.d.f. of the matrix variate t distribution is defined in Definition 2.2. Then the corresponding optimization problem can be formulated as

$$\begin{aligned}\min_{\mathbf{W},\mathbf{b},\mathbf{\Omega}} \quad & \sum_{i=1}^{m}\frac{1}{n_i}\sum_{j=1}^{n_i}(y_j^i-\mathbf{w}_i^T\mathbf{x}_j^i-b_i)^2+\frac{\lambda_1}{2}\mathrm{tr}(\mathbf{W}\mathbf{W}^T)+\frac{\lambda_2}{2}\ln|\mathbf{I}_d+\mathbf{W}\mathbf{\Omega}^{-1}\mathbf{W}^T| \\ \text{s.t.} \quad & \mathbf{\Omega}\succeq 0 \\ & \mathrm{tr}(\mathbf{\Omega})\leq 1. \end{aligned}$$

This is a non-convex optimization problem due to the non-convexity of the last term in the objective function. By using Lemma 3.1, we can obtain

$$\ln|\mathbf{I}_d + \mathbf{W}\mathbf{\Omega}^{-1}\mathbf{W}^T| \leq \mathrm{tr}(\mathbf{I}_d + \mathbf{W}\mathbf{\Omega}^{-1}\mathbf{W}^T) - d = \mathrm{tr}(\mathbf{W}\mathbf{\Omega}^{-1}\mathbf{W}^T). \tag{3.24}$$

So the objective function of problem (3.8) is the upper bound of that in this problem and hence this problem can be relaxed to the convex problem (3.8). Moreover, we may also use the majorization-minimization (MM) algorithm (Lange et al., 2000) to solve this problem. The MM algorithm is an iterative algorithm which seeks an upper bound of the objective function based on the solution from the previous iteration as a surrogate function for the minimization problem and then optimizes with respect to the surrogate function. The MM algorithm is guaranteed to find a local optimum and is widely used in many optimization problems. For our problem, we denote the solution of $\mathbf{W}$, $\mathbf{b}$ and $\mathbf{\Omega}$ in the t-th iteration as $\mathbf{W}^{(t)}$, $\mathbf{b}^{(t)}$ and $\mathbf{\Omega}^{(t)}$. Then by using Lemma 3.1, we can obtain

$$\begin{aligned}\ln|\mathbf{I}_d + \mathbf{W}\mathbf{\Omega}^{-1}\mathbf{W}^T| - \ln|\mathbf{M}| &= \ln|\mathbf{M}^{-1}(\mathbf{I}_d + \mathbf{W}\mathbf{\Omega}^{-1}\mathbf{W}^T)| \\ &\leq \mathrm{tr}\big(\mathbf{M}^{-1}(\mathbf{I}_d + \mathbf{W}\mathbf{\Omega}^{-1}\mathbf{W}^T)\big) - d,\end{aligned}$$

where $\mathbf{M} = \mathbf{I}_d + \mathbf{W}^{(t)}(\mathbf{\Omega}^{(t)})^{-1}(\mathbf{W}^{(t)})^T$. So we can get

$$\ln|\mathbf{I}_d + \mathbf{W}\mathbf{\Omega}^{-1}\mathbf{W}^T| \leq \mathrm{tr}\big(\mathbf{M}^{-1}(\mathbf{I}_d + \mathbf{W}\mathbf{\Omega}^{-1}\mathbf{W}^T)\big) + \ln|\mathbf{M}| - d. \tag{3.25}$$

We can prove that this bound is tighter than the previous one in Eq. (3.24) and the proof is given in Appendix A.3. So in the $(t+1)$-th iteration, the MM algorithm is to solve the following optimization problem:

$$\begin{aligned}\min_{\mathbf{W},\mathbf{b},\mathbf{\Omega}} & \sum_{i=1}^{m}\frac{1}{n_i}\sum_{j=1}^{n_i}(y_j^i - \mathbf{w}_i^T\mathbf{x}_j^i - b_i)^2 + \frac{\lambda_1}{2}\mathrm{tr}(\mathbf{W}\mathbf{W}^T) + \frac{\lambda_2}{2}\mathrm{tr}(\mathbf{W}^T\mathbf{M}^{-1}\mathbf{W}\mathbf{\Omega}^{-1}) \\ \text{s.t. } & \mathbf{\Omega} \succeq 0 \\ & \mathrm{tr}(\mathbf{\Omega}) \leq 1.\end{aligned}$$

This problem is similar to problem (3.23) with the difference that $\mathbf{\Sigma}$ in problem (3.23) is a variable but here $\mathbf{M}$ is a constant matrix. Similar formulations lead to similar limitations though. For example, the optimization problem is non-convex and the kernel extension is not very natural and needs some approximation.

3.3 Relationships with Existing Methods

3.3.1 Relationships with Existing Regularized Multi-Task Learning Methods

Some existing multi-task learning methods (Evgeniou & Pontil, 2004; Evgeniou et al., 2005; Kato et al., 2007; Jacob et al., 2008) also model the relationships between tasks under the

regularization framework. The methods in (Evgeniou & Pontil, 2004; Evgeniou et al., 2005; Kato et al., 2007) assume that the task relationships are given *a priori* and then utilize this prior knowledge to learn the model parameters. On the other hand, the method in (Jacob et al., 2008) learns the task cluster structure from data. In this section, we discuss the relationships between MTRL and these methods.

The objective functions of the methods in (Evgeniou & Pontil, 2004; Evgeniou et al., 2005; Kato et al., 2007; Jacob et al., 2008) are all of the following form which is similar to that of problem (3.8):

$$J = \sum_{i=1}^{m} \sum_{j=1}^{n_i} l(y_j^i, \mathbf{w}_i^T \mathbf{x}_j^i + b_i) + \frac{\lambda_1}{2} \mathrm{tr}(\mathbf{W}\mathbf{W}^T) + \frac{\lambda_2}{2} f(\mathbf{W}),$$

with different choices for the formulation of $f(\cdot)$. The first term in this objective function denotes the loss on the labeled data of all tasks, the second term is to control the complexity of the model parameters $\mathbf{W}$ and the last term is to measure the task relationship based on $\mathbf{W}$.

The method in (Evgeniou & Pontil, 2004) assumes that all tasks are similar and so the parameter vector of each task is similar to the average parameter vector. The corresponding formulation for $f(\cdot)$ is given by

$$f(\mathbf{W}) = \sum_{i=1}^{m} \left\| \mathbf{w}_i - \frac{1}{m} \sum_{j=1}^{m} \mathbf{w}_j \right\|_2^2.$$

After some algebraic operations, we can rewrite $f(\mathbf{W})$ as

$$f(\mathbf{W}) = \sum_{i=1}^{m} \sum_{j=1}^{m} \frac{1}{2m} \|\mathbf{w}_i - \mathbf{w}_j\|_2^2 = \mathrm{tr}(\mathbf{W}\mathbf{L}\mathbf{W}^T),$$

where $\mathbf{L}$ is the Laplacian matrix (Chung, 1997) defined on a fully connected graph with edge weights equal to $\frac{1}{2m}$. This corresponds to a special case of MTRL with $\mathbf{\Omega}^{-1} = \mathbf{L}$. Obviously, a limitation of this method is that only positive task correlation can be modeled.

The methods in (Evgeniou et al., 2005) assume that the task cluster structure or the task similarity between tasks is given. $f(\cdot)$ is formulated as

$$f(\mathbf{W}) = \sum_{i,j} s_{ij} \|\mathbf{w}_i - \mathbf{w}_j\|_2^2 = \mathrm{tr}(\mathbf{W}\mathbf{L}\mathbf{W}^T),$$

where $s_{ij} \geq 0$ denotes the similarity between tasks T_i and T_j and $\mathbf{L}$ is the Laplacian matrix defined on the graph based on $\{s_{ij}\}$. Again, it corresponds to a special case of MTRL with $\mathbf{\Omega}^{-1} = \mathbf{L}$. Note that this method requires that $s_{ij} \geq 0$ and so it also can only model positive task correlation and task unrelatedness. If negative task correlation is modeled as well, the

problem will become non-convex making it more difficult to solve. Moreover, in many real-world applications, prior knowledge about s_{ij} is not available.

In (Kato et al., 2007) the authors assume the existence of a task network and that the neighbors in the task network, encoded as index pairs (p_k, q_k), are very similar. $f(\cdot)$ can be formulated as

$$f(\mathbf{W}) = \sum_k \|\mathbf{w}_{p_k} - \mathbf{w}_{q_k}\|_2^2.$$

We can define a similarity matrix G whose (p_k, q_k)th elements are equal to 1 for all k and 0 otherwise. Then $f(\mathbf{W})$ can be simplified as $f(\mathbf{W}) = \mathrm{tr}(\mathbf{W}\mathbf{L}\mathbf{W}^T)$ where $\mathbf{L}$ is the Laplacian matrix of G, which is similar to (Evgeniou et al., 2005). Thus it also corresponds to a special case of MTRL with $\mathbf{\Omega}^{-1} = \mathbf{L}$. Similar to (Evgeniou et al., 2005), a difficulty of this method is that prior knowledge in the form of a task network is not available in many applications.

The method in (Jacob et al., 2008) is more general in that it learns the task cluster structure from data, making it more suitable for real-world applications. The formulation for $f(\cdot)$ is described as

$$f(\mathbf{W}) = \mathrm{tr}\left(\mathbf{W}\left[\alpha\mathbf{H}_m + \beta(\mathbf{M} - \mathbf{H}_m) + \gamma(\mathbf{I}_m - \mathbf{M})\right]\mathbf{W}^T\right),$$

where $\mathbf{H}_m$ is the centering matrix and $\mathbf{M} = \mathbf{E}(\mathbf{E}^T\mathbf{E})\mathbf{E}^T$ with the cluster assignment matrix $\mathbf{E}$. If we let $\mathbf{\Omega}^{-1} = \alpha\mathbf{H}_m + \beta(\mathbf{M} - \mathbf{H}_m) + \gamma(\mathbf{I}_m - \mathbf{M})$ or $\mathbf{\Omega} = \frac{1}{\alpha}\mathbf{H}_m + \frac{1}{\beta}(\mathbf{M} - \mathbf{H}_m) + \frac{1}{\gamma}(\mathbf{I}_m - \mathbf{M})$, MTRL will reduce to this method. However, (Jacob et al., 2008) is a local method which can only model positive task correlations within each cluster but cannot model negative task correlations among different task clusters. Another difficulty of this method lies in determining the number of task clusters.

Compared with existing methods, MTRL is very appealing in that it can learn all three types of task relationships in a nonparametric way. This makes it easy to identify the tasks that are useful for multi-task learning and those that should not be exploited.

3.3.2 Relationships with Multi-Task Gaussian Process

The multi-task GP model in (Bonilla et al., 2007) directly models the task covariance matrix $\mathbf{\Sigma}$ by incorporating it into the GP prior as follows:

$$\langle f_j^i, f_s^r \rangle = \Sigma_{ir} k(\mathbf{x}_j^i, \mathbf{x}_s^r), \tag{3.26}$$

where $\langle \cdot, \cdot \rangle$ denotes the covariance of two random variables, f_j^i is the latent function value for $\mathbf{x}_j^i$, and Σ_{ir} is the (i, r)th element of $\mathbf{\Sigma}$. The output y_j^i given f_j^i is distributed as

$$y_j^i \mid f_j^i \sim \mathcal{N}(f_j^i, \sigma_i^2),$$

which defines the likelihood for $\mathbf{x}_j^i$. Here σ_i^2 is the noise level of the ith task.

Recall that GP has an interpretation from the weight-space view (Rasmussen & Williams, 2006). In our previous work (Zhang & Yeung, 2010b), we also give a weight-space view of this multi-task GP model:

$$\begin{aligned} y_j^i &= \mathbf{w}_i^T \phi(\mathbf{x}_j^i) + \varepsilon_j^i \\ \mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_m] &\sim \mathcal{MN}_{d' \times m}(\mathbf{0}_{d' \times m}, \mathbf{I}_{d'} \otimes \mathbf{\Sigma}) \\ \varepsilon_j^i &\sim \mathcal{N}(0, \sigma_i^2), \end{aligned} \tag{3.27}$$

where $\phi(\cdot)$, which maps $\mathbf{x} \in \mathbb{R}^d$ to $\phi(\mathbf{x}) \in \mathbb{R}^{d'}$ and may have no explicit form, denotes a feature mapping corresponding to the kernel function $k(\cdot, \cdot)$. The equivalence between the model formulations in (3.26) and (3.27) is due to the following which is a consequence of the property of the matrix variate normal distribution:[3]

$$\begin{aligned} f_j^i &\stackrel{\text{def}}{=} \phi(\mathbf{x}_j^i)^T \mathbf{w}_i = \phi(\mathbf{x}_j^i)^T \mathbf{W} \mathbf{e}_{m,i} \sim \mathcal{N}(0, \Sigma_{ii} k(\mathbf{x}_j^i, \mathbf{x}_j^i)) \\ \langle f_j^i, f_s^r \rangle &= \int \phi(\mathbf{x}_j^i)^T \mathbf{W} \mathbf{e}_{m,i} \mathbf{e}_{m,r}^T \mathbf{W}^T \phi(\mathbf{x}_s^r) p(\mathbf{W}) d\mathbf{W} = \Sigma_{ir} k(\mathbf{x}_j^i, \mathbf{x}_s^r), \end{aligned}$$

where $\mathbf{e}_{m,i}$ is the ith column of $\mathbf{I}_m$. The weight-space view of the conventional GP can be seen as a special case of that of the multi-task GP with $m = 1$, under which the prior for $\mathbf{W}$ in (3.27) will become the ordinary normal distribution with zero mean and identity covariance matrix by setting $\mathbf{\Sigma} = 1$.

It is easy to see that the weight-space view model (3.27) is similar to our probabilistic model which shows the relationship of our method with multi-task GP. However, the optimization problem in (Bonilla et al., 2007) is non-convex which makes the multi-task GP more sensitive to the initial values of model parameters. To reduce the number of model parameters, multi-task GP seeks a low-rank approximation of the task covariance matrix which may weaken the expressive power of the task covariance matrix and limit the performance of the model. Moreover, since multi-task GP is based on the GP model, the complexity of multi-task GP is cubic with respect to the number of data points in all tasks. This high complexity requirement may limit the use of multi-task GP for large-scale applications.

3.4 Experiments

In this section, we study MTRL empirically on some data sets and compare it with a single-task learning (STL) method, multi-task feature learning (MTFL) (Argyriou et al., 2008a) method[4] and a multi-task GP (MTGP) method (Bonilla et al., 2007) which can also learn the global task relationships.

[3]The proofs for the following two equations can be found in Chapter 4.

[4]The implementation can be downloaded from http://www.cs.ucl.ac.uk/staff/A.Argyriou/code/.

3.4.1 Toy Problem

We first generate a toy data set to conduct a "proof of concept" experiment before we do experiments on real data sets. The toy data set is generated as follows. The regression functions corresponding to three regression tasks are defined as $y = 3x + 10$, $y = -3x - 5$ and $y = 1$. For each task, we randomly sample five points uniformly from $[0, 10]$. Each function output is corrupted by a Gaussian noise process with zero mean and variance equal to 0.1. One example of the data set is plotted in Figure 3.1, with each color (and point type) corresponding to one task. We repeat the experiment 10 times. From the coefficients of the regression functions, we expect the correlation between the first two tasks to approach -1 and those for the other two pairs of tasks to approach 0. To apply MTRL, we use the linear kernel and set λ_1 to 0.01 and λ_2 to 0.005. After the learning procedure converges, we find that the mean estimated regression functions for the three tasks are $y = 2.9964x + 10.0381$, $y = -3.0022x - 4.9421$ and $y = 0.0073x + 0.9848$. Based on the task covariance matrix learned, we obtain the following the mean task correlation matrix:

$$\mathbf{C} = \begin{pmatrix} 1.0000 & -0.9985 & 0.0632 \\ -0.9985 & 1.0000 & -0.0623 \\ 0.0632 & -0.0623 & 1.0000 \end{pmatrix}.$$

We can see that the task correlations learned confirm our expectation, showing that MTRL can indeed learn the relationships between tasks for this toy problem.

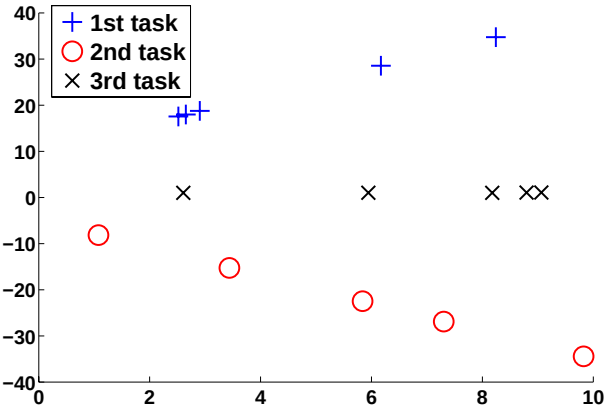


Figure 3.1: One example of the toy problem. The data points with each color (and point type) correspond to one task.

3.4.2 Robot Inverse Dynamics

We now study the problem of learning the inverse dynamics of a 7-DOF SARCOS anthropomorphic robot arm[5]. Each observation in the SARCOS data set consists of 21 input features,

[5]http://www.gaussianprocess.org/gpml/data/

corresponding to seven joint positions, seven joint velocities and seven joint accelerations, as well as seven joint torques for the seven degrees of freedom (DOF). Thus the input has 21 dimensions and there are seven tasks. We randomly select 600 data points for each task to form the training set and 1400 data points for each task for the test set. The performance measure used is the normalized mean squared error (nMSE), which is the mean squared error divided by the variance of the ground truth. The single-task learning method is kernel ridge regression. The kernel used is the RBF kernel. Five-fold cross validation is used to determine the values of the kernel parameter and the regularization parameters λ_1 and λ_2. We perform 10 random splits of the data and report the mean and standard derivation over the 10 trials. The results are summarized in Table 3.1 and the mean task correlation matrix over 10 trials is recorded in Table 3.2. From the results, we can see that the performance of MTRL is better than that of STL, MTFL and MTGP. From Table 3.2, we can see that some tasks are positively correlated (e.g., third and sixth tasks), some are negatively correlated (e.g., second and third tasks), and some are uncorrelated (e.g., first and seventh tasks).

Table 3.1: Comparison of different methods on SARCOS data. Each column represents one task. The first row of each method records the mean of nMSE over 10 trials and the second row records the standard derivation.

Method	1st DOF	2nd DOF	3rd DOF	4th DOF	5th DOF	6th DOF	7th DOF
STL	0.2874	0.2356	0.2310	0.2366	0.0500	0.5208	0.6748
	0.0067	0.0043	0.0068	0.0042	0.0034	0.0205	0.0048
MTFL	0.2876	0.1611	0.2125	0.2215	0.0858	0.5224	0.7135
	0.0178	0.0105	0.0225	0.0151	0.0225	0.0269	0.0196
MTGP	0.3430	0.7890	0.5560	0.3147	0.0100	**0.0690**	0.6455
	0.1038	0.0480	0.0511	0.1235	0.0067	0.0171	0.4722
MTRL	**0.0968**	**0.0229**	**0.0625**	**0.0422**	**0.0045**	0.0851	**0.3450**
	0.0047	0.0023	0.0044	0.0027	0.0002	0.0095	0.0127

Table 3.2: Mean task correlation matrix learned from SARCOS data on different tasks.

	1st	2nd	3rd	4th	5th	6th	7th
1st	1.0000	0.7435	-0.7799	0.4819	-0.5325	-0.4981	0.0493
2nd	0.7435	1.0000	-0.9771	0.1148	-0.0941	-0.7772	-0.4419
3rd	-0.7799	-0.9771	1.0000	-0.1872	0.1364	0.8145	0.3987
4th	0.4819	0.1148	-0.1872	1.0000	-0.1889	-0.3768	0.7662
5th	-0.5325	-0.0941	0.1364	-0.1889	1.0000	-0.3243	-0.2834
6th	-0.4981	-0.7772	0.8145	-0.3768	-0.3243	1.0000	0.2282
7th	0.0493	-0.4419	0.3987	0.7662	-0.2834	0.2282	1.0000

Moreover, we plot in Figure 3.2 the change in value of the objective function in problem (3.8). We find that the objective function value decreases rapidly and then levels off, showing the fast convergence of the algorithm which takes no more than 15 iterations.

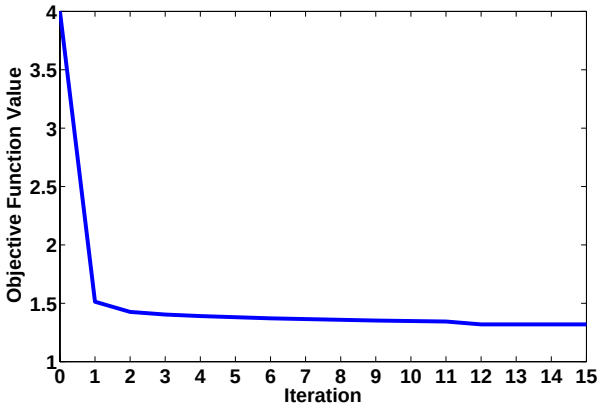


Figure 3.2: Convergence of objective function value for SARCOS data.

3.4.3 Multi-Domain Sentiment Application

We next study a multi-domain sentiment classification application[6] which is a multi-task classification problem. Its goal is to classify the reviews of some products into two classes: positive and negative reviews. In the data set, there are four different products (tasks) from Amazon.com: books, DVDs, electronics, and kitchen appliances. For each task, there are 1000 positive and 1000 negative data points corresponding to positive and negative reviews, respectively. Each data point has 473856 feature dimensions. To see the effect of varying the training set size, we randomly select 10%, 30% and 50% of the data for each task to form the training set and the rest for the test set. The performance measure used is the classification error. We use SVM as the single-task learning method. The kernel used is the linear kernel which is widely used for text applications with high feature dimensionality. Five-fold cross validation is used to determine the values of the regularization parameters λ_1 and λ_2. We perform 10 random splits of the data and report the mean and standard derivation over the 10 trials. The results are summarized in the left column of Table 3.3. From the table, we can see that the performance of MTRL is better than that of STL, MTFL and MTGP on every task under different training set sizes. Moreover, we can see that when the training size increases, the performance of our method MTRL becomes better on each task.

The mean task correlation matrices over 10 trials for different training set sizes are recorded in the right column of Table 3.3. From Table 3.3, we can see that the first task 'books' is more correlated with the second task 'DVDs' than with the other tasks; the third and fourth tasks achieve the largest correlation among all pairs of tasks. The findings from Table 3.3 can be easily interpreted as follows: 'books' and 'DVDs' are mainly for entertainment; almost all the elements in 'kitchen appliances' belong to 'electronics'. So the knowledge found by our method about the relationships between tasks matches our intuition. Moreover, some interesting patterns exist in the mean task correlation matrices for different training set sizes. For example,

[6] http://www.cs.jhu.edu/∼mdredze/datasets/sentiment/

the correlation between the third and fourth tasks is always the largest when training size varies; the correlation between the first and second tasks is larger than that between the first and third tasks, and also between the first and fourth tasks.

Table 3.3: Comparison of different methods on multi-domain sentiment data for different training set sizes. The three tables in the left column record the classification errors of different methods when 10%, 30% and 50%, respectively, of the data are used for training. Each column in a table represents one task. For each method, the first row records the mean classification error over 10 trials and the second row records the standard derivation. The three tables in the right column record the mean task correlation matrices learned on different tasks for different training set sizes (10%, 30% and 50% of the data). 1st task: books; 2nd task: DVDs; 3rd task: electronics; 4th task: kitchen appliances.

Method	1st Task	2nd Task	3rd Task	4th Task
STL	0.2680	0.3142	0.2891	0.2401
	0.0112	0.0110	0.0113	0.0154
MTFL	0.2667	0.3071	0.2880	0.2407
	0.0160	0.0136	0.0193	0.0160
MTGP	0.2332	0.2739	0.2624	0.2061
	0.0159	0.0231	0.0150	0.0152
MTRL	**0.2233**	**0.2564**	**0.2472**	**0.2027**
	0.0055	0.0050	0.0082	0.0044

	1st	2nd	3rd	4th
1st	1.0000	0.7675	0.6878	0.6993
2nd	0.7675	1.0000	0.6937	0.6805
3rd	0.6878	0.6937	1.0000	0.8793
4th	0.6993	0.6805	0.8793	1.0000

Method	1st Task	2nd Task	3rd Task	4th Task
STL	0.1946	0.2333	0.2143	0.1795
	0.0102	0.0119	0.0110	0.0076
MTFL	0.1932	0.2321	0.2089	0.1821
	0.0094	0.0115	0.0054	0.0078
MTGP	0.1852	0.2155	0.2088	0.1695
	0.0109	0.0101	0.0120	0.0074
MTRL	**0.1688**	**0.1987**	**0.1975**	**0.1482**
	0.0103	0.0120	0.0094	0.0087

	1st	2nd	3rd	4th
1st	1.0000	0.6275	0.5098	0.5936
2nd	0.6275	1.0000	0.4900	0.5345
3rd	0.5098	0.4900	1.0000	0.7286
4th	0.5936	0.5345	0.7286	1.0000

Method	1st Task	2nd Task	3rd Task	4th Task
STL	0.1854	0.2162	0.2072	0.1706
	0.0102	0.0147	0.0133	0.0024
MTFL	0.1821	0.2096	0.2128	0.1681
	0.0095	0.0095	0.0106	0.0085
MTGP	0.1722	0.2040	0.1992	0.1496
	0.0101	0.0152	0.0083	0.0051
MTRL	**0.1538**	**0.1874**	**0.1796**	**0.1334**
	0.0096	0.0149	0.0084	0.0036

	1st	2nd	3rd	4th
1st	1.0000	0.6252	0.5075	0.5901
2nd	0.6252	1.0000	0.4891	0.5328
3rd	0.5075	0.4891	1.0000	0.7256
4th	0.5901	0.5328	0.7256	1.0000

3.4.4 Examination Score Prediction

The school data set[7] has been widely used for studying multi-task regression. It consists of the examination scores of 15362 students from 139 secondary schools in London during the years 1985, 1986 and 1987. Thus, there are totally 139 tasks. The input consists of the year of the examination, four school-specific and three student-specific attributes. We replace each categorical attribute with one binary variable for each possible attribute value, as in (Evgeniou et al., 2005). As a result of this preprocessing, we have a total of 27 input attributes. The experimental settings are the same as those in (Argyriou et al., 2008a), i.e., we use the same 10 random splits of the data to generate the training and test sets, so that 75% of the examples from

[7] http://www.cs.ucl.ac.uk/staff/A.Argyriou/code/

each school belong to the training set and 25% to the test set. For our performance measure, we use the measure of percentage explained variance from (Argyriou et al., 2008a), which is defined as the percentage of one minus nMSE. We use five-fold cross validation to determine the values of the kernel parameter and the regularization parameters λ_1 and λ_2. Since the experimental setting is the same, we compare our result with the results reported in (Argyriou et al., 2008a; Bonilla et al., 2007). The results are summarized in Table 3.4. We can see that the performance of MTRL is better than both STL and MTFL and is slightly better than MTGP.

Table 3.4: Comparison of different methods on school data.

Method	Explained Variance
STL	23.5±1.9%
MTFL	26.7±2.0%
MTGP	29.2±1.6%
MTRL	**29.9±1.8%**

3.4.5 Experiments on Asymmetric Multi-Task Learning

The above sections mainly focus on symmetric multi-task learning. Here in this section we report some experiments on asymmetric multi-task learning. To the best of our knowledge, (Xue et al., 2007) is the only previous work reporting results on asymmetric multi-task learning. Since the DP-MTL method in (Xue et al., 2007) focuses on classification applications, we compare it with our method on the multi-domain sentiment application. Moreover, we also make comparison with conventional SVM which serves as a baseline single-task learning method.

All compared methods are tested under the leave-one-task-out (LOTO) setting. That is, in each fold, one task is treated as the new task while all other tasks are treated as existing tasks. Moreover, to see the effect of varying the training set size, we randomly sample 10%, 30% or 50% of the data in the new task to form the training set and the rest is used as the test set. Each configuration is repeated 10 times and we record the mean and standard deviation of the classification error in the experimental results. The results are recorded in Table 3.5. We can see that our method outperforms both single-task learning and DP-MTL. In fact the performance of DP-MTL is even worse than that of single-task learning. One reason is that the relationships between tasks do not exhibit strong cluster structure, as can be revealed from the task correlation matrix in Table 3.3. Since the tasks have no cluster structure, merging two tasks into one and learning common model parameters for the merged tasks will likely deteriorate the performance. Moreover, we can see that the more training data, the better the performance of our method on each task.

Table 3.5: Classification errors (in mean±std-dev) of different methods on the multi-domain sentiment data for different training set sizes under the asymmetric multi-task setting. The three tables record the classification errors of different methods when 10%, 30% and 50%, respectively, of the data are used for training. Each row in a table shows the results of the three methods when one task is chosen to be the new task. 'STL' represents conventional SVM trained on the training set of the new task only. 1st task: books; 2nd task: DVDs; 3rd task: electronics; 4th task: kitchen appliances.

New Task	STL	DP-MTL	MTRL
1st Task	0.3013±0.0265	0.3483±0.0297	**0.2781±0.0170**
2nd Task	0.3073±0.0117	0.3349±0.0121	**0.2801±0.0293**
3rd Task	0.2672±0.0267	0.2936±0.0274	**0.2451±0.0078**
4th Task	0.2340±0.0144	0.2537±0.0128	**0.2114±0.0208**

New Task	STL	DP-MTL	MTRL
1st Task	0.2434±0.0097	0.2719±0.0212	**0.2164±0.0098**
2nd Task	0.2479±0.0101	0.2810±0.0253	**0.2120±0.0160**
3rd Task	0.2050±0.0172	0.2306±0.0131	**0.1883±0.0106**
4th Task	0.1799±0.0057	0.2141±0.0362	**0.1561±0.0123**

New Task	STL	DP-MTL	MTRL
1st Task	0.2122±0.0083	0.2576±0.0152	**0.1826±0.0156**
2nd Task	0.2002±0.0112	0.2582±0.0275	**0.1870±0.0151**
3rd Task	0.1944±0.0069	0.2252±0.0208	**0.1692±0.0107**
4th Task	0.1678±0.0109	0.1910±0.0227	**0.1398±0.0131**

3.5 Application to Transfer Metric Learning

Many data mining algorithms, for example, k-means clustering algorithm and k-nearest neighbor classifier, work by relying on a distance metric. In order to deliver satisfactory results, finding a good distance metric for the problem at hand often plays a very crucial role. As such, metric learning (Xing et al., 2002) has received much attention in the research community (Baxter, 1997; Xing et al., 2002; Chang & Yeung, 2004; Weinberger et al., 2005; Davis et al., 2007; Yeung & Chang, 2007; Chen et al., 2007; Davis & Dhillon, 2008; Yeung et al., 2008; Zhan et al., 2009; Jin et al., 2009). Many metric learning methods have been proposed. From the perspective of the underlying learning paradigm, these methods can be grouped into three categories, namely, supervised metric learning, unsupervised metric learning, and semi-supervised metric learning. Supervised metric learning learns a metric for some supervised learning tasks, such as classification, so that data points from the same class are kept close while those from different classes remain far apart (Baxter, 1997; Weinberger et al., 2005; Davis et al., 2007; Davis & Dhillon, 2008; Zhan et al., 2009; Jin et al., 2009). It has also been used for regression by exploiting the manifold structure contained in the labeled data (Xiao et al., 2009). Unsupervised metric learning utilizes some information contained in the data to learn a metric for some unsupervised learning task, such as clustering (Chen et al., 2007). Semi-supervised metric learning,

which can be viewed as a combination of the supervised and unsupervised paradigms, utilizes both label information from the labeled data and geometric information from the unlabeled data to learn a good metric for classification or clustering. The need for semi-supervised metric learning arises from the fact that the labeled data available in a number of real-life applications is scarce because labeling data is very laborious and costly. With only limited labeled data, the metrics learned are often unsatisfactory. Semi-supervised metric learning tries to exploit additional information from the unlabeled data to alleviate this problem.

The focus of this work is on supervised metric learning for classification applications. However, we consider situations similar to those for semi-supervised metric learning in which there is deficiency in labeled data. While the amount of labeled data available in one learning task is limited, it is not uncommon that there exist other related learning tasks with labeled data available. Unlike semi-supervised learning which exploits unlabeled data, multi-task learning (Caruana, 1997; Baxter, 1997; Thrun, 1995) and transfer learning (Pan & Yang, 2010) seek to alleviate the labeled data deficiency problem by utilizing some related learning tasks to help improve the learning performance. In some sense, they mimic human learning activities in that people may learn faster when several related tasks are learned simultaneously, for example, playing different games. In essence, people often apply the knowledge gained from some previous learning tasks to help learn a new task. Even though both multi-task learning and transfer learning utilize information from other related learning tasks, there exist some differences between them in both the problem setting and the objective. In transfer learning, the learning tasks are usually classified into two types: source task and target task. It is assumed that there is enough data in the source tasks but not in the target task. The objective of transfer learning is to utilize the information in the source tasks to help learn the target task with no need for improving the performance of the source tasks. On the other hand, there is no distinction between the tasks in multi-task learning and the objective is to improve the performance of all tasks simultaneously.

Even though there exist differences between multi-task learning and transfer learning, a central issue common to both is to accurately characterize the relationships between tasks. Similar to MTRL, it is a better way to learn the task relationships in the form of the task covariance matrix from data automatically. In this work, we study metric learning under the transfer learning setting in which some source tasks are available in addition to the target task.

3.5.1 Multi-Task Metric Learning

In this section, we propose a multi-task metric learning method which can learn the task relationships between all pairs of tasks.

Suppose we are given m learning tasks $\{T_i\}_{i=1}^m$. For the ith task T_i, the training set $\mathcal{D}_i$

consists of n_i data points represented in the form of $(\mathbf{x}_j^i, y_j^i)$, $j = 1, \ldots, n_i$, with $\mathbf{x}_j^i \in \mathbb{R}^d$ and its corresponding class label $y_j^i \in \{1, \ldots, C_i\}$. Here the superscript denotes the task index and the subscript denotes the instance index in each task.

The optimization problem for multi-task metric learning is formulated as follows:

$$\begin{aligned}
\min_{\{\mathbf{\Sigma}_i\},\mathbf{\Omega}} \quad & \sum_{i=1}^{m} \frac{2}{n_i(n_i-1)} \sum_{j<k} g\Big(y_{j,k}^i \left[1 - \|\mathbf{x}_j^i - \mathbf{x}_k^i\|_{\mathbf{\Sigma}_i}^2\right]\Big) + \frac{\lambda_1}{2} \sum_{i=1}^{m} \|\mathbf{\Sigma}_i\|_F^2 + \frac{\lambda_2}{2} \mathrm{tr}(\tilde{\mathbf{\Sigma}} \mathbf{\Omega}^{-1} \tilde{\mathbf{\Sigma}}^T) \\
\text{s.t.} \quad & \mathbf{\Sigma}_i \succeq \mathbf{0} \; \forall i \\
& \tilde{\mathbf{\Sigma}} = (\mathrm{vec}(\mathbf{\Sigma}_1), \ldots, \mathrm{vec}(\mathbf{\Sigma}_m)) \\
& \mathbf{\Omega} \succeq 0 \\
& \mathrm{tr}(\mathbf{\Omega}) = 1,
\end{aligned} \tag{3.28}$$

where $y_{j,k}^i$ is equal to 1 when $y_j^i = y_k^i$ and -1 otherwise, $\mathrm{vec}(\cdot)$ denotes the operator which converts a matrix into a vector in a columnwise manner, and λ_1 and λ_2 are the regularization parameters. $\mathbf{\Omega}$ is a task covariance matrix which describes the relationships between tasks and so it is a PSD matrix. The first term in the objective function of problem (3.28) measures the empirical loss for the training sets of the m tasks, the second term penalizes the complexity of each $\mathbf{\Sigma}_i$, and the last term measures the task relationships between all pairs of tasks based on each $\mathbf{\Sigma}_i$. The last constraint in (3.28) is to restrict the scale of $\mathbf{\Omega}$ to prevent it from reaching a degenerate solution.

From a probabilistic viewpoint, RDML can be seen as obtaining the MAP solution of a probabilistic model where the likelihood corresponds to the loss in RDML and the prior on the metric is Gaussian prior corresponding to the second term. Similar to RDML, our multi-task metric learning is also a MAP solution of a probabilistic model where the likelihood is the same as that in RDML for each task and the prior on the metrics of all tasks is matrix variate normal distribution (Gupta & Nagar, 2000).

We will prove below that problem (3.28) is a convex optimization problem by proving that each term in the objective function is convex and each constraint is also convex.

Theorem 3.2 *Problem (3.28) is convex with respect to* $\{\mathbf{\Sigma}_i\}$ *and* $\mathbf{\Omega}$.

Proof

It is easy to see that the first two terms in the objective function are convex with respect to (w.r.t.) all variables and the constraints in (3.28) are also convex. We rewrite the third term as

$$\mathrm{tr}(\tilde{\mathbf{\Sigma}} \mathbf{\Omega}^{-1} \tilde{\mathbf{\Sigma}}^T) = \sum_t \tilde{\mathbf{\Sigma}}(t,:) \mathbf{\Omega}^{-1} \tilde{\mathbf{\Sigma}}(t,:)^T,$$

where $\tilde{\mathbf{\Sigma}}(t,:)$ is the tth row of $\tilde{\mathbf{\Sigma}}$. Since $\tilde{\mathbf{\Sigma}}(t,:) \mathbf{\Omega}^{-1} \tilde{\mathbf{\Sigma}}(t,:)^T$ is a matrix fractional function as in Example 3.4 on page 76 of (Boyd & Vandenberghe, 2004), it is convex w.r.t. $\tilde{\mathbf{\Sigma}}(t,:)$ and $\mathbf{\Omega}$ when

$\mathbf{\Omega}$ is a PSD matrix (which is satisfied by the first constraint of (3.28)). Since $\tilde{\mathbf{\Sigma}}(t,:)$ is a row of $\tilde{\mathbf{\Sigma}}$, $\tilde{\mathbf{\Sigma}}(t,:)\mathbf{\Omega}^{-1}\tilde{\mathbf{\Sigma}}(t,:)^T$ is also convex w.r.t. $\{\mathbf{\Sigma}_i\}$ and $\mathbf{\Omega}$. Because the summation operation can preserve convexity according to the analysis on page 79 of (Boyd & Vandenberghe, 2004), $\text{tr}(\tilde{\mathbf{\Sigma}}\mathbf{\Omega}^{-1}\tilde{\mathbf{\Sigma}}^T) = \sum_t \tilde{\mathbf{\Sigma}}(t,:)\mathbf{\Omega}^{-1}\tilde{\mathbf{\Sigma}}(t,:)^T$ is convex w.r.t. $\{\mathbf{\Sigma}_i\}$ and $\mathbf{\Omega}$. So the objective function and the constraints in problem (3.28) are convex w.r.t. all variables and hence problem (3.28) is jointly convex. □

Even though problem (3.28) is convex with respect to $\{\mathbf{\Sigma}_i\}$ and $\mathbf{\Omega}$ jointly, it is not easy to optimize it with respect to all the variables simultaneously. Here we propose an alternating method to solve the problem more efficiently. Specifically, we first optimize the objective function with respect to $\mathbf{\Sigma}_i$ when $\mathbf{\Omega}$ and $\{\mathbf{\Sigma}\}_{-i} \stackrel{\text{def}}{=} \{\mathbf{\Sigma}_1, \ldots, \mathbf{\Sigma}_{i-1}, \mathbf{\Sigma}_{i+1}, \ldots, \mathbf{\Sigma}_m\}$ are fixed, and then optimize it with respect to $\mathbf{\Omega}$ when $\{\mathbf{\Sigma}_i\}$ are fixed. This procedure is repeated until convergence.

Because multi-task metric learning is not the focus of this work, we leave the detailed optimization procedure to Appendix A.4.

3.5.2 Transfer Metric Learning

Based on the multi-task metric learning problem formulated in the previous section, we propose a transfer metric learning formulation as a special case which can learn the task relationships between all source tasks and the target task.

Suppose we are given $m-1$ source tasks $\{T_i\}_{i=1}^{m-1}$ and one target task T_m, for $m > 1$. In the target task, the training set contains n_m labeled data points $\{(\mathbf{x}_j^m, y_j^m)\}_{j=1}^{n_m}$. In transfer learning, it is assumed that each source task has enough labeled data and can learn an accurate model with no need to seek help from the other source tasks. So the source tasks are considered to be independent since each source task does not need help from other source tasks. So, similar to the setting in (Zha et al., 2009), we assume that the metric matrix $\mathbf{\Sigma}_i$ for the ith source task has been learned independently. We hope to use the metric matrices learned to help the learning of the target task because the labeled data there is scarce.

Optimization Problem

Based on problem (3.28), we formulate the optimization problem for TML as follows:

$$\begin{aligned}
\min_{\mathbf{\Sigma}_m, \mathbf{\Omega}} \quad & \frac{2}{n_m(n_m-1)} \sum_{j<k} g\Big(y_{j,k}^m \left[1 - \|\mathbf{x}_j^m - \mathbf{x}_k^m\|_{\mathbf{\Sigma}_m}^2\right]\Big) + \frac{\lambda_1}{2}\|\mathbf{\Sigma}_m\|_F^2 + \frac{\lambda_2}{2}\text{tr}(\tilde{\mathbf{\Sigma}}\mathbf{\Omega}^{-1}\tilde{\mathbf{\Sigma}}^T) \\
\text{s.t.} \quad & \mathbf{\Sigma}_m \succeq \mathbf{0} \\
& \tilde{\mathbf{\Sigma}} = (\text{vec}(\mathbf{\Sigma}_1), \ldots, \text{vec}(\mathbf{\Sigma}_{m-1}), \text{vec}(\mathbf{\Sigma}_m)) \\
& \mathbf{\Omega} \succeq 0 \\
& \text{tr}(\mathbf{\Omega}) = 1.
\end{aligned} \tag{3.29}$$

Since we assume that the source tasks are independent and each source is of equal importance, we can express $\mathbf{\Omega}$ as

$$\mathbf{\Omega} = \begin{pmatrix} \alpha \mathbf{I}_{m-1} & \boldsymbol{\omega}_m \\ \boldsymbol{\omega}_m^T & \omega \end{pmatrix},$$

where $\mathbf{I}_a$ denotes the $a \times a$ identity matrix, $\boldsymbol{\omega}_m$ denotes the task covariances between the target task and the source tasks, and ω denotes the variance of the target task. According to the last constraint in problem (3.29), we can get

$$\alpha = \frac{1-\omega}{m-1}.$$

From Theorem 3.2, it is easy to show that problem (3.29) is also jointly convex with respect to all variables. Moreover, from the block matrix inversion formula, we can get

$$\begin{aligned}
\mathbf{\Omega}^{-1} &= \begin{pmatrix} \frac{1-\omega}{m-1}\mathbf{I}_{m-1} & \boldsymbol{\omega}_m \\ \boldsymbol{\omega}_m^T & \omega \end{pmatrix}^{-1} \\
&= \begin{pmatrix} \mathbf{I}_{m-1} & \mathbf{a} \\ \mathbf{0}_{m-1}^T & 1 \end{pmatrix} \begin{pmatrix} \frac{(m-1)\mathbf{I}_{m-1}}{1-\omega} & \mathbf{0}_{m-1} \\ \mathbf{0}_{m-1}^T & \frac{1}{c} \end{pmatrix} \begin{pmatrix} \mathbf{I}_{m-1} & \mathbf{0}_{m-1} \\ \mathbf{a}^T & 1 \end{pmatrix},
\end{aligned}$$

where $\mathbf{a} = -\frac{(m-1)\boldsymbol{\omega}_m}{1-\omega}$ and $c = \omega - \frac{(m-1)\boldsymbol{\omega}_m^T\boldsymbol{\omega}_m}{1-\omega}$.

Let $\tilde{\mathbf{\Sigma}}_s = (\text{vec}(\mathbf{\Sigma}_1), \ldots, \text{vec}(\mathbf{\Sigma}_{m-1}))$, which is a constant matrix here, denote the parameter

matrix of the source tasks. Then we can get

$$
\begin{aligned}
&\mathrm{tr}(\tilde{\mathbf{\Sigma}}\mathbf{\Omega}^{-1}\tilde{\mathbf{\Sigma}}^T) \\
=&\mathrm{tr}\Big(\Big(\tilde{\mathbf{\Sigma}}_s, \mathrm{vec}(\mathbf{\Sigma}_m)\Big)\mathbf{\Omega}^{-1}\begin{pmatrix}\tilde{\mathbf{\Sigma}}_s^T \\ \mathrm{vec}(\mathbf{\Sigma}_m)^T\end{pmatrix}\Big) \\
=&\mathrm{tr}\Big(\begin{pmatrix}\tilde{\mathbf{\Sigma}}_s^T \\ \mathrm{vec}(\mathbf{\Sigma}_m)^T - \frac{(m-1)}{1-\omega}\boldsymbol{\omega}_m^T\tilde{\mathbf{\Sigma}}_s^T\end{pmatrix}^T\begin{pmatrix}\frac{(m-1)\mathbf{I}_{m-1}}{1-\omega} & \mathbf{0}_{m-1} \\ \mathbf{0}_{m-1}^T & \frac{1}{c}\end{pmatrix}^{-1}\begin{pmatrix}\tilde{\mathbf{\Sigma}}_s^T \\ \mathrm{vec}(\mathbf{\Sigma}_m)^T - \frac{(m-1)}{1-\omega}\boldsymbol{\omega}_m^T\tilde{\mathbf{\Sigma}}_s^T\end{pmatrix}\Big) \\
=&\frac{m-1}{1-\omega}\mathrm{tr}(\tilde{\mathbf{\Sigma}}_s^T\tilde{\mathbf{\Sigma}}_s) + \frac{1}{c}\|\mathrm{vec}(\mathbf{\Sigma}_m) - \frac{(m-1)}{1-\omega}\tilde{\mathbf{\Sigma}}_s\boldsymbol{\omega}_m\|_2^2 \\
=&\frac{(1-\omega)\|\mathbf{\Sigma}_m\|_F^2 - 2(m-1)\mathrm{vec}(\mathbf{\Sigma}_m)^T\tilde{\mathbf{\Sigma}}_s\boldsymbol{\omega}_m + (m-1)\omega\mathrm{tr}(\tilde{\mathbf{\Sigma}}_s^T\tilde{\mathbf{\Sigma}}_s)}{\omega(1-\omega) - (m-1)\boldsymbol{\omega}_m^T\boldsymbol{\omega}_m}.
\end{aligned} \tag{3.30}
$$

Moreover, according to the Schur complement (Boyd & Vandenberghe, 2004), we have

$$
\mathbf{\Omega} \succeq 0 \iff \omega \geq \frac{m-1}{1-\omega}\boldsymbol{\omega}_m^T\boldsymbol{\omega}_m \text{ and } \frac{(m-1)\mathbf{I}_{m-1}}{1-\omega} \succeq 0,
$$

which is equivalent to

$$
\mathbf{\Omega} \succeq 0 \iff \omega(1-\omega) \geq (m-1)\boldsymbol{\omega}_m^T\boldsymbol{\omega}_m.
$$

Then problem (3.29) can be simplified to

$$
\begin{aligned}
\min_{\mathbf{\Sigma}_m, \boldsymbol{\omega}_m, \omega, \mathbf{\Omega}} \quad & \frac{2}{n_m(n_m-1)}\sum_{j<k} g\Big(y_{j,k}^m\left[1 - \|\mathbf{x}_j^m - \mathbf{x}_k^m\|_{\mathbf{\Sigma}_m}^2\right]\Big) + \frac{\lambda_1}{2}\|\mathbf{\Sigma}_m\|_F^2 + \frac{\lambda_2}{2}\mathrm{tr}(\tilde{\mathbf{\Sigma}}\mathbf{\Omega}^{-1}\tilde{\mathbf{\Sigma}}^T) \\
\text{s.t.} \quad & \mathbf{\Sigma}_m \succeq \mathbf{0} \\
& \mathbf{\Omega} = \begin{pmatrix}\frac{1-\omega}{m-1}\mathbf{I}_{m-1} & \boldsymbol{\omega}_m \\ \boldsymbol{\omega}_m^T & \omega\end{pmatrix} \\
& \tilde{\mathbf{\Sigma}} = (\tilde{\mathbf{\Sigma}}_s, \mathrm{vec}(\mathbf{\Sigma}_m)) \\
& \omega(1-\omega) \geq (m-1)\boldsymbol{\omega}_m^T\boldsymbol{\omega}_m,
\end{aligned} \tag{3.31}
$$

where the last term in the objective function can be simplified as in Eq. (3.30).

Moreover, compared with problem (3.29), there is no PSD constraint on $\mathbf{\Omega}$ in problem (3.31) making it simpler than problem (3.29). In the next section, we will discuss how to solve problem (3.31).

Optimization Procedure

As in multi-task metric learning, problem (3.31) is a convex problem and we still use an alternating method to solve it. Specifically, we first optimize the objective function with respect to $\mathbf{\Sigma}_m$ when $\boldsymbol{\omega}_m$ and ω are fixed, and then optimize it with respect to $\boldsymbol{\omega}_m$ and ω when $\mathbf{\Sigma}_m$ is fixed. In what follows, we will present the two subproblems separately.

Table 3.6: Online learning algorithm for Problem (3.32)

Input: labeled data $(\mathbf{x}_j^m, y_j^m)$ $(j = 1, \ldots, n_m)$, matrix $\mathbf{M}$, λ_1', λ_2' and predefined learning rate η

Initialize $\boldsymbol{\Sigma}_m^{(0)} = \frac{\lambda_2'}{\lambda_1'}\mathbf{M}$;
for $t = 1, \ldots, T_{max}$ **do**
  Receive a pair of training data points $\{(\mathbf{x}_j^m, y_j^m), (\mathbf{x}_k^m, y_k^m)\}$;
  Compute y: $y = 1$ if $y_j^m = y_k^m$, and $y = -1$ otherwise;
  if the training pair $(\mathbf{x}_j^m, \mathbf{x}_k^m)$, y is classified correctly, i.e., $y(1 - \|\mathbf{x}_j^m - \mathbf{x}_k^m\|^2_{\boldsymbol{\Sigma}_m^{(t-1)}}) > 0$
    $\boldsymbol{\Sigma}_m^{(t)} = \boldsymbol{\Sigma}_m^{(t-1)}$;
  else if $y == -1$
    $\boldsymbol{\Sigma}_m^{(t)} = \boldsymbol{\Sigma}_m^{(t-1)} + \eta(\mathbf{x}_j^m - \mathbf{x}_k^m)(\mathbf{x}_j^m - \mathbf{x}_k^m)^T$;
  else
    $\boldsymbol{\Sigma}_m^{(t)} = \pi_{S_+}\left(\boldsymbol{\Sigma}_m^{(t-1)} - \eta(\mathbf{x}_j^m - \mathbf{x}_k^m)(\mathbf{x}_j^m - \mathbf{x}_k^m)^T\right)$ where $\pi_{S_+}(\mathbf{A})$ projects matrix $\mathbf{A}$ into the positive semidefinite cone;
  end if
end for

Output: metric $\boldsymbol{\Sigma}_m^{(T_{max})}$

Optimizing w.r.t. $\boldsymbol{\Sigma}_m$ when $\boldsymbol{\omega}_m$ and ω are fixed

Utilizing Eq. (3.30), the optimization problem with respect to $\boldsymbol{\Sigma}_m$ is formulated as

$$\begin{aligned} \min_{\boldsymbol{\Sigma}_m} \quad & \frac{2}{n_m(n_m-1)} \sum_{j<k} g\Big(y_{j,k}^m \left[1 - \|\mathbf{x}_j^m - \mathbf{x}_k^m\|^2_{\boldsymbol{\Sigma}_m}\right]\Big) + \frac{\lambda_1'}{2}\|\boldsymbol{\Sigma}_m\|_F^2 - \lambda_2' \mathrm{tr}(\boldsymbol{\Sigma}_m^T \mathbf{M}) \\ \text{s.t.} \quad & \boldsymbol{\Sigma}_m \succeq \mathbf{0}, \end{aligned} \tag{3.32}$$

where

$$\begin{aligned} \lambda_1' &= \lambda_1 + \frac{\lambda_2(1-\omega)}{\omega(1-\omega) - (m-1)\boldsymbol{\omega}_m^T\boldsymbol{\omega}_m}, \\ \lambda_2' &= \frac{\lambda_2(m-1)}{\omega(1-\omega) - (m-1)\boldsymbol{\omega}_m^T\boldsymbol{\omega}_m}, \end{aligned}$$

$\mathbf{M}$ is a matrix such that $\mathrm{vec}(\mathbf{M}) = \tilde{\boldsymbol{\Sigma}}_s\boldsymbol{\omega}_m$. It is easy to show that $\mathbf{M}$ is a combination of $\boldsymbol{\Sigma}_i$ $(i = 1, \ldots, m-1)$ as $\mathbf{M} = \sum_{i=1}^{m-1} \omega_{mi}\boldsymbol{\Sigma}_i$ where ω_{mi} is the ith element of $\boldsymbol{\omega}_m$.

Similar to (Jin et al., 2009), we can use an online learning method to solve problem (3.32) and the algorithm is depicted in Table 3.6. This algorithm is similar to that in (Jin et al., 2009) except the initial step for $\boldsymbol{\Sigma}_m^{(0)}$. In (Jin et al., 2009), the initial value for $\boldsymbol{\Sigma}_m^{(0)}$ is a zero matrix but here it is $\frac{\lambda_2'}{\lambda_1'}\mathbf{M}$. Note that $\mathbf{M}$ is a combination of the metrics learned from the source tasks where each combination weight is the task covariance between a source task and the target task. This agrees with our intuition that a positively correlated source task will have a large weight on the initial value for $\boldsymbol{\Sigma}_m$, an outlier task has negligle contribution and a negatively correlated task even has opposite effect.

Optimizing w.r.t. $\boldsymbol{\omega}_m$ and ω when $\boldsymbol{\Sigma}_m$ is fixed

Utilizing Eq. (3.30), the optimization problem with respect to $\boldsymbol{\omega}_m$ and ω is formulated as

$$\begin{aligned} \min_{\boldsymbol{\omega}_m,\omega,\boldsymbol{\Omega}} \quad & \text{tr}(\tilde{\boldsymbol{\Sigma}}\boldsymbol{\Omega}^{-1}\tilde{\boldsymbol{\Sigma}}^T) \\ \text{s.t.} \quad & \boldsymbol{\Omega} = \begin{pmatrix} \frac{1-\omega}{m-1}\mathbf{I}_{m-1} & \boldsymbol{\omega}_m \\ \boldsymbol{\omega}_m^T & \omega \end{pmatrix} \\ & \omega(1-\omega) \geq (m-1)\boldsymbol{\omega}_m^T\boldsymbol{\omega}_m. \end{aligned} \tag{3.33}$$

We impose a constraint as $\tilde{\boldsymbol{\Sigma}}\boldsymbol{\Omega}^{-1}\tilde{\boldsymbol{\Sigma}}^T \preceq \frac{1}{t}\mathbf{I}_{d^2}$ and the objective function becomes $\min \frac{1}{t}$. Using the Schur complement, we can get

$$\tilde{\boldsymbol{\Sigma}}\boldsymbol{\Omega}^{-1}\tilde{\boldsymbol{\Sigma}}^T \preceq \frac{1}{t}\mathbf{I}_{d^2} \Longleftrightarrow \begin{pmatrix} \boldsymbol{\Omega} & \tilde{\boldsymbol{\Sigma}}^T \\ \tilde{\boldsymbol{\Sigma}} & \frac{1}{t}\mathbf{I}_{d^2} \end{pmatrix} \succeq \mathbf{0}.$$

By using the Schur complement again, we get

$$\begin{pmatrix} \boldsymbol{\Omega} & \tilde{\boldsymbol{\Sigma}}^T \\ \tilde{\boldsymbol{\Sigma}} & \frac{1}{t}\mathbf{I}_{d^2} \end{pmatrix} \succeq \mathbf{0} \Longleftrightarrow \boldsymbol{\Omega} - t\tilde{\boldsymbol{\Sigma}}^T\tilde{\boldsymbol{\Sigma}} \succeq \mathbf{0}.$$

We write $\tilde{\boldsymbol{\Sigma}}^T\tilde{\boldsymbol{\Sigma}} = \begin{pmatrix} \boldsymbol{\Psi}_{11} & \boldsymbol{\Psi}_{12} \\ \boldsymbol{\Psi}_{12}^T & \Psi_{22} \end{pmatrix}$ where $\boldsymbol{\Psi}_{11} \in \mathbb{R}^{(m-1)\times(m-1)}$, $\boldsymbol{\Psi}_{12} \in \mathbb{R}^{(m-1)\times 1}$ and $\Psi_{22} \in \mathbb{R}$. Then $\boldsymbol{\Omega} - t\tilde{\boldsymbol{\Sigma}}^T\tilde{\boldsymbol{\Sigma}} \succeq \mathbf{0}$ is equivalent to

$$\begin{aligned} & \frac{1-\omega}{m-1}\mathbf{I}_{m-1} - t\boldsymbol{\Psi}_{11} \succeq \mathbf{0} \\ & \omega - t\Psi_{22} \geq (\boldsymbol{\omega}_m - t\boldsymbol{\Psi}_{12})^T \left(\frac{1-\omega}{m-1}\mathbf{I}_{m-1} - t\boldsymbol{\Psi}_{11}\right)^{-1} (\boldsymbol{\omega}_m - t\boldsymbol{\Psi}_{12}). \end{aligned}$$

Let $\mathbf{U}$ and $\lambda_1, \ldots, \lambda_{m-1}$ denote the eigenvector matrix and eigenvalues of $\boldsymbol{\Psi}_{11}$ with $\lambda_1 \geq \ldots \geq \lambda_{m-1} \geq 0$. Then

$$\frac{1-\omega}{m-1}\mathbf{I}_{m-1} - t\boldsymbol{\Psi}_{11} \succeq \mathbf{0} \Longleftrightarrow \frac{1-\omega}{m-1} \geq \lambda_1 t$$

and

$$\left(\frac{1-\omega}{m-1}\mathbf{I}_{m-1} - t\boldsymbol{\Psi}_{11}\right)^{-1} = \mathbf{U}\,\text{diag}\left(\frac{1-\omega}{m-1} - t\lambda_1, \ldots, \frac{1-\omega}{m-1} - t\lambda_{m-1}\right)\mathbf{U}^T.$$

Combining the above results, problem (3.33) is formulated as

$$\begin{aligned}
\min_{\boldsymbol{\omega}_m,\omega,\mathbf{f},t} \quad & -t \\
\text{s.t.} \quad & \frac{1-\omega}{m-1} \geq t\lambda_1 \\
& \mathbf{f} = \mathbf{U}^T(\boldsymbol{\omega}_m - t\boldsymbol{\Psi}_{12}) \\
& \sum_{j=1}^{m-1} \frac{f_j^2}{\frac{1-\omega}{m-1} - t\lambda_j} \leq \omega - t\Psi_{22} \\
& \omega(1-\omega) \geq (m-1)\boldsymbol{\omega}_m^T\boldsymbol{\omega}_m,
\end{aligned} \tag{3.34}$$

where f_j is the jth element of $\mathbf{f}$. By introducing new variables h_j and r_j $(j = 1, \ldots, m-1)$, (3.34) is reformulated as

$$\begin{aligned}
\min_{\boldsymbol{\omega}_m,\omega,\mathbf{f},t,\{h_j\},\{r_j\}} \quad & -t \\
\text{s.t.} \quad & \frac{1-\omega}{m-1} \geq t\lambda_1 \\
& \mathbf{f} = \mathbf{U}^T(\boldsymbol{\omega}_m - t\boldsymbol{\Psi}_{12}) \\
& \sum_{j=1}^{m-1} h_j \leq \omega - t\Psi_{22} \\
& r_j = \frac{1-\omega}{m-1} - t\lambda_j \; \forall j \\
& \frac{f_j^2}{r_j} \leq h_j \; \forall j \\
& \omega(1-\omega) \geq (m-1)\boldsymbol{\omega}_m^T\boldsymbol{\omega}_m.
\end{aligned} \tag{3.35}$$

Since

$$\frac{f_j^2}{r_j} \leq h_j \iff \left\| \begin{pmatrix} f_j \\ \frac{r_j - h_j}{2} \end{pmatrix} \right\|_2 \leq \frac{r_j + h_j}{2}$$

and

$$\omega(1-\omega) \geq (m-1)\boldsymbol{\omega}_m^T\boldsymbol{\omega}_m \iff \left\| \begin{pmatrix} \sqrt{m-1}\boldsymbol{\omega}_m \\ \frac{\omega-1}{2} \\ \omega \end{pmatrix} \right\|_2 \leq \frac{\omega+1}{2},$$

problem (A.9) is a second-order cone programming (SOCP) problem (Lobo et al., 1998) with $O(m)$ variables and $O(m)$ constraints. In many applications, m is very small and we can use a standard solver to solve problem (A.9) very efficiently.

We set the initial value of ω to $\frac{1}{m}$ and that of $\boldsymbol{\omega}_m$ to a zero vector which corresponds to the assumption that the target task is unrelated to the source tasks.

After learning the optimal values of $\mathbf{\Sigma}_m$, we can make prediction for a new data point. Given a test data point $\mathbf{x}_\star^m$ for the target task T_m, we first calculate the distances between $\mathbf{x}_\star^m$ and all training data points in T_m based on the learned metric $\mathbf{\Sigma}_m$ and then use the k-nearest neighbor classifier to classify $\mathbf{x}_\star^m$, where we choose $k = 1$ for simplicity in our experiments.

3.5.3 Experiments

We study TML empirically in this section by comparing it with two metric learning methods, ITML[8] (Davis et al., 2007) and RDML (Jin et al., 2009), and another metric learning method for transfer learning, L-DML (Zha et al., 2009). We use the CVX solver (Grant & Boyd, 2008)[9] to solve problem (A.9). We set the learning rate η in Table 3.6 to 0.01. For ITML, RDML and L-DML, the best parameters reported in (Davis et al., 2007; Jin et al., 2009; Zha et al., 2009) are used.

Wine Quality Classification

The wine dataset[10] is about wine quality including red and white wine samples. The features include objective tests (e.g., PH values) and the output is based on sensory data. The labels are given by experts with grades between 0 (very bad) and 10 (very excellent). There are 1599 records for the red wine and 4898 for the white wine and so there are two tasks, one for red wine classification and the other for white wine classification. Each task is treated as the target task and the other task as the source task. To see the effect of varying the size of the training set, we vary the percentage of the training data used from 5% to 20%. Each configuration is repeated 10 times. The mean and standard deviation of the classification accuracy are reported in Fig. 3.3(a) and 3.3(b). From the results, we can see that the performance of L-DML is comparable with that of ITML and RDML and TML is always the best one for both tasks.

Handwritten Letter Classification

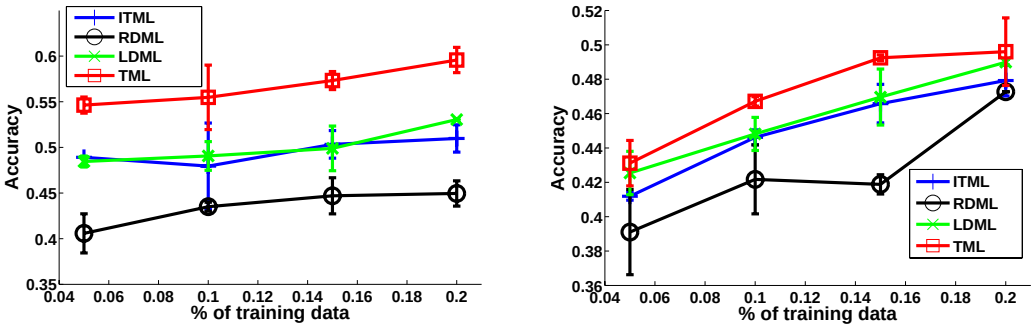
The handwritten letter classification applicaton[11] consists of seven tasks where each task is a binary classification problem. The corresponding letters for each task are: c/e, g/y, m/n, a/g, a/o, f/t and h/n. Each data point has 128 features corresponding to the pixel values of the handwritten letter images. For each task, there are about 1000 positive and 1000 negative data points. The experimental settings are the same as those for wine quality classification above. The results

[8]The implementation of ITML can be found in http://www.cs.utexas.edu/users/pjain/itml/.

[9]http://stanford.edu/∼boyd/cvx

[10]http://archive.ics.uci.edu/ml/datasets/Wine+Quality

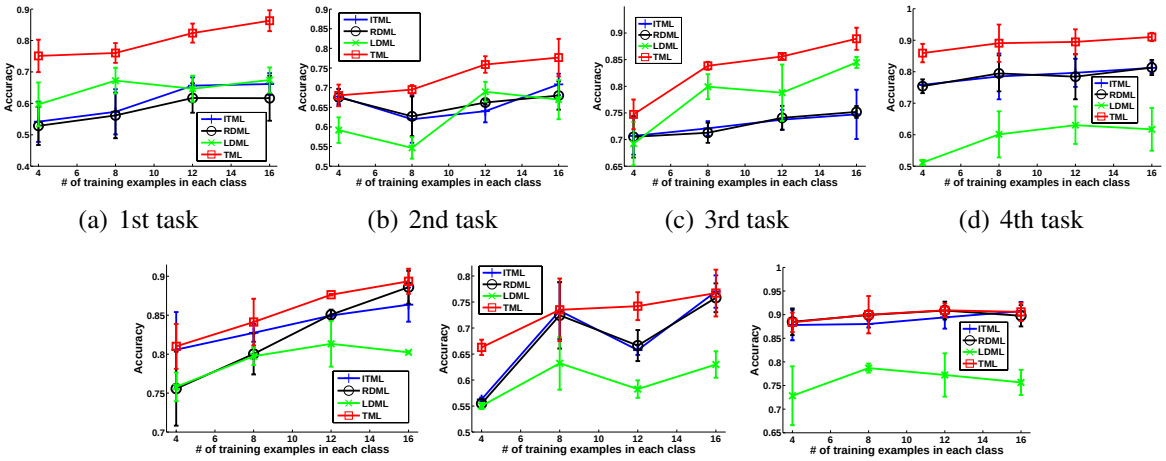
[11]http://multitask.cs.berkeley.edu/



(a) 1st task (b) 2nd task

Figure 3.3: Overall performance on wine quality classification application.

are plotted in Fig. 3.4(a) to 3.4(g). From the results, we find that the performance of L-MDL is worse than that of ITML and RDML on some tasks (4th, 6th and 7th tasks). This may be due to the fact that the objective function of L-MDL is non-convex and hence it is easy to get trapped in bad local minima. TML shows the best performance on almost every task.



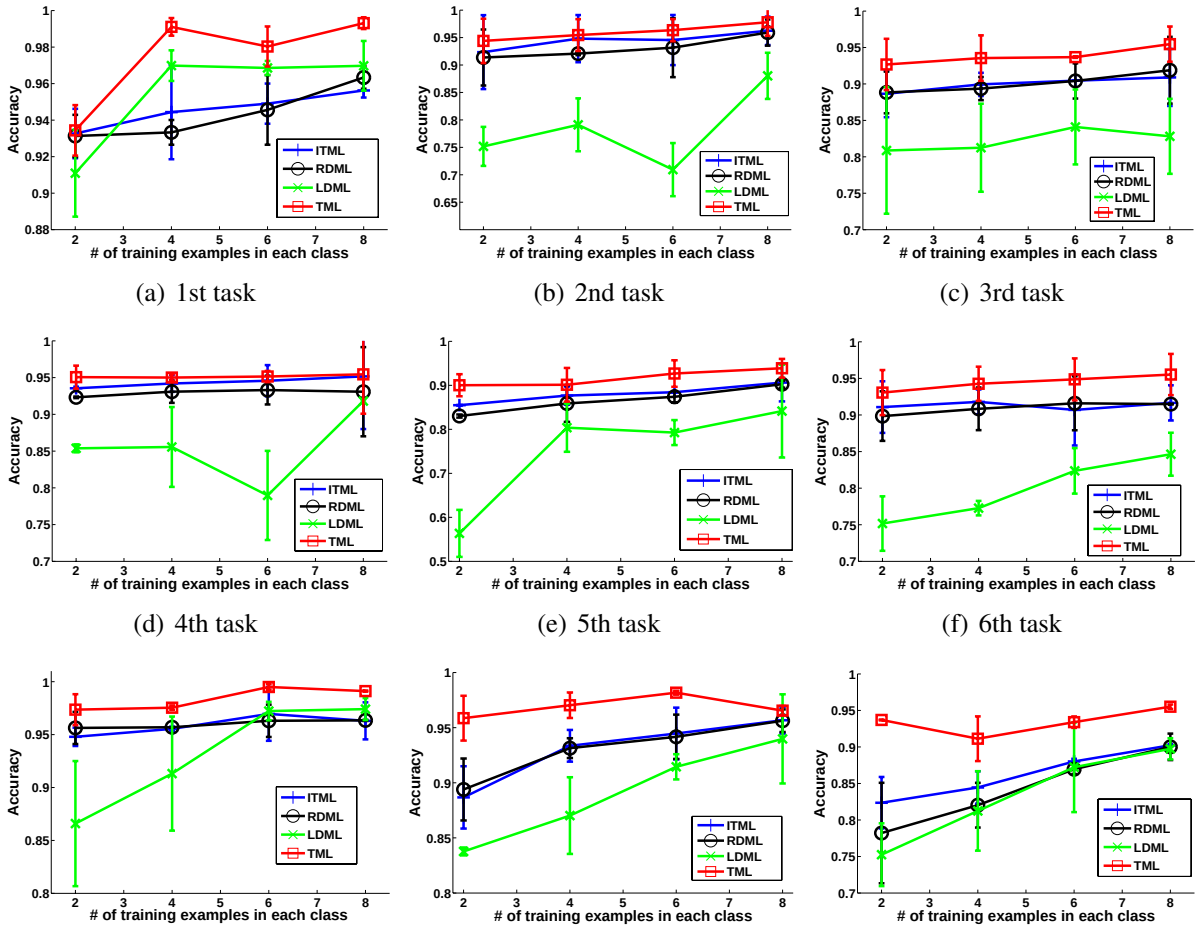
(a) 1st task (b) 2nd task (c) 3rd task (d) 4th task

(e) 5th task (f) 6th task (g) 7th task

Figure 3.4: Overall performance on handwritten letter classification application when one task is the target and the others are source tasks.

USPS Digit Classification

The USPS digit dataset[11] contains 7291 examples each of 255 features. There are nine classification tasks, each corresponding to the classification of two digits. The experimental settings are the same as those for handwritten letter classification. The results are reported in Fig. 3.5(a) to 3.5(i). Similar to handwritten digit classification, L-MDL is worse than ITML and RDML on some tasks and TML is better than other methods on almost all tasks.



(a) 1st task (b) 2nd task (c) 3rd task

(d) 4th task (e) 5th task (f) 6th task

(g) 7th task (h) 8th task (i) 9th task

Figure 3.5: Overall performance on USPS digit classification application.

3.6 Concluding Remarks

In this chapter, we have presented a probabilistic approach to learning the relationships between tasks in multi-task learning. Our method can model global task relationships and the learning problem can be formulated directly as a convex optimization problem by utilizing the matrix variate normal distribution as a prior. We study the proposed method under both symmetric and asymmetric multi-task learning settings. By utilizing a similar idea, we propose a transfer metric learning method where the relationships between the source tasks and the target tasks as well as the distance metric in the target task can be learned in a convex regularized objective function.

CHAPTER 4

MULTI-TASK GENERALIZED t PROCESS

4.1 Introduction

Bonilla et al. proposed a multi-task GP model (Bonilla et al., 2007) which uses a task covariance matrix to model the relationships between tasks. The advantage of using a task covariance matrix is that it can model not only tasks with positive or zero correlation but also tasks with negative correlation. Moreover, since the model is very similar to conventional GP regression models, it can make inference as efficiently as other GP models. Despite these advantages, the model does have some drawbacks. For example, since the task covariance matrix is learned in a nonparametric manner, the computational cost will be very high if there exist many tasks because the task covariance matrix will be very large. To overcome this problem, Bonilla et al. resorted to a low-rank approximation of the task covariance matrix. While this scheme can partially address the computational concern, the expressive power of the (approximated) task covariance matrix may be impaired. Moreover, since the log-likelihood in (Bonilla et al., 2007) is non-convex with respect to the task covariance matrix (or its low-rank decomposition matrix), there may exist many local extrema and so there is no guarantee of finding the globally optimal solution even when sophisticated optimization methods are used.

In this chapter, we propose a Bayesian model to address the problem via Bayesian model averaging. Instead of treating the task covariance matrix as a fixed parameter matrix that is estimated using point estimation, we model it as a random matrix with an inverse-Wishart prior and integrate it out for the subsequent inference. However, since directly integrating out the task covariance matrix is computationally difficult, we first give an alternative weight-space view of the model in (Bonilla et al., 2007) and then integrate out the task covariance matrix in the model, leading to a multi-task generalized t process (MTGTP). Unlike the model in (Bonilla et al., 2007), we adopt a generalized t noise model for the likelihood which, together with the generalized t process prior, not only has the robustness advantage but also leads to an analytical form for the marginal likelihood (or model evidence). In order to specify the inverse-Wishart prior, we use the maximum mean discrepancy (MMD) statistic (Gretton et al., 2006) to estimate the parameter matrix of the inverse-Wishart prior. Moreover, we investigate some theoretical properties of MTGTP, such as its asymptotic analysis and learning curve.

4.2 Multi-Task Generalized t Process

In this section, we propose an extension to (Bonilla et al., 2007) via Bayesian model averaging (Gelman et al., 2003) by integrating out the task covariance matrix $\mathbf{\Sigma}$ for the subsequent inference. Our model also uses MMD, as in (Bonilla et al., 2007), but for learning the parameter matrix of the prior distribution for the task covariance matrix.

4.2.1 A Weight-Space View of Multi-Task Gaussian Process

Before we present our model, we first give a weight-space view of the multi-task GP model in (Bonilla et al., 2007). This alternative view will be very useful for our subsequent derivation.

Recall that GP has an interpretation from the weight-space view (Rasmussen & Williams, 2006). Here we also give a weight-space view of the multi-task GP model in (Bonilla et al., 2007):

$$\begin{aligned} y_j^i &= \mathbf{w}_i^T \phi(\mathbf{x}_j^i) + \varepsilon_j^i \\ \mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_m] &\sim \mathcal{MN}_{d' \times m}(\mathbf{0}_{d' \times m}, \mathbf{I}_{d'} \otimes \mathbf{\Sigma}) \\ \varepsilon_j^i &\sim \mathcal{N}(0, \sigma_i^2), \end{aligned} \tag{4.1}$$

where $\phi(\cdot)$, which maps $\mathbf{x} \in \mathbb{R}^d$ to $\phi(\mathbf{x}) \in \mathbb{R}^{d'}$ and may have no explicit form, denotes a feature map corresponding to the kernel function $k(\cdot, \cdot)$. The equivalence between the model formulations in (3.26) and (4.1) is due to the following which is a consequence of the property of the matrix variate normal distribution:[1]

$$f_j^i \stackrel{\text{def}}{=} \phi(\mathbf{x}_j^i)^T \mathbf{w}_i = \phi(\mathbf{x}_j^i)^T \mathbf{W} \mathbf{e}_{m,i} \sim \mathcal{N}(0, \Sigma_{ii} k(\mathbf{x}_j^i, \mathbf{x}_j^i)) \tag{4.2}$$

$$\langle f_j^i, f_s^r \rangle = \int \phi(\mathbf{x}_j^i)^T \mathbf{W} \mathbf{e}_{m,i} \mathbf{e}_{m,r}^T \mathbf{W}^T \phi(\mathbf{x}_s^r) p(\mathbf{W}) d\mathbf{W} = \Sigma_{ir} k(\mathbf{x}_j^i, \mathbf{x}_s^r), \tag{4.3}$$

where $\mathbf{e}_{m,i}$ is the ith column of $\mathbf{I}_m$. The weight-space view of the conventional GP can be seen as a special case of that of the multi-task GP with $m = 1$, under which the prior for $\mathbf{W}$ in (4.1) will become the ordinary normal distribution with zero mean and identity covariance matrix by setting $\mathbf{\Sigma} = 1$.

4.2.2 Our Model

Since the random matrix $\mathbf{\Sigma}$ is positive semidefinite, we place an inverse-Wishart prior on it:

$$\mathbf{\Sigma} \sim \mathcal{IW}_m(\nu, \mathbf{\Psi}), \tag{4.4}$$

[1] The proofs for the following two equations can be found in Appendix B.

where the probability density function for $\mathcal{IW}_m(\nu, \mathbf{\Psi})$ is

$$\frac{|\mathbf{\Psi}|^{\nu/2} |\mathbf{\Sigma}|^{-(m+\nu+1)/2} e^{-\mathrm{tr}(\mathbf{\Psi}\mathbf{\Sigma}^{-1})/2}}{2^{m\nu/2}\Gamma_m(\nu/2)}.$$

Note that it is computationally not easy to integrate out $\mathbf{\Sigma}$ in (3.26) with the inverse-Wishart prior in (4.4). This is where the alternative weight-space view in (4.1) can play an important role in making Bayesian model averaging possible.

Since $\mathbf{W} \mid \mathbf{\Sigma} \sim \mathcal{MN}_{d' \times m}(\mathbf{0}_{d' \times m}, \mathbf{I}_{d'} \otimes \mathbf{\Sigma})$ and $\mathbf{\Sigma} \sim \mathcal{IW}_m(\nu, \mathbf{\Psi})$, then according to (Gupta & Nagar, 2000) we can get

$$p(\mathbf{W}) = \int p(\mathbf{W}|\mathbf{\Sigma}) p(\mathbf{\Sigma}) d\mathbf{\Sigma} = \mathcal{MT}_{d' \times m}(\nu, \mathbf{0}_{d' \times m}, \mathbf{I}_{d'} \otimes \mathbf{\Psi}).$$

Then, from the property of matrix variate t distribution, we can get

$$f_j^i \stackrel{\text{def}}{=} \phi(\mathbf{x}_j^i)^T \mathbf{w}_i = \phi(\mathbf{x}_j^i)^T \mathbf{W} \mathbf{e}_{m,i} \sim t(\nu, 1, 0, \Psi_{ii} k(\mathbf{x}_j^i, \mathbf{x}_j^i)),$$

where Ψ_{ij} is the (i, j)th element of $\mathbf{\Psi}$ and $t(\nu, \omega, \mathbf{m}, \mathbf{\Sigma})$ denotes the univariate or multivariate generalized t distribution (Arellano-Valle & Bolfarine, 1995). The probability density function of the generalized t distribution $t(\nu, \omega, \mathbf{m}, \mathbf{\Sigma})$ for a random variable $\mathbf{x} \in \mathbb{R}^d$ is

$$\frac{\Gamma((\nu + d)/2)}{\pi^{d/2}\Gamma(\nu/2)|\omega\mathbf{\Sigma}|^{\frac{1}{2}}} \Big[1 + \frac{1}{\omega}(\mathbf{x} - \mathbf{m})^T \mathbf{\Sigma}^{-1} (\mathbf{x} - \mathbf{m})\Big]^{-(\nu+d)/2}, \tag{4.5}$$

where $\Gamma(\cdot)$ denotes the gamma function. When $\nu = \omega$, the generalized t distribution reduces to the t distribution. From (4.5), we can see that the probability density functions of $t(\nu, \omega, \mathbf{m}, \mathbf{\Sigma})$ and $t(\nu, 1, \mathbf{m}, \omega\mathbf{\Sigma})$ are the same. Moreover, the generalized t distribution can be viewed as a special case of the matrix variate t distribution. The following property of the generalized t distribution is easy to obtain using the property of the matrix variate t distribution.

Proposition 4.1 *Let* $\mathbf{x} \sim t(\nu, \omega, \mathbf{m}, \mathbf{\Sigma})$*, then*

$$\mathbb{E}(\mathbf{x}) = \mathbf{m},$$
$$\mathrm{cov}(\mathbf{x}) = \frac{\omega}{\nu - 2}\mathbf{\Sigma} \quad \textit{for } \nu > 2.$$

Moreover, from Theorem 4.2.1 in (Gupta & Nagar, 2000), $\mathbf{W}$ can be represented as $\mathbf{W} = \mathbf{S}^{-1/2}\mathbf{Z}$ where the two random variables $\mathbf{S} \sim \mathcal{W}_{d'}(\nu + d' - 1, \mathbf{I}_{d'})$ and $\mathbf{Z} \sim \mathcal{MN}_{d' \times m}(\mathbf{0}_{d' \times m}, \mathbf{I}_{d'} \otimes$

$\mathbf{\Psi}$) are independent. Then we get

$$\begin{aligned}
\langle f_j^i, f_s^r \rangle &= \int\int \phi(\mathbf{x}_j^i)^T \mathbf{w}_i \mathbf{w}_r^T \phi(\mathbf{x}_s^r) p(\mathbf{w}_i) p(\mathbf{w}_r) d\mathbf{w}_i d\mathbf{w}_r \\
&= \int \phi(\mathbf{x}_j^i)^T \mathbf{W} \mathbf{e}_{m,i} \mathbf{e}_{m,r}^T \mathbf{W}^T \phi(\mathbf{x}_s^r) p(\mathbf{W}) d\mathbf{W} \\
&= \int\int \phi(\mathbf{x}_j^i)^T \mathbf{S}^{-1/2} \mathbf{Z} \mathbf{e}_{m,i} \mathbf{e}_{m,r}^T \mathbf{Z}^T \mathbf{S}^{-1/2} \phi(\mathbf{x}_s^r) p(\mathbf{Z}) p(\mathbf{S}) d\mathbf{Z} d\mathbf{S} \\
&= \Psi_{ir} \int \phi(\mathbf{x}_j^i)^T \mathbf{S}^{-1} \phi(\mathbf{x}_s^r) p(\mathbf{S}) d\mathbf{S} \\
&= \frac{\Psi_{ir} k(\mathbf{x}_j^i, \mathbf{x}_s^r)}{\nu - 2},
\end{aligned} \tag{4.6}$$

where the last two equations hold because of the properties of the matrix variate normal distribution and the Wishart distribution.[2] So the prior for $\mathbf{f} = (f_1^1, \ldots, f_{n_m}^m)^T$ is formulated as[3]

$$\mathbf{f} \sim t(\nu, 1, \mathbf{0}_{n\times 1}, \mathbf{K}). \tag{4.7}$$

The element in $\mathbf{K}$ corresponding to $\mathbf{x}_j^i$ and $\mathbf{x}_s^r$ is equal to $\Psi_{ir} k(\mathbf{x}_j^i, \mathbf{x}_s^r)$. Since the generalized t distribution has the consistency property (Arellano-Valle & Bolfarine, 1995), the prior for $\mathbf{f}$ is the generalized t process.

Definition 4.1 *(Generalized t Process) A random, real-valued function $f : \mathbb{R}^d \to \mathbb{R}$ is said to follow a* generalized t process *with degrees of freedom ν and ω, mean function $h(\cdot)$ and covariance function $\kappa(\cdot, \cdot)$, if for any positive integer N and any $\mathbf{x}_1, \ldots, \mathbf{x}_N \in \mathbb{R}^d$, $\mathbf{f} = (f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N))^T \sim t(\nu, \omega, \mathbf{h}, \mathbf{K})$ where $\mathbf{h} = (h(\mathbf{x}_1), \ldots, h(\mathbf{x}_N))^T$ and $\mathbf{K} = [\kappa(\mathbf{x}_i, \mathbf{x}_j)]_{i,j=1}^N$.*

In our case, the mean function $h(\cdot)$ is the zero function.

We use the generalized t noise model for the likelihood, i.e., for y_j^i,

$$y_j^i \,|\, f_j^i \sim t(\nu, 1, f_j^i, \sigma_i^2).$$

The purpose for this choice is two-fold: to make our model more robust against outliers and to obtain an analytical form for the marginal likelihood in regression problems, as we will see later. We can then get the likelihood for $\mathbf{y} = (y_1^1, \ldots, y_{n_m}^m)^T$ as

$$\mathbf{y} \,|\, \mathbf{f} \sim t(\nu, 1, \mathbf{f}, \mathbf{D}), \tag{4.8}$$

where $\mathbf{D}$ denotes a diagonal matrix in which each diagonal element is σ_i^2 if the corresponding data point belongs to the ith task.

[2] The proof can be found in Appendix B.

[3] Since the probability density functions of $t(\nu, 1, \mathbf{m}, \mathbf{K})$ and $t(\nu, \omega, \mathbf{m}, \frac{1}{\omega}\mathbf{K})$ where $\omega \neq 1$ are the same, the expressive power of our model is the same as that with $\omega \neq 1$.

It is not easy to integrate out f to obtain the marginal likelihood though and so we take an alternative approach. We rewrite the model as

$$\begin{aligned}
&\mathbf{y} = \mathbf{f} + \boldsymbol{\varepsilon} \\
&\mathbf{f} \sim t(\nu, 1, \mathbf{0}_n, \mathbf{K}) \\
&\boldsymbol{\varepsilon} \sim t(\nu, 1, \mathbf{0}_n, \mathbf{D}) \\
&\mathbf{f} \perp \boldsymbol{\varepsilon},
\end{aligned}$$

where $\mathbf{f} \perp \boldsymbol{\varepsilon}$ means that the two random variables $\mathbf{f}$ and $\boldsymbol{\varepsilon}$ are independent of each other. Since $\mathbf{f}$ and $\boldsymbol{\varepsilon}$ are independent, then according to the property of the generalized t distribution, we can get

$$\begin{pmatrix} \mathbf{f} \\ \boldsymbol{\varepsilon} \end{pmatrix} \sim t\left(\nu, 1, \mathbf{0}_{2n}, \begin{pmatrix} \mathbf{K} & \mathbf{0}_{n\times n} \\ \mathbf{0}_{n\times n} & \mathbf{D} \end{pmatrix}\right).$$

Since

$$\mathbf{y} = \mathbf{f} + \boldsymbol{\varepsilon} = (\mathbf{I}_n, \mathbf{I}_n) \begin{pmatrix} \mathbf{f} \\ \boldsymbol{\varepsilon} \end{pmatrix},$$

then according to the property of the generalized t distribution, we can get

$$\mathbf{y} \sim t(\nu, 1, \mathbf{0}_n, \mathbf{K} + \mathbf{D}), \tag{4.9}$$

which is the marginal likelihood.

To see the relation between MTGTP and the proposed framework in section 1.3, we rewrite the model as

$$\begin{aligned}
\mathbf{W} &\sim \mathcal{MT}_{d'\times m}(\nu, \mathbf{0}_{d'\times m}, \mathbf{I}_{d'} \otimes \boldsymbol{\Psi}) \\
y_j^i \,|\, \mathbf{W} &\sim t(\nu, 1, \mathbf{w}_i^T \mathbf{x}_j^i, \sigma_i^2).
\end{aligned}$$

So MTGTP is a special case of the proposed framework with matrix variate t prior and t likelihood. Different from MTRL, we propose a Bayesian model by integrating out $\mathbf{W}$ to achieve Bayesian model averaging.

4.2.3 Parameter Learning and Inference

The negative log-likelihood of all data points can be expressed as

$$l = \frac{1}{2}\Big[(\nu+n)\ln\left(1+\mathbf{y}^T(\mathbf{K}+\mathbf{D})^{-1}\mathbf{y}\right) + \ln|\mathbf{K}+\mathbf{D}|\Big] - \ln\frac{\Gamma((\nu+n)/2)}{\Gamma(\nu/2)} + \text{Const.} \tag{4.10}$$

When n is large, $\Gamma((\nu+n)/2)$ is very large and so the calculation of $\Gamma((\nu+n)/2)$ may be numerically unstable. By utilizing the fact that n is an integer and a property of the gamma

function that $\Gamma(z+1) = z\Gamma(z)$, we get the following simplification:

$$\ln \frac{\Gamma((\nu+n)/2)}{\Gamma(\nu/2)} = \begin{cases} \sum_{i=0}^{n/2-1} \ln(\nu/2+i) & \text{if } n \text{ is even} \\ \ln \frac{\Gamma((\nu+1)/2)}{\Gamma(\nu/2)} + \sum_{i=0}^{(n-3)/2} \ln((\nu+1)/2+i) & \text{otherwise.} \end{cases}$$

We use a gradient method to minimize l in order to estimate the optimal values of $\{\sigma_i\}$, ν and $\boldsymbol{\theta}$, which contains the kernel parameters in the kernel function $k(\cdot,\cdot)$ and the parameters in the parameter matrix for the inverse-Wishart prior of $\boldsymbol{\Sigma}$. In our implementation, we use $\ln \sigma_i$, $\ln \nu$ and $\ln \theta_j$, where θ_j is the jth element in $\boldsymbol{\theta}$, as the optimization variables because $\{\sigma_i\}$, ν and $\{\theta_j\}$ have to be positive. The gradients of the negative log-likelihood with respect to $\ln \sigma_i$, $\ln \nu$ and $\ln \theta_j$ can be computed as:

$$\begin{aligned} \frac{\partial l}{\partial \ln \sigma_i} =& \sigma_i^2 \text{tr}\left[\left(\tilde{\mathbf{K}}^{-1} - \frac{\nu+n}{1+\mathbf{y}^T\tilde{\mathbf{K}}^{-1}\mathbf{y}}\tilde{\mathbf{K}}^{-1}\mathbf{y}\mathbf{y}^T\tilde{\mathbf{K}}^{-1}\right)\mathbf{I}_n^i\right] \\ \frac{\partial l}{\partial \ln \theta_j} =& \frac{\theta_j}{2}\text{tr}\left[\left(\tilde{\mathbf{K}}^{-1} - \frac{\nu+n}{1+\mathbf{y}^T\tilde{\mathbf{K}}^{-1}\mathbf{y}}\tilde{\mathbf{K}}^{-1}\mathbf{y}\mathbf{y}^T\tilde{\mathbf{K}}^{-1}\right)\frac{\partial \mathbf{K}}{\partial \theta_j}\right] \\ \frac{\partial l}{\partial \ln \nu} =& \frac{\nu}{2}\left[\ln\left(1+\mathbf{y}^T\tilde{\mathbf{K}}^{-1}\mathbf{y}\right) - g(n,\nu)\right], \end{aligned}$$

where

$$g(n,\nu) = \begin{cases} \sum_{i=0}^{n/2-1} \frac{1}{\nu/2+i} & \text{if } n \text{ is even} \\ \psi\left((\nu+1)/2\right) - \psi(\nu/2) + \sum_{i=0}^{(n-3)/2} \frac{1}{(\nu+1)/2+i} & \text{otherwise.} \end{cases}$$

Here $\psi(\cdot)$ is the digamma function, $\tilde{\mathbf{K}} = \mathbf{K} + \mathbf{D}$, and $\mathbf{I}_n^i$ denotes an $n \times n$ diagonal 0-1 matrix where a diagonal element is equal to 1 if and only if the data point with the corresponding index belongs to the ith task.

Suppose we are given a test data point $\mathbf{x}_\star^i$ for the ith task and we need to determine the corresponding output $y_\star^i$. From the marginal likelihood calculated in Eq. (4.9), we get

$$\begin{pmatrix} \mathbf{y} \\ y_\star^i \end{pmatrix} \sim t\left(\nu, 1, \mathbf{0}_{n+1}, \begin{pmatrix} \mathbf{K}+\mathbf{D} & \mathbf{k}_\star^i \\ (\mathbf{k}_\star^i)^T & \Psi_{ii}k(\mathbf{x}_\star^i, \mathbf{x}_\star^i) + \sigma_i^2 \end{pmatrix}\right),$$

where $\mathbf{k}_\star^i = (\Psi_{i1}k(\mathbf{x}_\star^i, \mathbf{x}_1^1), \ldots, \Psi_{im}k(\mathbf{x}_\star^i, \mathbf{x}_{n_m}^m))^T$. Then, from the property of the generalized t distribution, the predictive distribution $p(y_\star^i \mid x_\star^i, \mathbf{X}, \mathbf{y})$ is $t(n+\nu, 1+\mathbf{y}^T\tilde{\mathbf{K}}^{-1}\mathbf{y}, m_\star^i, \rho_\star^i)$ with

$$m_\star^i = (\mathbf{k}_\star^i)^T\tilde{\mathbf{K}}^{-1}\mathbf{y} \tag{4.11}$$

$$\rho_\star^i = \Psi_{ii}k(\mathbf{x}_\star^i, \mathbf{x}_\star^i) + \sigma_i^2 - (\mathbf{k}_\star^i)^T\tilde{\mathbf{K}}^{-1}\mathbf{k}_\star^i, \tag{4.12}$$

where $\tilde{\mathbf{K}} = \mathbf{K} + \mathbf{D}$.

4.2.4 The Choice of $\mathbf{\Psi}$

The parameter matrix $\mathbf{\Psi}$ contains prior information about the relationships between tasks. Here we use the maximum mean discrepancy (MMD) in (Gretton et al., 2006) to measure the task relationships. MMD is a nonparametric test statistic with a simple intuitive form which has been found to be quite powerful in its test performance compared to other test criteria. Moreover, since MMD is defined based on some kernel, it is easy to use it with GP and t processes. In (Gretton et al., 2006), the distance between the sample means of two distributions was used to measure the distance between the two distributions. Here we use it to measure the distance between two tasks instead. Specifically, the (squared) distance between the ith and jth tasks is given by

$$d_{ij}^2 = \left\| \frac{1}{n_i} \sum_{k=1}^{n_i} \phi'(\mathbf{x}_k^i) - \frac{1}{n_j} \sum_{l=1}^{n_j} \phi'(\mathbf{x}_l^j) \right\|_2^2,$$

where $\phi'(\cdot)$ denotes the feature map corresponding to a universal kernel $k'(\cdot, \cdot)$. Then, according to the relationship between a distance matrix and a kernel matrix as in multidimensional scaling (Williams, 2000), we can get

$$\mathbf{\Psi} = \mathbf{H}\mathbf{\Psi}'\mathbf{H},$$

where $\mathbf{H} = \mathbf{I}_m - \frac{1}{m}\mathbf{1}_m\mathbf{1}_m^T$ is the $m \times m$ centering matrix and the (i, j)th element of $\mathbf{\Psi}'$ is

$$\Psi'_{ij} = \left(\frac{1}{n_i} \sum_{k=1}^{n_i} \phi'(\mathbf{x}_k^i) \right) \left(\frac{1}{n_j} \sum_{l=1}^{n_j} \phi'(\mathbf{x}_l^j) \right) = \frac{1}{n_i n_j} \sum_{k=1}^{n_i} \sum_{l=1}^{n_j} k'(\mathbf{x}_k^i, \mathbf{x}_l^j).$$

So the kernel function used in MTGTP is $k_{MT}(\mathbf{x}_j^i, \mathbf{x}_q^p) = \Psi_{ip} k(\mathbf{x}_j^i, \mathbf{x}_q^p)$.

4.2.5 Discussions

The marginal likelihood in our model has the analytical form given in Eq. (4.9) which facilitates solving the model selection problem. This nice property can be attributed to the assumption in our model that the prior and the likelihood are both generalized t distributions with the same number of degrees of freedom ν. It is possible to relax this assumption by allowing different numbers of degrees of freedom, but more sophisticated techniques such as EM have to be used to learn the model parameters. To reduce the computation cost, we just adopt the assumption in this chapter to work with the nice analytical form. Moreover, for other likelihood models such as probit likelihood for classification problems, we need to resort to sampling methods or approximation methods such as Laplace approximation, variational method or expectation propagation.

The task relationship such as task covariance can be found in (Bonilla et al., 2007). In our model, it is also easy to find the task covariance if we so wish. After obtaining the optimal

$\mathbf{\Psi}$, we can find the task covariance matrix based on the MAP framework. Moreover, using the inverse-Wishart prior, the size of the search space for the task covariance matrix is reduced and hence the optimal task covariance matrix can be found more efficiently.

The complexity of our algorithm is $O(n^3)$, which makes it unfavorable for large-scale applications. To address this problem, we use the Nyström approximation of $\mathbf{K}$ in the marginal likelihood of Eq. (4.9) and the negative log-likelihood of Eq. (4.10), that is, $\mathbf{K} \approx \hat{\mathbf{K}} \stackrel{\text{def}}{=} \mathbf{K}_{\cdot I}\mathbf{K}_{II}^{-1}\mathbf{K}_{I\cdot}$ where I indexes $r\,(\ll n)$ rows or columns of $\mathbf{K}$. Then we can compute $(\mathbf{K}+\mathbf{D})^{-1}$ using the Woodbury identity

$$(\mathbf{K}+\mathbf{D})^{-1} \approx (\hat{\mathbf{K}}+\mathbf{D})^{-1} = \mathbf{D}^{-1} - \mathbf{D}^{-1}\mathbf{K}_{\cdot I}(\mathbf{K}_{II} + \mathbf{K}_{I\cdot}\mathbf{D}^{-1}\mathbf{K}_{\cdot I})^{-1}\mathbf{K}_{I\cdot}\mathbf{D}^{-1}.$$

Note that $\mathbf{D}$ is a diagonal matrix and so $\mathbf{D}^{-1}$ can be computed very efficiently. Moreover, $|\mathbf{K}+\mathbf{D}|$ can be approximated as

$$\begin{aligned}|\mathbf{K}+\mathbf{D}| &\approx |\hat{\mathbf{K}}+\mathbf{D}| \\ &= |\mathbf{D}|\left|\mathbf{I}_n + \mathbf{D}^{-\frac{1}{2}}\mathbf{K}_{\cdot I}\mathbf{K}_{II}^{-1}\mathbf{K}_{I\cdot}\mathbf{D}^{-\frac{1}{2}}\right| \\ &= |\mathbf{D}|\left|\mathbf{I}_r + \mathbf{K}_{II}^{-1}\mathbf{K}_{I\cdot}\mathbf{D}^{-1}\mathbf{K}_{\cdot I}\right|.\end{aligned}$$

In other words, this can be reduced to the calculation of the determinant of an $r \times r$ matrix but not that of the original $n \times n$ matrix, which is much larger.

4.3 Theoretical Analysis

There exist some previous research works on the asymptotic analysis and learning curve of GP, e.g., (Opper & Vivarelli, 1998; Sollich, 1998; Williams & Vivarelli, 2000; Sollich & Halees, 2002). For the t process as well as generalized t process, however, we are not aware of any such previous effort. In this section, we wish to fill this gap by analyzing the multi-task generalized t process $t(\nu, \omega, \mathbf{h}, \mathbf{K})$. The property of the multi-task t process can be obtained by setting $\omega = \nu$ and that of our model by setting $\omega = 1$.

4.3.1 Asymptotic Analysis

We first analyze the asymptotics of the MTGTP model for regression when the data set sizes $n_i \to \infty$ for $i = 1, \ldots, m$. The predictive mean for MTGTP in Eq. (4.11) can be obtained as the function which minimizes the functional

$$J[h] = \frac{1}{2}\|h\|_H^2 + \sum_{i=1}^{m}\frac{1}{2\sigma_i^2}\sum_{j=1}^{n_i}(y_j^i - h(\mathbf{x}_j^i))^2, \tag{4.13}$$

where $\|h\|_H$ is the RKHS norm corresponding to the kernel k_{MT}. This follows from the representer theorem which states that each minimizer $h \in H$ of $J[h]$ has the form $h(\mathbf{x}) = \sum_{i=1}^{m}\sum_{j=1}^{n_i} \alpha_j^i k_{MT}(\mathbf{x}, \mathbf{x}_j^i)$. Let $\mu(\mathbf{x}, y)$ be the probability measure of the RKHS from which the data points $(\mathbf{x}_j^i, y_j^i)$ are generated.[4] We consider a smoothed version of Eq. (4.13) as follows:

$$J_\mu[h] = \frac{1}{2}\|h\|_H^2 + \mathbb{E}_\mu\Big[\sum_{i=1}^{m}\frac{1}{2\sigma_i^2}\sum_{j=1}^{n_i}(y_j^i - h(\mathbf{x}_j^i))^2\Big].$$

Observe that

$$\mathbb{E}_\mu\Big[\sum_{i=1}^{m}\frac{1}{2\sigma_i^2}\sum_{j=1}^{n_i}(y_j^i - h(\mathbf{x}_j^i))^2\Big] \tag{4.14}$$

$$=\Big(\sum_{i=1}^{m}\frac{n_i}{2\sigma_i^2}\Big)\int (y - h(\mathbf{x}))^2 d\mu(\mathbf{x}, y)$$

$$=\Big(\sum_{i=1}^{m}\frac{n_i}{2\sigma_i^2}\Big)\Big[\int (\eta(\mathbf{x}) - h(\mathbf{x}))^2 d\mu(\mathbf{x}) + \int (y - \eta(\mathbf{x}))^2 d\mu(\mathbf{x}, y)\Big], \tag{4.15}$$

where $\eta(\mathbf{x}) = \mathbb{E}[y|\mathbf{x}]$ is the regression function corresponding to $\mu(y|\mathbf{x})$ and the cross term vanishes because of the definition of $\eta(\mathbf{x})$. Note that the second term in Eq. (4.15) is independent of h and so minimizing $J_\mu[h]$ is equivalent to minimizing the following equation:

$$J_\mu[h] = \Big(\sum_{i=1}^{m}\frac{n_i}{2\sigma_i^2}\Big)\int (\eta(\mathbf{x}) - h(\mathbf{x}))^2 d\mu(\mathbf{x}) + \frac{1}{2}\|h\|_H^2.$$

We assume the kernel k_{MT} is non-degenerate so that the eigenfunctions $\{\psi_i(\mathbf{x})\}$ form a complete orthonormal basis, that is,

$$\int \psi_i(\mathbf{x})\psi_j(\mathbf{x}) d\mu(\mathbf{x}) = \delta(i, j),$$

where $\delta(i, j)$ is the Kronecker delta function. Let λ_i be the eigenvalue corresponding to $\psi_i(\cdot)$. We can write $h(\mathbf{x}) = \sum_{i=1}^{\infty} h_i\psi_i(\mathbf{x})$ and $\eta(\mathbf{x}) = \sum_{i=1}^{\infty}\eta_i\psi_i(\mathbf{x})$. Then $J_\mu[h]$ can be reformulated as

$$J_\mu[h] = \Big(\sum_{k=1}^{m}\frac{n_k}{2\sigma_k^2}\Big)\sum_{i=1}^{\infty}(\eta_i - h_i)^2 + \frac{1}{2}\sum_{i=1}^{\infty}\frac{h_i^2}{\lambda_i}.$$

Differentiating it with respect to h_i, we can get

$$h_i = \frac{\lambda_i}{\lambda_i + 1/(\sum_{k=1}^{m}\frac{n_k}{\sigma_k^2})}\eta_i.$$

So when $n_k \to \infty$, h_i will converge to η_i and hence $h(\mathbf{x})$ will converge to $\eta(\mathbf{x})$.

[4]Since the task differences are captured by the kernel k_{MT}, data points for different tasks can be generated from the same probability measure.

4.3.2 Learning Curve

Next we consider the generalization bound of MTGTP. It is also referred to as the learning curve which relates the generalization error of a learning model to the amount of training data used. We assume that the data was actually generated from a generalized t process, similar to that for GP in (Opper & Vivarelli, 1998; Sollich, 1998; Sollich & Halees, 2002). Given a test data point $\mathbf{x}_\star^i$ for the ith task, the prediction for $\mathbf{x}_\star^i$ can be calculated using Eq. (4.11) and the Bayesian prediction error for the ith task on the training set D_n (which contain n_i data points for the ith task) can be calculated as

$$\begin{aligned}
\varepsilon_{D_n}^i(\mathbf{x}_\star^i) \stackrel{\text{def}}{=} & \quad \mathbb{E}_y\left[\left(y_\star^i - (\mathbf{k}_\star^i)^T\tilde{\mathbf{K}}^{-1}\mathbf{y}\right)^2\right] \\
= & \quad \mathbb{E}[(y_\star^i)^2] - 2(\mathbf{k}_\star^i)^T\tilde{\mathbf{K}}^{-1}\mathbb{E}[y_\star^i\mathbf{y}] + \text{tr}(\tilde{\mathbf{K}}^{-1}\mathbf{k}_\star^i(\mathbf{k}_\star^i)^T\tilde{\mathbf{K}}^{-1}\mathbb{E}[\mathbf{y}\mathbf{y}^T]) \\
= & \quad \frac{\omega}{\nu-2}\left[k_{MT}(\mathbf{x}_\star^i,\mathbf{x}_\star^i) + \sigma_i^2 - (\mathbf{k}_\star^i)^T\tilde{\mathbf{K}}^{-1}\mathbf{k}_\star^i\right],
\end{aligned}$$

where $y_\star^i$ denotes the ground truth of the output value for $\mathbf{x}_\star^i$. Recall that the eigen-decomposition of k_{MT} is $k_{MT}(\mathbf{x},\mathbf{x}') = \sum_{j=1}^\infty \lambda_j\psi_j(\mathbf{x})\psi_j(\mathbf{x}')$. So the generalization error for the ith task with respect to the training set D_n can be calculated as

$$\begin{aligned}
\varepsilon_{D_n}^i \stackrel{\text{def}}{=} & \quad \mathbb{E}_x[\varepsilon_{D_n}(\mathbf{x}_\star^i)] \\
= & \quad \frac{\omega}{\nu-2}\left[\mathbb{E}[k_{MT}(\mathbf{x}_\star^i,\mathbf{x}_\star^i)] + \sigma_i^2 - \text{tr}\left((\mathbf{K}+\mathbf{D})^{-1}\mathbb{E}[\mathbf{k}_\star^i(\mathbf{k}_\star^i)^T]\right)\right].
\end{aligned}$$

Let $\mathbf{\Lambda}$ denote a diagonal matrix whose (j,j)th element is λ_j and $\mathbf{\Omega}$ be a matrix whose (r,s)th element corresponding to $\mathbf{x}_q^p$ is $\psi_r(\mathbf{x}_q^p)$. Then

$$\mathbb{E}[k_{MT}(\mathbf{x}_\star^i,\mathbf{x}_\star^i)] = \int\sum_{j=1}^\infty \lambda_j\psi_j(\mathbf{x}_\star^i)\psi_j(\mathbf{x}_\star^i)d\mu(\mathbf{x}_\star^i) = \text{tr}(\mathbf{\Lambda})$$

and the (a,b)th element of $\mathbb{E}[\mathbf{k}_\star^i(\mathbf{k}_\star^i)^T]$ corresponding to $\mathbf{x}_q^p$ and $\mathbf{x}_s^r$ is equal to

$$\int\sum_{j=1}^\infty \lambda_j\psi_j(\mathbf{x}_\star^i)\psi_j(\mathbf{x}_q^p)\sum_{k=1}^\infty \lambda_k\psi_k(\mathbf{x}_\star^i)\psi_k(\mathbf{x}_s^r)d\mu(\mathbf{x}_\star^i) = \sum_{j=1}^\infty \lambda_j^2\psi_j(\mathbf{x}_q^p)\psi_j(\mathbf{x}_s^r)$$

and so $\mathbb{E}[\mathbf{k}_\star^i(\mathbf{k}_\star^i)^T] = \mathbf{\Omega}^T\mathbf{\Lambda}^2\mathbf{\Omega}$. It is easy to show that $\mathbf{K} = \mathbf{\Omega}^T\mathbf{\Lambda}\mathbf{\Omega}$ and then $\varepsilon_{D_n}^i$ can be simplified as

$$\begin{aligned}
\varepsilon_{D_n}^i = & \frac{\omega}{\nu-2}\left[\sigma_i^2 + \text{tr}(\mathbf{\Lambda}) - \text{tr}\left((\mathbf{\Omega}^T\mathbf{\Lambda}\mathbf{\Omega}+\mathbf{D})^{-1}\mathbf{\Omega}^T\mathbf{\Lambda}^2\mathbf{\Omega}\right)\right] \\
= & \frac{\omega}{\nu-2}\left[\sigma_i^2 + \text{tr}\left((\mathbf{\Lambda}^{-1}+\mathbf{\Omega}\mathbf{D}^{-1}\mathbf{\Omega}^T)^{-1}\right)\right]. \qquad (4.16)
\end{aligned}$$

The last equation holds as a consequence of the Woodbury identity. We introduce a lemma from (Opper & Vivarelli, 1998) which is useful for deriving the bound for $\varepsilon_{D_n}^i$.

Lemma 4.1 *Let* $\mathbf{H}$ *be a real symmetric matrix and* f *be a real convex function, then tr*$(f(\mathbf{H})) \geq \sum_k f(H_{kk})$ *where* H_{kk} *is the* (k,k)*th element of* $\mathbf{H}$.

Here f is defined as the multiplicative inverse of a positive scalar or matrix inverse of a positive definite matrix. Then

$$\text{tr}\left((\mathbf{\Lambda}^{-1} + \mathbf{\Omega}\mathbf{D}^{-1}\mathbf{\Omega}^T)^{-1}\right) \geq \sum_k \frac{1}{\frac{1}{\lambda_k} + \sum_i \frac{\Omega_{ki}^2}{\sigma_{T(i)}^2}} \geq \sum_k \frac{1}{\frac{1}{\lambda_k} + \nu_k \sum_{i=1}^m \frac{n_i}{\sigma_i^2}},$$

where $\nu_k = \sup_{\mathbf{x}\in R}\psi_k^2(\mathbf{x})$ by assuming that all n data points lie in a compact region R and $T(i)$ is the task indicator for the ith data point. Then we can get the generalization bound for a fixed training set as

$$\varepsilon_{D_n}^i \geq \frac{\omega}{\nu - 2}\left[\sigma_i^2 + \sum_k \frac{1}{\frac{1}{\lambda_k} + \nu_k \sum_{j=1}^m \frac{n_j}{\sigma_j^2}}\right].$$

Based on this, we get the average-case bound for the ith task as

$$\varepsilon^i = \mathbb{E}[\varepsilon_{D_n}^i] \geq \frac{\omega}{\nu - 2}\left[\sigma_i^2 + \sum_k \frac{1}{\frac{1}{\lambda_k} + \mu_k \sum_{j=1}^m \frac{n_j}{\sigma_j^2}}\right],$$

where $\mu_k = \mathbb{E}[\psi_k^2(\mathbf{x})] = 1$. So the average-case generalization bound is

$$\varepsilon^i \geq \frac{\omega}{\nu - 2}\left[\sigma_i^2 + \sum_k \frac{1}{\frac{1}{\lambda_k} + \sum_{j=1}^m \frac{n_j}{\sigma_j^2}}\right].$$

Similar to (Sollich & Halees, 2002), we give here an upper bound on the learning curve.

It is useful to see how the matrix $\mathbf{G} = (\mathbf{\Lambda}^{-1} + \mathbf{\Omega}\mathbf{D}^{-1}\mathbf{\Omega}^T)^{-1}$ changes when a new data point from the ith task is added to the training set. The change is

$$\mathbf{G}(n+1) - \mathbf{G}(n) = \left[\mathbf{G}^{-1}(n) + \sigma_i^{-2}\boldsymbol{\varphi}\boldsymbol{\varphi}^T\right]^{-1} - \mathbf{G}(n) = -\frac{\mathbf{G}(n)\boldsymbol{\varphi}\boldsymbol{\varphi}^T\mathbf{G}(n)}{\sigma_i^2 + \boldsymbol{\varphi}^T\mathbf{G}(n)\boldsymbol{\varphi}},$$

where $\boldsymbol{\varphi}$ is a column vector with the ith element $\psi_i(\mathbf{x}_\star^i)$ and $\mathbf{x}_\star^i$ is the newly added data point from the ith task. To get the exact learning curve, we have to average this change with respect to all training sets that include $\mathbf{x}_\star^i$. This is difficult to achieve though. Here we ignore the correlation between the numerator and denominator and average them separately. Moreover, we treat n as a continuous variable and get

$$\frac{\partial \mathbf{H}(n)}{\partial n} = -\frac{\mathbb{E}[\mathbf{G}^2(n)]}{\sigma_i^2 + \text{tr}(\mathbf{H}(n))},$$

where $\mathbf{H}(n) = \mathbb{E}[\mathbf{G}(n)]$. We also neglect the fluctuations in $\mathbf{G}(n)$ and then get $\mathbb{E}[\mathbf{G}^2(n)] = \mathbf{H}^2(n)$. So we can get

$$\frac{\partial \mathbf{H}(n)}{\partial n} = -\frac{\mathbf{H}^2(n)}{\sigma_i^2 + \text{tr}(\mathbf{H}(n))}$$
$$\frac{\partial \mathbf{H}^{-1}(n)}{\partial n} = -\mathbf{H}^{-1}(n)\frac{\partial \mathbf{H}(n)}{\partial n}\mathbf{H}^{-1}(n) = (\sigma_i^2 + \text{tr}(\mathbf{H}(n)))^{-1}\mathbf{I}.$$

Since $\mathbf{H}^{-1}(0) = \mathbf{\Lambda}^{-1}$, $\mathbf{H}^{-1}(n) = \mathbf{\Lambda}^{-1} + \sigma_i^{-2}n'\mathbf{I}$ where n' needs to obey the following

$$\frac{\partial \sigma_i^{-2}n'}{\partial n} = \frac{1}{\sigma_i^2 + \text{tr}(\mathbf{H}(n))} = \frac{1}{\sigma_i^2 + \text{tr}\left((\mathbf{\Lambda}^{-1} + \sigma_i^{-2}n'\mathbf{I})^{-1}\right)},$$

which is equivalent to

$$\frac{\partial n'}{\partial n} + \text{tr}\left((\mathbf{\Lambda}^{-1} + \sigma_i^{-2}n'\mathbf{I})^{-1}\right)\sigma_i^{-2}\frac{\partial n'}{\partial n} = 1.$$

By integrating both sides, we can see that n' satisfies the following equation

$$n' + \sum_j \ln(n' + \sigma_i^2\lambda_j^{-1}) = n.$$

Then we can get the upper bound as

$$\varepsilon_{UB}^i = \frac{\omega}{\nu - 2}\left[\sigma_i^2 + \text{tr}\left((\mathbf{\Lambda}^{-1} + \sigma_i^{-2}n'\mathbf{I})^{-1}\right)\right].$$

4.4 Related Work

Some multi-task learning methods based on GP were previously proposed. Lawrence and Platt (2004) adopted the assumption that all tasks share the same kernel parameters in a GP and learn the GP from multiple tasks. The same assumption was adopted in (Schwaighofer et al., 2004; Yu et al., 2005) but the difference is that the kernel matrix in the GP is learned in a nonparametric manner. Since there may exist outlier tasks among the tasks in real-world applications, the assumption that all tasks share the same kernel parameters or data representation may not hold well. To reduce the adverse effect of the outlier tasks, Yu et al. (2007) proposed a robust version of the method in (Schwaighofer et al., 2004; Yu et al., 2005) using a t process. Bonilla et al. (2007) proposed to use a task covariance matrix to model the relationships between tasks, making it capable of modeling tasks with positive, zero or negative correlation. Teh et al. (2005) proposed a semiparametric latent factor model for multi-output regression problems, a special case of multi-task learning, in which the latent function values are modeled as linear combinations of multiple GP priors.

As GP is not very robust against outliers, some recent attempts have been made based on a (heavy-tailed) t process which can be seen as a robust extension to GP. One of these is (Yu et al., 2007) as discussed above. Zhang et al. (2007) proposed a t process for multi-output regression problems in which the priors of the latent function values are modeled by a matrix variate t distribution. Zhu et al. (2007) also proposed a multi-task t process which is a Bayesian extension of (Schwaighofer et al., 2004). As such, it inherits the same assumption of (Schwaighofer et al., 2004) that all tasks share the same representation. Moreover, the model in (Zhu et al., 2007) is more suitable for multi-output problems since it models the function values of a data point for all tasks, which is not needed in the general multi-task learning setting.

4.5 Experiments

In this section, we evaluate our MTGTP model on two applications. The first one is the prediction of student examination scores. The goal is to predict the examination scores of students in a school (task). The second application is the learning of the inverse dynamics of a robot arm. The goal is to predict the joint torques of a robot arm from the joint angles, velocities and accelerations.

The performance measure we use is the normalized mean squared error (nMSE), which is the mean squared error divided by the variance of the target. It is equal to 1 minus the percentage explained variance (Argyriou et al., 2008a), which is often used as a performance measure in multi-task learning. Both the kernel $k(\cdot, \cdot)$ and the universal kernel $k'(\cdot, \cdot)$ for MMD are RBF kernels. We use Carl Rasmussen's implementation (`minimize.m`)[5] of a gradient method to optimize the marginal likelihood to learn the model parameters. We compare MTGTP with single-task learning based on GP[6], multi-task feature learning (MTFL)[7] (Argyriou et al., 2008a) and multi-task GP (Bonilla et al., 2007). Even though there exist some methods based on the t process (Yu et al., 2007; Zhang et al., 2007; Zhu et al., 2007), they mainly focus on multi-output problems and it is difficult to extend them to more general multi-task learning problems. Thus we do not include them in this comparative study.

4.5.1 Examination Score Prediction

The description for this application is depicted in section 3.4.4. We randomly select 10%, 20% and 30% of all the data to form training sets of different sizes. Data not selected for training form the test set. Each configuration is repeated for 10 trials and we record the mean and

[5]http://www.kyb.tuebingen.mpg.de/bs/people/carl/code/minimize/

[6]http://www.gaussianprocess.org/gpml/code/

[7]http://www.cs.ucl.ac.uk/staff/A.Argyriou/code/

Table 4.1: nMSE (in mean±std-dev) on examination score prediction for different training set sizes.

Training Set	Single-Task GP	MTFL	Multi-Task GP	MTGTP
10%	1.0015±0.0267	0.9570±0.0057	0.8955±0.0968	**0.6700±0.0037**
20%	0.8897±0.0127	0.8409±0.0095	0.8502±0.0972	**0.6512±0.0033**
30%	0.8234±0.0110	0.8260±0.0082	0.8094±0.0886	**0.6412±0.0031**

Table 4.2: nMSE (in mean±std-dev) on inverse dynamics learning for different training set sizes.

Training Set	Single-Task GP	MTFL	Multi-Task GP	MTGTP
100	0.4920±0.0027	0.0976±0.0073	0.2483±0.0124	**0.0456±0.0043**
200	0.4903±0.0018	0.0883±0.0075	0.2491±0.0154	**0.0290±0.0050**
300	0.4903±0.0019	0.0797±0.0072	0.2442±0.0050	**0.0212±0.0038**
400	0.4884±0.0016	0.0733±0.0040	0.2434±0.0033	**0.0161±0.0055**
500	0.4891±0.0032	0.0749±0.0044	0.2453±0.0061	**0.0138±0.0038**

standard deviation in each entry of Table 4.1. The best results[8] are shown in bold. We can see that MTGTP outperforms single-task GP, MTFL and multi-task GP. From the last column of Table 4.1, the performance of MTGTP is better when the training size becomes larger.

4.5.2 Robot Inverse Dynamics

The description for this application as well as the data is recorded in section 3.4.2. We randomly select $p \in \{100, 200, 300, 400, 500\}$ data points for each task to form the training set and 2000 data points for each task for the test set. We perform 10 random splits of the data and report the mean and standard deviation over the 10 trials. The results are summarized in Table 4.2 where the best results are shown in bold. From the results, we can see that the performance of MTGTP is also significantly better than that of the other methods compared. According to the performance of MTGTP under different configurations, we get the conclusion that the more the data, the better the performance of MTGTP.

4.6 Concluding Remarks

In this chapter, we have proposed a novel Bayesian extension of the model in (Bonilla et al., 2007) to alleviate some problems that arise when estimating the task covariance matrix. By integrating out the task covariance matrix, our model is as efficient as the conventional GP and its performance is significantly better than that of (Bonilla et al., 2007).

[8] Since the standard deviation is so small, the improvement is clearly very significant under significance testing.

CHAPTER 5

MULTI-TASK HIGH-ORDER TASK RELATIONSHIP LEARNING

5.1 Introduction

The task relationship learning approach based on a task covariance matrix is a promising approach because it considers more general types of task relationships than those exploited by previous methods. Specifically, not only can two tasks be positively correlated or unrelated, they can also be negatively correlated. Moreover, the task relationships are learned from data automatically but not prespecified and fixed based on possibly incorrect assumptions. Nevertheless, the existing task relationship learning methods (Bonilla et al., 2007; Zhang & Yeung, 2010a; Zhang et al., 2010) only model and learn pairwise task relationships. Some relationships between tasks found in real applications may be more complicated than what pairwise relationships can characterize. For example, it is possible for two tasks to appear uncorrelated when considering pairwise relationships, but their partial similarity can be revealed by considering high-order task relationships after incorporating a third task which is positively correlated with the two tasks. This motivates us to explore the use of high-order task relationships for multi-task learning.

We first briefly review an existing task relationship learning method and propose an alternative formulation of the model based on a matrix variate probability distribution. The new formulation allows us to generalize, though in a nontrivial way, the use of pairwise task relationships to high-order task relationships, leading to a new prior for the model parameters of different tasks. We then propose a new model which we refer to as Multi-Task High-Order relationship Learning (MTHOL). Experiments on some benchmark datasets are reported.

5.2 High-Order Task Relationship Learning

In this section, we present our method for modeling and learning high-order task relationships in the context of multi-task learning.

Let there be m supervised learning tasks and T_i denote the ith learning task. The training data for T_i is represented by a set of n_i independent and identically distributed (i.i.d.) input-output pairs $\left\{(\mathbf{x}_j^i, y_j^i)\right\}_{j=1}^{n_i}$. We assume that each input $\mathbf{x}_j^i \in \mathbb{R}^d$ and its corresponding output

$y_j^i \in \mathbb{R}$ for a regression task and $y_j^i \in \{-1, 1\}$ for a binary classification task. For both regression and classification tasks, we consider linear functions in the form $f_i(\mathbf{x}) = \mathbf{w}_i^T\mathbf{x} + b_i$ for the ith task T_i.

5.2.1 Generalization of an Existing Method

All existing multi-task learning methods that learn the relationships between tasks only exploit pairwise relationships. We first briefly review a recently proposed method, called Multi-Task Relationship Learning (MTRL) (Zhang & Yeung, 2010a), which belongs to this category of multi-task learning methods. The brief review also serves the purpose of introducing the notation to be used later in the chapter. We then propose a novel extension of MTRL for modeling and learning high-order task relationships.

Let the random matrix $\mathbf{W} = (\mathbf{w}_1, \ldots, \mathbf{w}_m)$ denote the model parameters for all m tasks. In MTRL, a matrix variate normal prior is placed on $\mathbf{W}$:

$$\mathbf{W} \sim \mathcal{MN}_{d\times m}(\mathbf{0}, \mathbf{I}_d \otimes \mathbf{\Omega}). \tag{5.1}$$

In Eq. (5.1), the column covariance matrix $\mathbf{\Omega}$ models the pairwise correlation between different columns of $\mathbf{W}$ which are the model parameters for different tasks. Thus $\mathbf{\Omega}$ can be thought of as modeling the pairwise correlation between different tasks via the corresponding model parameters.

Based on the matrix variate formulation of MTRL given in Eq. (5.1), we now proceed to propose a generalization which allows high-order task relationships to be modeled as well. Our point of departure is to exploit the close relationships between the matrix variate normal distribution and the Wishart distribution (Gupta & Nagar, 2000). It is easy to see that $\mathbf{W}^T$ has the following prior distribution

$$\mathbf{W}^T \sim \mathcal{MN}_{m\times d}(\mathbf{0}, \mathbf{\Omega} \otimes \mathbf{I}_d),$$

if $\mathbf{W}$ follows the prior distribution in Eq. (5.1). It thus follows that the square matrix $\mathbf{W}^T\mathbf{W}$ follows the Wishart distribution:

$$\mathbf{W}^T\mathbf{W} \sim \mathcal{W}_m(d, \mathbf{\Omega}). \tag{5.2}$$

Here we assume that $d \geq m$ or otherwise it will degenerate into a singular distribution (Uhling, 1994). As we will see later, this assumption is satisfied in our method and in most applications.

The Wishart prior on $\mathbf{W}^T\mathbf{W}$ can help us to understand the role played by the matrix $\mathbf{\Omega}$ better. From the properties of the Wishart distribution (Gupta & Nagar, 2000), we know that the mean and mode of $\mathbf{W}^T\mathbf{W}$ are $d\mathbf{\Omega}$ and $(d - m - 1)\mathbf{\Omega}$, respectively, which both depend on $\mathbf{\Omega}$. Since each element of $\mathbf{W}^T\mathbf{W}$ is the dot product of the model parameter vectors of the

corresponding tasks, it is a way to express the pairwise correlation between tasks. Thus $\mathbf{\Omega}$ is also related to pairwise task correlation.

We now propose a generalization of the above model for pairwise task relationships to high-order task relationships. We use the following prior distribution

$$(\mathbf{W}^T\mathbf{W})^t \sim \mathcal{W}_m(d, \mathbf{\Omega}), \tag{5.3}$$

where the integer $t \geq 1$ is called the degree and $\mathbf{X}^t = \mathbf{X}^{t-1}\mathbf{X}$ for a square matrix $\mathbf{X}$. Note that the prior in Eq. (5.2) is a special case of Eq. (5.3) when $t = 1$. We first investigate the physical meaning of each element in $(\mathbf{W}^T\mathbf{W})^t$. As discussed above, the (i, j)th element s_{ij} of $\mathbf{W}^T\mathbf{W}$ describes the pairwise similarity between the ith and jth tasks. Since the (i, j)th element s'_{ij} of $(\mathbf{W}^T\mathbf{W})^2$ is calculated as $s'_{ij} = \sum_{k=1}^{m} s_{ik}s_{jk}$, we can think of it as describing the similarity between the ith and jth tasks via other tasks. Hence $(\mathbf{W}^T\mathbf{W})^t$ describes the similarity between any two tasks with the help of other tasks and it can be viewed as a way to model high-order task relationships.

However, Eq. (5.3) only gives us the prior distribution on $(\mathbf{W}^T\mathbf{W})^t$ but what we actually need is the distribution on $\mathbf{W}$. In what follows, we will show how to obtain the prior $p(\mathbf{W})$ based on Eq. (5.3).

5.2.2 Computation of the Prior

For notational simplicity, let us introduce the variable $\mathbf{S}$ to denote $\mathbf{W}^T\mathbf{W}$. Thus

$$\mathbf{S}^t \sim \mathcal{W}_m(d, \mathbf{\Omega}).$$

By using the Jacobian of the transformation from $\mathbf{S}^t$ to $\mathbf{S}$, we obtain the density function for $\mathbf{S}$ as

$$p(\mathbf{S}) = \frac{|\mathbf{S}|^{\frac{t(d-m-1)}{2}} \exp\left\{-\frac{1}{2}\mathrm{tr}[\mathbf{\Omega}^{-1}\mathbf{S}^t]\right\}}{2^{md/2}\,\Gamma_m(\frac{d}{2})\,|\mathbf{\Omega}|^{d/2}} J(\mathbf{S}^t \to \mathbf{S}).$$

where $J(\mathbf{A} \to \mathbf{B})$ denotes the Jacobian of the transformation from variable $\mathbf{A}$ to $\mathbf{B}$. The following lemma from (Mathai, 1997) tells us how to compute the Jacobian $J(\mathbf{S}^t \to \mathbf{S})$ needed for the computation of $p(\mathbf{S})$.

Lemma 5.1 *For an $m \times m$ square matrix $\mathbf{S}$, we have*

$$J(\mathbf{S}^t \to \mathbf{S}) = \prod_{i,j=1}^{m} F(\lambda_i, \lambda_j, t), \tag{5.4}$$

where λ_i is the ith largest eigenvalue of $\mathbf{S}$ and $F(\lambda_i, \lambda_j, t) = \sum_{k=0}^{t-1} \lambda_i^k \lambda_j^{t-1-k}$.

With this result, we can now compute $p(\mathbf{S})$ as

$$p(\mathbf{S}) = \frac{|\mathbf{S}|^{\frac{t(d-m-1)}{2}} \exp\left\{-\frac{1}{2}\text{tr}[\mathbf{\Omega}^{-1}\mathbf{S}^t]\right\} \prod_{i,j=1}^{m} F(\lambda_i, \lambda_j, t)}{2^{md/2}\, \Gamma_m(\frac{d}{2})\, |\mathbf{\Omega}|^{d/2}}. \tag{5.5}$$

Based on $p(\mathbf{S})$ in Eq. (5.5), Theorem 5.1 below allows us to compute the density function of $\mathbf{W}$.

Theorem 5.1 *If a random matrix* $\mathbf{S}$ *has probability density function as expressed in Eq. (5.5) and* $\mathbf{S} = \mathbf{W}^T\mathbf{W}$ *with* $\mathbf{W} \in \mathbb{R}^{d\times m}$*, the probability density function of* $\mathbf{W}$ *can be computed as*

$$p(\mathbf{W}) = \frac{|\mathbf{W}^T\mathbf{W}|^{\frac{(t-1)\tilde{d}}{2}} \exp\left\{-\frac{1}{2}\text{tr}[\mathbf{\Omega}^{-1}(\mathbf{W}^T\mathbf{W})^t]\right\} \prod_{i,j} F(\lambda_i, \lambda_j, t)}{(2\pi)^{md/2}\, |\mathbf{\Omega}|^{d/2}}, \tag{5.6}$$

where $\tilde{d} = d - m - 1$ *and* λ_i*, for* $i = 1, \ldots, m$*, is the* i*th largest eigenvalue of* $\mathbf{W}^T\mathbf{W}$ *or, equivalently, the* i*th largest squared singular value of* $\mathbf{W}$*.*

Proof: Since $\mathbf{S}$ is symmetric and positive definite, we can express it in terms of the (unique) Cholesky decomposition as $\mathbf{S} = \mathbf{C}\mathbf{C}^T$, where $\mathbf{C}$ is a lower triangular matrix with positive diagonal elements. We define an independent random matrix $\mathbf{L} \in \mathbb{R}^{m\times d}$ such that $\mathbf{L}\mathbf{L}^T = \mathbf{I}_m$, with density given by $\frac{\Gamma_m(\frac{d}{2})}{2^m \pi^{md/2}} g_{d,m}(\mathbf{L})$ where $g_{d,m}(\mathbf{L})$ is as defined in Eq. (1.3.26) in (Gupta & Nagar, 2000). Then the joint density of $\mathbf{S}$ and $\mathbf{L}$ is

$$p(\mathbf{S}, \mathbf{L}) = \frac{|\mathbf{S}|^{\frac{t\tilde{d}}{2}} \exp\left\{-\frac{1}{2}\text{tr}[\mathbf{\Omega}^{-1}\mathbf{S}^t]\right\} g_{d,m}(\mathbf{L}) \prod_{i,j} F(\lambda_i, \lambda_j, t)}{(2\pi)^{md/2}\, 2^m\, |\mathbf{\Omega}|^{d/2}}.$$

Let us define $\tilde{\mathbf{W}} = \mathbf{L}^T\mathbf{C}^T$. It is easy to show that $\tilde{\mathbf{W}}^T\tilde{\mathbf{W}} = \mathbf{S}$. In order to compute the density of $\tilde{\mathbf{W}}$, we first calculate the Jacobian of the transformation from $\mathbf{S}$ to $\tilde{\mathbf{W}}$ as

$$\begin{aligned}
J(\mathbf{S} \to \tilde{\mathbf{W}}) =& J(\mathbf{S} \to \mathbf{C}) J(\mathbf{C}, \mathbf{L} \to \tilde{\mathbf{W}}) \text{ (chain rule)} \\
=& \frac{J(\mathbf{S} \to \mathbf{C})}{J(\tilde{\mathbf{W}} \to \mathbf{C}, \mathbf{L})} \text{ (property of Jacobian transformation)} \\
=& \frac{2^m \prod_{i=1}^{m} c_{ii}^{m-i+1}}{g_{d,m}(\mathbf{L}) \prod_{i=1}^{m} c_{ii}^{d-i}} \text{ (property of Jacobian transformation)} \\
=& \frac{2^m}{g_{d,m}(\mathbf{L})\, |\tilde{\mathbf{W}}^T\tilde{\mathbf{W}}|^{\frac{\tilde{d}}{2}}}, \text{ (property of Cholesky decomposition)}
\end{aligned}$$

where c_{ij} is the (i, j)th element of $\mathbf{C}$. We can then get the density function of $\tilde{\mathbf{W}}$ as

$$p(\tilde{\mathbf{W}}) = \frac{|\tilde{\mathbf{W}}^T\tilde{\mathbf{W}}|^{\frac{(t-1)\tilde{d}}{2}} \exp\left\{-\frac{1}{2}\text{tr}[\mathbf{\Omega}^{-1}(\tilde{\mathbf{W}}^T\tilde{\mathbf{W}})^t]\right\} \prod_{i,j} F(\lambda_i, \lambda_j, t)}{(2\pi)^{md/2}\, |\mathbf{\Omega}|^{d/2}}.$$

It is easy to show that $\tilde{\mathbf{W}} = \mathbf{PW}$ for some orthogonal matrix $\mathbf{P} \in \mathbb{R}^{d\times d}$. Because $J(\tilde{\mathbf{W}} \rightarrow \mathbf{W}) = |\mathbf{P}|^d = 1$, we can get the density function of $\mathbf{W}$ in Eq. (5.6). Since λ_i is the ith largest eigenvalue of $\mathbf{S} = \mathbf{W}^T\mathbf{W}$, λ_i is also the ith largest squared singular value of $\mathbf{W}$. This completes the proof. ■

When $t = 1$, the density function in Eq. (5.6) will degenerate to the matrix variate normal distribution used in the MTRL model (Zhang & Yeung, 2010a) since the first term in the numerator of Eq. (5.6) disappears and $F(\lambda_i, \lambda_j, t) = 1$. Hence the prior defined in Eq. (5.6) can be viewed as a generalization of the matrix variate normal prior.

5.2.3 Properties of the New Prior

We briefly present here some properties of the prior on $\mathbf{W}$ with the general form of the density function given in Eq. (5.6).

Theorem 5.2 $\mathbb{E}[\mathbf{W}] = \mathbf{0}$*, where* $\mathbb{E}[\cdot]$ *denotes the expectation operator.*

Proof: From the definition of expectation, $\mathbb{E}[\mathbf{W}] = \int \mathbf{W}p(\mathbf{W})d\mathbf{W}$. It is easy to show that $p(\mathbf{W}) = p(-\mathbf{W})$ implying $\mathbf{W}p(\mathbf{W}) + (-\mathbf{W})p(-\mathbf{W}) = \mathbf{0}$. So we can obtain

$$\mathbb{E}[\mathbf{W}] = \int \mathbf{W}p(\mathbf{W})d\mathbf{W} = \mathbf{0}$$

which is the conclusion. □

We may generalize the density of $\mathbf{W}$ in Eq. (5.6) by replacing $\mathbf{W}$ with $(\mathbf{W} - \mathbf{M})$ for any $\mathbf{M} \in \mathbb{R}^{d\times m}$ and then we can easily get $\mathbb{E}[\mathbf{W}] = \mathbf{M}$.

As a consequence of using the matrix variate normal prior in MTRL, we note that the different rows of $\mathbf{W}$ under the prior in Eq. (5.1) are independent due to the fact that $p(\mathbf{W}) = \prod_{j=1}^{d} p(\mathbf{w}^{(j)})$ where $\mathbf{w}^{(j)}$ is the jth row of $\mathbf{W}$. So the prior in MTRL cannot capture the dependency between different features. For our method, it is easy to show that $p(\mathbf{W}) \neq \prod_{j=1}^{d} p(\mathbf{w}^{(j)})$ and hence our method can capture feature dependency to a certain extent. This capability is an extra advantage of our method over MTRL in addition to exploiting high-order task relationships.

From the properties of the Wishart distribution, we also have the following results.

Theorem 5.3 $\mathbb{E}[(\mathbf{W}^T\mathbf{W})^t] = d\boldsymbol{\Omega}$.

Theorem 5.4 $\mathrm{Mode}[(\mathbf{W}^T\mathbf{W})^t] = (d - m - 1)\boldsymbol{\Omega}$ *when* $d \geq m + 1$*, where* $\mathrm{Mode}[\cdot]$ *denotes the mode operator.*

5.2.4 Model Formulation and Learning

Eq. (5.6) defines the prior of $\mathbf{W}$. For the likelihood, we use the (univariate) normal distribution which corresponds to the squared loss:

$$y_j^i \mid \mathbf{x}_j^i, \mathbf{w}_i, \mathbf{b}_i \sim \mathcal{N}(\mathbf{w}_i^T \mathbf{x}_j^i + b_i, \sigma_i^2). \tag{5.7}$$

So this model has prior defined in Eq. (5.6) and the normal likelihood in Eq. (5.7), which coincides with the proposed framework in section 1.3.

For computational efficiency, as in MTRL, we seek to find the MAP solution as a point estimate based on the prior in Eq. (5.6) and the likelihood in Eq. (5.7). The optimization problem can equivalently be formulated as the following minimization problem:

$$\begin{aligned}\min_{\mathbf{W},\mathbf{b},\boldsymbol{\sigma},\boldsymbol{\Omega}} L = &\sum_{i=1}^{m}\sum_{j=1}^{n_i}\left[\frac{(\mathbf{w}_i^T\mathbf{x}_j^i + b_i - y_j^i)^2}{2\sigma_i^2} + \ln\sigma_i\right] + \frac{d}{2}\ln|\boldsymbol{\Omega}| + \frac{1}{2}\mathrm{tr}\left[\boldsymbol{\Omega}^{-1}(\mathbf{W}^T\mathbf{W})^t\right] \\ &- \sum_{i,j=1}^{m}\ln F(\lambda_i,\lambda_j,t) - \frac{(t-1)\tilde{d}}{2}\ln|\mathbf{W}^T\mathbf{W}|,\end{aligned} \tag{5.8}$$

where $\mathbf{b} = (\mathbf{b}_1, \ldots, \mathbf{b}_m)^T$ and $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_m)^T$. This problem is non-convex and it is not easy to optimize with respect to all variables simultaneously. We use an alternating method instead.

Each iteration of the iterative learning procedure involves two steps. In the first step, $\mathbf{W}$ is held fixed while minimizing the objective function in (5.8) with respect to $\mathbf{b}$, $\boldsymbol{\sigma}$ and $\boldsymbol{\Omega}$. We compute the gradients of L with respect to $\mathbf{b}$, $\boldsymbol{\sigma}$ and $\boldsymbol{\Omega}^{-1}$ as

$$\begin{aligned}\frac{\partial L}{\partial \sigma_i} &= \sum_{j=1}^{n_i}\left[-\frac{(\mathbf{w}_i^T\mathbf{x}_j^i + b_i - y_j^i)^2}{\sigma_i^3} + \frac{1}{\sigma_i}\right] \\ \frac{\partial L}{\partial b_i} &= \sum_{j=1}^{n_i}\frac{b_i + \mathbf{w}_i^T\mathbf{x}_j^i - y_j^i}{\sigma_i^2} \\ \frac{\partial L}{\partial \boldsymbol{\Omega}^{-1}} &= \Big((\mathbf{W}^T\mathbf{W})^t - d\boldsymbol{\Omega}\Big) - \frac{1}{2}\Big((\mathbf{W}^T\mathbf{W})^t - d\boldsymbol{\Omega}\Big) \odot \mathbf{I}_m,\end{aligned}$$

where $\odot$ denotes the Hadamard product which is the matrix elementwise product. We then set the gradients to 0 to obtain the analytical solution for making the update:

$$\begin{aligned}\sigma_i^2 &= \frac{1}{n_i}\sum_{j=1}^{n_i}(\mathbf{w}_i^T\mathbf{x}_j^i + b_i - y_j^i)^2 \\ b_i &= \frac{1}{n_i}\sum_{j=1}^{n_i}(y_j^i - \mathbf{w}_i^T\mathbf{x}_j^i) \\ \boldsymbol{\Omega} &= \frac{1}{d}(\mathbf{W}^T\mathbf{W})^t.\end{aligned}$$

In the second step of each iteration, optimization is done with respect to $\mathbf{W}$ while keeping all other variables fixed. Since there is no analytical solution for $\mathbf{W}$, we use the conjugate gradient method to find the solution using the following gradient

$$\frac{\partial L}{\partial \mathbf{W}} = \sum_{i=1}^{m}\sum_{j=1}^{n_i} \frac{\mathbf{x}_j^i(\mathbf{x}_j^i)^T\mathbf{W}\mathbf{e}_i\mathbf{e}_i^T + (b_i - y_j^i)\mathbf{x}_j^i\mathbf{e}_i^T}{\sigma_i^2} - \sum_{i,j=1}^{m} \frac{\partial \ln F(\lambda_i, \lambda_j, t)}{\partial \mathbf{W}} + \sum_{k=0}^{t-1} \mathbf{W}(\mathbf{W}^T\mathbf{W})^{t-k-1}\mathbf{\Omega}^{-1}(\mathbf{W}^T\mathbf{W})^k - \tilde{d}(t-1)\mathbf{W}(\mathbf{W}^T\mathbf{W})^{-1},$$

where $\mathbf{e}_i$ is the ith column of the $m \times m$ identity matrix. The term $\frac{\partial \ln F(\lambda_i,\lambda_j,t)}{\partial \mathbf{W}}$ can be calculated as

$$\frac{\partial \ln F(\lambda_i, \lambda_j, t)}{\partial \mathbf{W}} = \begin{cases} \frac{t-1}{\lambda_i}\frac{\partial \lambda_i}{\partial \mathbf{W}} & \text{if } \lambda_i = \lambda_j \\ \frac{t\left(\lambda_i^{t-1}\frac{\partial \lambda_i}{\partial \mathbf{W}} - \lambda_j^{t-1}\frac{\partial \lambda_j}{\partial \mathbf{W}}\right)}{\lambda_i^t - \lambda_j^t} - \frac{\frac{\partial \lambda_i}{\partial \mathbf{W}} - \frac{\partial \lambda_j}{\partial \mathbf{W}}}{\lambda_i - \lambda_j} & \text{otherwise} \end{cases}$$

where, by using the chain rule, we can calculate $\frac{\partial \lambda_i}{\mathbf{W}}$ as

$$\frac{\partial \lambda_i}{\partial \mathbf{W}} = \frac{\partial \lambda_i}{\partial \rho_i}\frac{\partial \rho_i}{\partial \mathbf{W}} = \frac{\partial \rho_i^2}{\partial \rho_i}\frac{\partial \frac{1}{\alpha_i}\mathrm{tr}(\mathbf{U}_i^T\mathbf{W}\mathbf{V}_i)}{\partial \mathbf{W}} = \frac{2\rho_i}{\alpha_i}\mathbf{U}_i\mathbf{V}_i^T, \tag{5.9}$$

where $\mathbf{W} = \mathbf{U}\mathbf{D}\mathbf{V}^T$ is the thin singular value decomposition of $\mathbf{W}$ (since $d \geq m$) with orthogonal matrices $\mathbf{U} \in \mathbb{R}^{d\times m}$ and $\mathbf{V} \in \mathbb{R}^{m\times m}$ and the diagonal matrix $\mathbf{D}$ containing the singular values ρ_i, α_i is the repeatedness of ρ_i, and $\mathbf{U}_i \in \mathbb{R}^{d\times\alpha_i}$ and $\mathbf{V}_i \in \mathbb{R}^{m\times\alpha_i}$ are the left and right singular vectors or matrices corresponding to the singular value ρ_i. The second step of the above derivation (5.9) holds since $\mathbf{U}_i^T\mathbf{W}\mathbf{V}_i = \rho_i\mathbf{I}_{\alpha_i}$ and $\lambda_i = \rho_i^2$ due to the relation between the ith largest eigenvalue λ_i of $\mathbf{W}^T\mathbf{W}$ and the ith largest singular value of $\mathbf{W}$.

5.2.5 Kernel Extension

A natural question to ask is whether the method presented above can be extended to the general nonlinear case using the kernel trick.

Unlike MTRL (Zhang & Yeung, 2010a) and other kernel methods such as the support vector machine, using the higher-order form of $\mathbf{W}$ in our method does not allow us to use the dual form to facilitate kernel extension. However, kernel extension is still possible by adopting a different strategy which was used before by some Bayesian kernel methods such as the relevance vector machine (Tipping, 2001).

Specifically, the kernel information is directly incorporated into the data representation by defining a new data representation as follows:

$$\tilde{\mathbf{x}}_j^i = \left(k(\mathbf{x}_j^i, \mathbf{x}_1^1), \ldots, k(\mathbf{x}_j^i, \mathbf{x}_{n_m}^m)\right)^T,$$

where $k(\cdot, \cdot)$ is some kernel function such as the linear kernel or RBF kernel. With the new representation, we can use the method presented in the previous section to learn a good model. It is easy to show that $\tilde{\mathbf{x}}_j^i \in \mathbb{R}^n$ where n is the total number of data points in all tasks. As a consequence, another benefit of this kernel extension scheme is that the dimensionality n of each data point is larger than the number of tasks m and hence satisfies the requirement of our new prior and the model. In case the dimensionality n of the new data representation is very high, we may first apply a dimensionality reduction method such as principal component analysis to find a lower-dimensional representation before model learning.

5.3 Experiments

We report empirical studies in this section to compare MTHOL with several related methods. Since MTHOL generalizes MTRL in Chapter 3 by learning high-order task relationships, MTRL will be included in the comparative study. Moreover, we have also included the other methods considered in the comparative study of the MTRL paper, including a single-task learning (STL) method, multi-task feature learning (MTFL) method (Argyriou et al., 2006) as well as multi-task GP (MTGP) method (Bonilla et al., 2007) which, like MTRL, can learn task relationships. For fair comparison, the data we used for all methods adopt the new representation described in section 5.2.5.

5.3.1 Robot Inverse Dynamics

The description for this application as well as the data is depicted in section 3.4.2. We use the normalized mean squared error (nMSE), the mean squared error divided by the variance of the ground-truth output, as the performance measure. Kernel ridge regression is used as the STL baseline for this data set.

We perform two sets of experiments on all the methods with different numbers of training data points. The first setting randomly selects 10% of the data for each task as the training set and the second setting selects 20%. Those not selected for training are used for testing. For all methods, the RBF kernel is adopted. For each setting, we report the mean and standard deviation of the performance measure over 10 random splits of the data. Table 5.1 depicts the results, with the best ones shown in bold. Paired t-test at 5% significance level shows that MTHOL with $t = 2$ is the best for some tasks and is comparable with the best for other tasks. Comparing the performance of MTHOL under these two configurations, the performance becomes better when the size of training data increases.

To see whether MTHOL is sensitive to the degree t, we compare MTHOL with $t = 2$ and $t = 3$. The results are recorded in Table 5.2. For all tasks, we can see that the two variants

Table 5.1: Results on learning robot inverse dynamics. For each method and each task, the two rows report the mean and standard deviation of the nMSE over 10 trials. The upper (lower) table records the results of the setting with 10% (20%) of the data for the training set.

Method	1st	2nd	3rd	4th	5th	6th	7th
STL	0.1507	0.1920	0.1866	0.0283	0.3563	0.6454	0.0409
	0.0077	0.0087	0.0054	0.0038	0.0246	0.0202	0.0016
MTFL	0.1404	0.1953	0.1823	0.0221	0.3294	0.6497	0.0470
	0.0056	0.0079	0.0195	0.0127	0.0139	0.0118	0.0044
MTGP	0.0926	0.1471	0.1084	0.0116	0.3465	0.6833	0.0438
	0.0035	0.0076	0.0067	0.0013	0.0070	0.0109	0.0190
MTRL	0.0879	0.1358	**0.1030**	0.0098	0.3248	0.6223	**0.0387**
	0.0008	0.0038	0.0034	0.0017	0.0054	0.0034	0.0073
MTHOL	**0.0706**	**0.1242**	**0.1035**	**0.0082**	**0.3158**	**0.6032**	**0.0372**
(t=2)	0.0019	0.0048	0.0050	0.0015	0.0096	0.0094	0.0042

Method	1st	2nd	3rd	4th	5th	6th	7th
STL	0.1140	0.1714	0.1537	0.0141	0.2805	0.5501	0.0298
	0.0027	0.0032	0.0046	0.0005	0.0062	0.0161	0.0018
MTFL	0.1131	0.1800	0.1516	0.0187	0.2832	0.5596	0.0371
	0.0029	0.0050	0.0045	0.0014	0.0081	0.0142	0.0034
MTGP	0.0813	0.1208	**0.0893**	**0.0088**	0.2982	0.5453	0.0346
	0.0023	0.0058	0.0048	0.0004	0.0093	0.0252	0.0134
MTRL	0.0812	0.1122	0.0947	**0.0087**	0.2712	0.5429	0.0282
	0.0021	0.0032	0.0028	0.0013	0.0031	0.0060	0.0007
MTHOL	**0.0782**	**0.0980**	**0.0898**	**0.0078**	**0.1333**	**0.5243**	**0.0200**
(t=2)	0.0029	0.0034	0.0041	0.0016	0.0050	0.0077	0.0027

give very similar results. Thus we always set t to 2 in all the following experiments without performing further tuning.

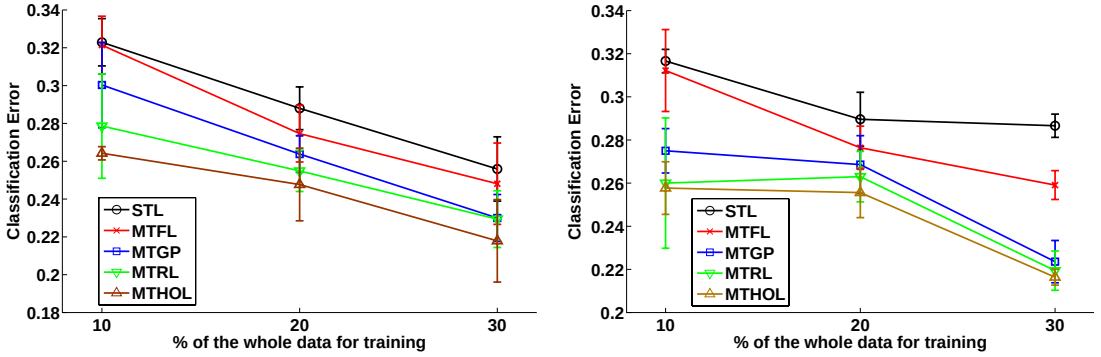
Table 5.2: Comparison of MTHOL with $t = 2$ and $t = 3$ on learning robot inverse dynamics. The two tables correspond to the two settings as in Table 5.1.

Method	1st	2nd	3rd	4th	5th	6th	7th
MTHOL	0.0706	0.1242	0.1035	0.0082	0.3158	0.6032	0.0372
($t = 2$)	0.0019	0.0048	0.0050	0.0015	0.0096	0.0094	0.0042
MTHOL	0.0708	0.1241	0.1038	0.0081	0.3150	0.6029	0.0380
($t = 3$)	0.0018	0.0049	0.0052	0.0015	0.0093	0.0089	0.0046

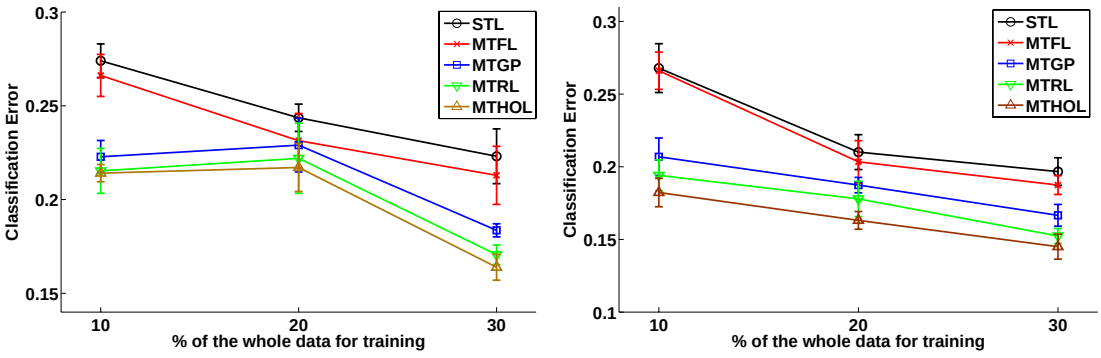
Method	1st	2nd	3rd	4th	5th	6th	7th
MTHOL	0.0782	0.0980	0.0898	0.0078	0.1333	0.5243	0.0200
($t = 2$)	0.0029	0.0034	0.0041	0.0016	0.0050	0.0077	0.0027
MTHOL	0.0778	0.0984	0.0893	0.0074	0.1329	0.5240	0.0209
($t = 3$)	0.0031	0.0030	0.0048	0.0016	0.0045	0.0078	0.0022

5.3.2 Multi-Domain Sentiment Application

The description for this application is record in section 3.4.3. Like the experiments for the first data set, we use training sets of different sizes for training, corresponding to 10%, 20% and 30% of the data for each task. Classification error is used as the performance measure. For the STL baseline, we use a linear SVM which has been demonstrated to perform well for such text classification applications with high feature dimensionality. For other compared methods, we also use the linear kernel. As above, we perform 10 random data splits and report the mean and standard deviation of the classification error from Figure 5.1(a) to Figure 5.1(d). Again, for all settings, MTHOL is either the best or among the best.



(a) 1st task (b) 2nd task



(c) 3rd task (d) 4th task

Figure 5.1: Performance of STL, MTFL, MTGP, MTRL and MTHOL on each task of the multi-domain sentiment application when the training set size is varied.

5.3.3 Comparison of the Three Proposed Methods

In this thesis, I have proposed three methods for multi-task learning, i.e., MTRL, MTGTP and MTHOL. In this section, I compare these three methods on the above two data sets with identical experimental settings.

The comparison results of the three methods on robot inverse dynamics and multi-domain sentiment classification applications are depicted in Table 5.3 and 5.4. From the results, we

see that the performance of the three methods is comparable, i.e., some methods perform well on some tasks while other methods have good performance on other tasks. From the perspective of computational complexity shown in Table 5.5, MTRL has the lowest complexity and MTGTP has the highest one. Making a choice among the three choices does not seem to be straightforward except perhaps for the computational concern. This is an issue that needs further investigation in the future.

Table 5.3: Comparison results of MTRL, MTGTP, and MTHOL on learning robot inverse dynamics. For each method and each task, the two rows report the mean and standard deviation of the nMSE over 10 trials. The upper (lower) table records the results of the setting with 10% (20%) of the data for the training set.

Method	1st	2nd	3rd	4th	5th	6th	7th
MTRL	0.0879	0.1358	**0.1030**	0.0098	0.3248	0.6223	0.0387
	0.0008	0.0038	0.0034	0.0017	0.0054	0.0034	0.0073
MTGTP	0.0946	0.1529	**0.1005**	0.0095	**0.2621**	**0.5421**	**0.0202**
	0.0053	0.0164	0.0052	0.0012	0.0088	0.0156	0.0021
MTHOL	**0.0706**	**0.1242**	**0.1035**	**0.0082**	0.3158	0.6032	0.0372
	0.0019	0.0048	0.0050	0.0015	0.0096	0.0094	0.0042

Method	1st	2nd	3rd	4th	5th	6th	7th
MTRL	0.0812	0.1122	0.0947	0.0087	0.2712	0.5429	0.0282
	0.0021	0.0032	0.0028	0.0013	0.0031	0.0060	0.0007
MTGTP	**0.0754**	0.1121	**0.0737**	**0.0058**	**0.0850**	**0.3912**	**0.0102**
	0.0039	0.0050	0.0106	0.0007	0.0140	0.0185	0.0007
MTHOL	**0.0782**	**0.0980**	**0.0898**	0.0078	0.1333	0.5243	0.0200
	0.0029	0.0034	0.0041	0.0016	0.0050	0.0077	0.0027

5.4 Concluding Remarks

In this chapter, we have proposed a novel generalization of the MTRL method in Chapter 3 by modeling and learning high-order task relationships. The model parameters as well as the task relationships can be learned in a principal framework.

Table 5.4: Comparison results of MTRL, MTGTP, and MTHOL on multi-domain sentiment classification. The three tables are for different training set sizes (10%, 20%, 30%).

Method	1st	2nd	3rd	4th
MTRL	0.2786	**0.2600**	**0.2153**	0.1941
	0.0276	0.0302	0.0120	0.0107
MTGTP	**0.2634**	**0.2594**	**0.2169**	0.2058
	0.0160	0.0157	0.0171	0.0165
MTHOL	**0.2642**	**0.2577**	**0.2141**	**0.1823**
	0.0035	0.0122	0.0146	0.0158

Method	1st	2nd	3rd	4th
MTRL	0.2549	0.2630	**0.2220**	0.1779
	0.0109	0.0117	0.0187	0.0122
MTGTP	0.2516	**0.2424**	**0.2042**	0.1849
	0.0112	0.0098	0.0104	0.0065
MTHOL	**0.2477**	0.2556	**0.2171**	**0.1631**
	0.0192	0.0116	0.0128	0.0161

Method	1st	2nd	3rd	4th
MTRL	0.2294	0.2194	0.1706	0.1524
	0.0150	0.0091	0.0052	0.0052
MTGTP	**0.2034**	**0.1913**	0.1881	0.1677
	0.0147	0.0046	0.0153	0.0054
MTHOL	**0.2179**	0.2164	**0.1639**	**0.1450**
	0.0218	0.0037	0.0069	0.0086

Table 5.5: Comparison of the computational complexity of the three proposed methods.

Method	Complexity
MTRL	$O(n^3)$ or $O(n^2)$ (for approximation)
MTGTP	$O(n^3)$
MTHOL	$O(n^2m)$

CHAPTER 6

PROBABILISTIC MULTI-TASK FEATURE SELECTION

6.1 Introduction

Different from the above chapters which discuss the standard multi-task classification or regression problems, in this chapter, we study the problem of learning task relationships in the scenario of multi-task feature selection which aims to identify the common useful features for all tasks.

Learning algorithms based on l_1 regularization have a long history in machine learning and statistics. A well-known property of l_1 regularization is its ability to enforce sparsity into the solutions. Recently, some variants of the l_1 norm, particularly matrix norms such as the $l_{1,2}$ and $l_{1,\infty}$ norms, were proposed to enforce sparsity via joint regularization (Turlach et al., 2005; Obozinski et al., 2006; Xiong et al., 2007; Argyriou et al., 2008a; Bi et al., 2008; Liu et al., 2009a; Quattoni et al., 2009; Liu et al., 2009b; Obozinski1 et al., 2010). The $l_{1,2}$ norm is the sum of the l_2 norms of the rows and the $l_{1,\infty}$ norm is the sum of the l_∞ norms of the rows. Regularizers based on these two matrix norms encourage row sparsity, that is, they encourage entire rows of the matrix to have zero elements. Moreover, these norms have also been used for enforcing group sparsity among features in conventional classification and regression problems, for example, group LASSO (Yuan & Lin, 2006). Recently, they have been widely used in multi-task learning, compressed sensing and other related areas. However, when given a specific application, we often have no idea which norm is the most appropriate choice to use. Moreover, an underlying assumption of multi-task feature selection using the $l_{1,q}$ norm is that all tasks are similar to each other and they share the same features. This assumption may not be correct in practice because there may exist outlier tasks (that is, tasks that are not related to all other tasks) or tasks with negative correlation (that is, tasks that are negatively correlated with some other tasks). Similar to the proposed methods in this thesis, a better way is to learn the task relationships from data which is not restricted to modeling positive correlations between tasks.

In this chapter, we study the problem of determining the most appropriate sparsity enforcing norm to use in the context of multi-task feature selection as well as learning the task relationships. Instead of choosing between specific choices such as the $l_{1,2}$ and $l_{1,\infty}$ norms, we consider a family of $l_{1,q}$ norms. We restrict q to the range $1 < q \leq \infty$ to ensure that all norms in this family are convex, making it easier to solve the optimization problem formulated based on it.

Within this family, the $l_{1,2}$ and $l_{1,\infty}$ norms are just two special cases. Using the $l_{1,q}$ norm, we formulate the general multi-task feature selection problem and give it a probabilistic interpretation. It is noted that the automatic relevance determination (ARD) prior (Figueiredo, 2003; Bishop, 2006; Wipf & Nagarajan, 2007) comes as a special case under this interpretation. Based on this probabilistic interpretation, we develop a probabilistic formulation using a noninformative prior called the Jeffreys prior (Gelman et al., 2003). We devise an expectation-maximization (EM) algorithm (Dempster et al., 1977) to learn all model parameters, including q, automatically. As another contribution of this work, we propose to use a matrix variate generalized normal prior (Gupta & Varga, 1995) for the model parameters to learn the relationships between tasks. The task relationships learned here can be seen as an extension of the task covariance used in (Bonilla et al., 2007; Zhang & Yeung, 2010b, 2010a). Experiments are reported on two cancer classification applications using microarray gene expression data.

6.2 Multi-Task Feature Selection

Suppose we are given m learning tasks $\{T_i\}_{i=1}^m$. For the ith task T_i, the training set $\mathcal{D}_i$ consists of n_i labeled data points in the form of ordered pairs $(\mathbf{x}_j^i, y_j^i)$, $j = 1, \ldots, n_i$, with $\mathbf{x}_j^i \in \mathbb{R}^d$ and its corresponding output $y_j^i \in \mathbb{R}$ if it is a regression problem and $y_j^i \in \{-1, 1\}$ if it is a binary classification problem. The linear function for T_i is defined as $f_i(\mathbf{x}) = \mathbf{w}_i^T\mathbf{x} + b_i$. For applications that need feature selection, e.g., document classification, the feature dimensionality is usually very high and it has been found that linear methods usually perform better.

The objective functions of most existing multi-task feature selection methods (Turlach et al., 2005; Obozinski et al., 2006; Xiong et al., 2007; Argyriou et al., 2008a; Bi et al., 2008; Liu et al., 2009a; Quattoni et al., 2009; Liu et al., 2009b; Obozinski1 et al., 2010) can be expressed in the following form:

$$\sum_{i=1}^{m}\sum_{j=1}^{n_i} L(y_j^i, \mathbf{w}_i^T\mathbf{x}_j^i + b_i) + \lambda R(\mathbf{W}), \tag{6.1}$$

where $\mathbf{W} = (\mathbf{w}_1, \ldots, \mathbf{w}_m)$, $L(\cdot, \cdot)$ denotes the loss function (e.g., squared loss for regression and hinge loss for classification), $R(\cdot)$ is the regularization function that enforces feature sparsity under the multi-task setting, and λ is the regularization parameter controlling the relative contribution of the empirical loss and the regularizer. Multi-task feature selection seeks to minimize the objective function above to obtain the optimal parameters $\{\mathbf{w}_i, b_i\}$. Two regularization functions are widely used in existing multi-task feature selection methods. One of them is based on the $l_{1,2}$ norm of $\mathbf{W}$ (Obozinski et al., 2006; Xiong et al., 2007; Argyriou et al., 2008a; Bi et al., 2008; Liu et al., 2009b; Obozinski1 et al., 2010):

$$R(\mathbf{W}) = \sum_{k=1}^{d} \|\mathbf{w}^k\|_2,$$

where $\|\cdot\|_q$ denotes the q-norm (or l_q norm) of a vector and $\mathbf{w}^k$ denotes the kth row of $\mathbf{W}$. Another one is based on the $l_{1,\infty}$ norm of $\mathbf{W}$ (Turlach et al., 2005; Liu et al., 2009a; Quattoni et al., 2009): $R(\mathbf{W}) = \sum_{k=1}^{d} \|\mathbf{w}^k\|_\infty$.

Here we unify these two cases by using the $l_{1,q}$ norm of $\mathbf{W}$ to define a more general regularization function:

$$R(\mathbf{W}) = \sum_{k=1}^{d} \|\mathbf{w}^k\|_q, \quad 1 < q \leq \infty.$$

Note that when $q < 1$, $R(\mathbf{W})$ is non-convex with respect to $\mathbf{W}$. Although $R(\mathbf{W})$ is convex when $q = 1$, each element of $\mathbf{W}$ is independent of each other and so the regularization function cannot enforce feature sparsity. Thus we restrict the range to $1 < q \leq \infty$.

Even though restricting the range to $1 < q \leq \infty$ can enforce feature sparsity between different tasks, different values of q imply different 'group discounts' for sharing the same feature. Specifically, when q approaches 1, the cost grows almost linearly with the number of tasks that use a feature, and when $q = \infty$, only the most demanding task matters. So selecting a proper q can potentially have a significant effect on the performance of the learning algorithms.

In the following, we first give a probabilistic interpretation for multi-task feature selection methods. Based on this probabilistic interpretation, we then develop a probabilistic model which, among other things, can solve the model selection problem automatically by estimating q from data.

6.3 Probabilistic Interpretation

In this section, we will show that existing multi-task feature selection methods are related to the MAP solution of a probabilistic model. This probabilistic interpretation sets the stage for introducing our probabilistic model in the next section.

We first introduce the generalized normal distribution (Goodman & Kotz, 1973) which is useful for the model to be introduced.

Definition 6.1 *z is a univariate generalized normal random variable iff its probability density function (p.d.f.) is given as follows:*

$$p(z) = \frac{1}{2\rho\Gamma(1+\frac{1}{q})} \exp\Big(-\frac{|z-\mu|^q}{\rho^q}\Big).$$

For simplicity, if z is a univariate generalized normal random variable, we write $z \sim \mathcal{GN}(\mu, \rho, q)$. The (ordinary) normal distribution can be viewed as a special case of the generalized normal distribution when $q = 2$ and the Laplace distribution is a special case when $q = 1$. When q

approaches $+\infty$, the generalized normal distribution approaches the uniform distribution in the range $[\mu-\rho, \mu+\rho]$. The generalized normal distribution has proven useful in Bayesian analysis and robustness studies.

Definition 6.2 *A standardized $r \times 1$ multivariate generalized normal random variable $\mathbf{z} = (z_1, \ldots, z_r)^T$ consists of r independent and identically distributed (i.i.d.) univariate generalized normal random variables.*

If $\mathbf{z}$ is a standardized $r \times 1$ multivariate generalized normal random variable, we write $\mathbf{z} \sim \mathcal{MGN}(\mu, \rho, q)$ with the following p.d.f.:

$$p(\mathbf{z}) = \frac{1}{\left[2\rho\Gamma(1+\frac{1}{q})\right]^r} \exp\Big(-\frac{\sum_{i=1}^r |z_i - \mu|^q}{\rho^q}\Big).$$

With these definitions, we now begin to present our probabilistic interpretation for multi-task feature selection by proposing a probabilistic model. For notational simplicity, we assume that all tasks perform regression. Extension to include classification tasks will go through similar derivation.

For a regression problem, we use the normal distribution to define the likelihood for $\mathbf{x}_j^i$:

$$y_j^i \sim \mathcal{N}(\mathbf{w}_i^T \mathbf{x}_j^i + b_i, \sigma^2). \tag{6.2}$$

We impose the generalized normal prior on each element of $\mathbf{W}$:

$$w_{ij} \sim \mathcal{GN}(0, \rho_i, q), \tag{6.3}$$

where w_{ij} is the (i,j)th element of $\mathbf{W}$ (or, equivalently, the ith element of $\mathbf{w}_j$ or the jth element of $\mathbf{w}^i$). Then we can express the prior on $\mathbf{w}^i$ as

$$(\mathbf{w}^i)^T \sim \mathcal{MGN}(0, \rho_i, q).$$

When $q = 2$, this becomes the ARD prior (Figueiredo, 2003; Bishop, 2006; Wipf & Nagarajan, 2007) commonly used in Bayesian methods for enforcing sparsity. From this view, the generalized normal prior can be viewed as a generalization of the ARD prior.

With the above likelihood and prior, we can obtain the MAP solution of $\mathbf{W}$ by solving the following problem:

$$\min_{\mathbf{W},\mathbf{b},\boldsymbol{\rho}} J = \frac{1}{\sigma^2} \sum_{i=1}^m \sum_{j=1}^{n_i} L(y_j^i, \mathbf{w}_i^T \mathbf{x}_j^i + b_i) + \sum_{i=1}^d \Big(\frac{\|\mathbf{w}^i\|_q^q}{\rho_i^q} + m \ln \rho_i\Big), \tag{6.4}$$

where $\mathbf{b} = (b_1, \ldots, b_m)^T$ and $\boldsymbol{\rho} = (\rho_1, \ldots, \rho_m)^T$.

We set the derivative of J with respect to ρ_i to zero and get

$$\rho_i = \left(\frac{q}{m}\right)^{1/q} \|\mathbf{w}^i\|_q.$$

Plugging this into problem (6.4), the optimization problem can be reformulated as

$$\min_{\mathbf{W},\mathbf{b}} J = \frac{1}{\sigma^2} \sum_{i=1}^{m} \sum_{j=1}^{n_i} L(y_j^i, \mathbf{w}_i^T \mathbf{x}_j^i + b_i) + m \sum_{i=1}^{d} \ln \|\mathbf{w}^i\|_q. \tag{6.5}$$

Note that problem (6.5) is non-convex since the second term is non-convex with respect to $\mathbf{W}$. Because $\ln z \leq z - 1$ for any $z > 0$, problem (6.5) can be relaxed to problem (6.1) by setting $\lambda = m\sigma^2$. So the solutions of multi-task feature selection methods can be viewed as the solution of the relaxed optimization problem above. In many previous works such as (Candès et al., 2008; Wipf & Nagarajan, 2010), $\ln(x)$ can be used as an approximation of $I(x \neq 0)$ where $I(\cdot)$ is an indicator function. Using this view, we can regard the second term in problem (6.5) as an approximation of the number of rows with nonzero q-norms.

Note that we can directly solve problem (6.5) using a majorization-minimization (MM) algorithm (Lange et al., 2000). For numerical stability, we can slightly modify the objective function in problem (6.5) by replacing the second term with $m \sum_{i=1}^{d} \ln(\|\mathbf{w}^i\|_q + \alpha)$ where α can be regarded as a regularization parameter. We denote the solution obtained in the kth iteration as $\mathbf{w}_{(k)}^i$. In the $(k+1)$th iteration, due to the concavity property of $\ln(\cdot)$, we can bound the second term in problem (6.5) as follows

$$\sum_{i=1}^{d} \ln(\|\mathbf{w}^i\|_q + \alpha) \leq \sum_{i=1}^{d} \left[\ln(\|\mathbf{w}_{(k)}^i\|_q + \alpha) + \frac{\|\mathbf{w}^i\|_q - \|\mathbf{w}_{(k)}^i\|_q}{\|\mathbf{w}_{(k)}^i\|_q + \alpha} \right].$$

Thus, in the $(k+1)$th iteration, we need to solve a weighted version of problem (6.1):

$$\min_{\mathbf{W},\mathbf{b}} \frac{1}{\sigma^2} \sum_{i=1}^{m} \sum_{j=1}^{n_i} L(y_j^i, \mathbf{w}_i^T \mathbf{x}_j^i + b_i) + m \sum_{i=1}^{d} \frac{\|\mathbf{w}^i\|_q}{\|\mathbf{w}_{(k)}^i\|_q + \alpha}.$$

According to (Lange et al., 2000), the MM algorithm is guaranteed to converge to a local optimum.

6.4 A Probabilistic Framework for Multi-Task Feature Selection

In the probabilistic interpretation above, we use a type II method (Bishop, 2006) to estimate $\{\rho_i\}$ in the generalized normal prior which can be viewed as a generalization of the ARD prior.

In the ARD prior, according to (Qi et al., 2004), this approach is likely to lead to overfitting because the hyperparameters in the ARD prior are treated as points. Similar to the ARD prior, the model in the above section may overfit since $\{\rho_i\}$ are estimated via point estimation. In the following, we will present our probabilistic framework for multi-task feature selection by imposing priors on the hyperparameters.

6.4.1 The Model

As in the above section, the likelihood for $\mathbf{x}_j^i$ is also defined based on the normal distribution:

$$y_j^i \sim \mathcal{N}(\mathbf{w}_i^T \mathbf{x}_j^i + b_i, \sigma_i^2). \tag{6.6}$$

Here we use different noise variances σ_i for different tasks to make our model more flexible. The prior on $\mathbf{W}$ is also defined similarly:

$$w_{ij} \sim \mathcal{GN}(0, \rho_i, q). \tag{6.7}$$

The main difference here is that we treat ρ_i as a random variable with the noninformative Jeffreys prior[1]:

$$p(\rho_i) \propto \sqrt{I(\rho_i)} = \sqrt{\mathbb{E}_{\mathbf{w}^i|\rho_i}\Big[\Big(\frac{\partial \ln p(\mathbf{w}^i|\rho_i)}{\partial \rho_i}\Big)^2\Big]} \propto \frac{1}{\rho_i}, \tag{6.8}$$

where $I(\rho_i)$ denotes the Fisher information for ρ_i and $\mathbb{E}_\theta[\cdot]$ denotes the expectation with respect to θ. One advantage of using the Jeffreys prior is that the distribution has no hyperparameters.

6.4.2 Parameter Learning and Inference

Here we use the EM algorithm (Dempster et al., 1977) to learn the model parameters. In our model, we denote $\mathbf{\Theta} = \{\mathbf{W}, \mathbf{b}, \{\sigma_i\}, q\}$ as the model parameters and $\boldsymbol{\rho} = (\rho_1, \ldots, \rho_d)^T$ as the hidden variables.

In the E-step, we construct the so-called Q-function as the surrogate for the log-likelihood:

$$Q(\mathbf{\Theta}|\mathbf{\Theta}^{(k)}) = \int \ln p(\mathbf{\Theta}|\mathbf{y}, \boldsymbol{\rho}) p(\boldsymbol{\rho}|\mathbf{y}, \mathbf{\Theta}^{(k)}) d\boldsymbol{\rho},$$

where $\mathbf{\Theta}^{(k)}$ denotes the estimate of $\mathbf{\Theta}$ in the kth iteration and $\mathbf{y} = (y_1^1, \ldots, y_{n_m}^m)^T$. It is easy to show that

$$\begin{aligned}\ln p(\mathbf{\Theta}|\mathbf{y}, \boldsymbol{\rho}) &\propto \ln p(\mathbf{y}|\mathbf{W}, \{\sigma_i\}) + \ln p(\mathbf{W}|\boldsymbol{\rho}) \\ &\propto -\sum_{i=1}^{m}\left[\sum_{j=1}^{n_i} \frac{(y_j^i - \mathbf{w}_i^T \mathbf{x}_j^i - b_i)^2}{2\sigma_i^2} + \frac{n_i \ln \sigma_i^2}{2}\right] - \sum_{i=1}^{d} \frac{1}{\rho_i^q} \sum_{j=1}^{m} |w_{ij}|^q - md \ln \Gamma(1 + \frac{1}{q})\end{aligned}$$

[1]The detailed derivation of the noninformative Jeffreys prior is recorded in Appendix C

and $p(\boldsymbol{\rho}|\mathbf{y}, \boldsymbol{\Theta}^{(k)}) \propto \prod_{i=1}^{d} \Big(p(\rho_i)p(\mathbf{w}_{(k)}^i|\rho_i)\Big)$. We then compute $\mathbb{E}[\frac{1}{\rho_i^q}|\mathbf{y}, \boldsymbol{\Theta}^{(k)}]$ as

$$\mathbb{E}\Big[\frac{1}{\rho_i^q}|\mathbf{y}, \boldsymbol{\Theta}^{(k)}\Big] = \frac{\int_0^\infty \frac{1}{\rho_i^q} p(\rho_i)p(\mathbf{w}_{(k)}^i|\rho_i)d\rho_i}{\int_0^\infty p(\rho_i)p(\mathbf{w}_{(k)}^i|\rho_i)d\rho_i} = \frac{m}{q\|\mathbf{w}_{(k)}^i\|_q^q}.$$

So we can get

$$Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(k)}) = -\sum_{i=1}^{m}\left[\sum_{j=1}^{n_i}\frac{(y_j^i - \mathbf{w}_i^T\mathbf{x}_j^i - b_i)^2}{2\sigma_i^2} + \frac{n_i \ln \sigma_i^2}{2}\right] - \sum_{i=1}^{d}\beta_i\sum_{j=1}^{m}|w_{ij}|^q - md\ln\Gamma(1+\frac{1}{q}),$$

where $\beta_i = \frac{m}{q\|\mathbf{w}_{(k)}^i\|_q^q}$.

In the M-step, we maximize $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(k)})$ to update the estimates of $\mathbf{W}$, $\mathbf{b}$, $\{\sigma_i\}$ and q.

For the estimation of $\mathbf{W}$, we need to solve m convex optimization problems

$$\min_{\mathbf{w}_i} \; J = \beta_0\|\hat{\mathbf{y}}_i - \mathbf{X}_i^T\mathbf{w}_i\|_2^2 + \sum_{j=1}^{d}\beta_j|w_{ji}|^q, \quad i = 1, \ldots, m, \tag{6.9}$$

where $\hat{\mathbf{y}}_i = (y_1^i - b_i^{(k)}, \ldots, y_{n_i}^i - b_i^{(k)})^T$, $\mathbf{X}_i = (\mathbf{x}_1^i, \ldots, \mathbf{x}_{n_i}^i)$, and $\beta_0 = \frac{1}{2(\sigma_i^{(k)})^2}$. When $q = 2$, this becomes the conventional ridge regression problem. Here β_j is related to the sparsity of the jth row in $\mathbf{W}^{(k)}$: the more sparse the jth row in $\mathbf{W}^{(k)}$, the larger the β_j. When β_j is large, w_{ji} will be enforced to approach 0. We use a gradient method such as conjugate gradient to optimize problem (6.9). The subgradient with respect to $\mathbf{w}_i$ is

$$\frac{\partial J}{\partial \mathbf{w}_i} = 2\beta_0\big(\mathbf{X}_i\mathbf{X}_i^T\mathbf{w}_i - \mathbf{X}_i\hat{\mathbf{y}}_i\big) + q\boldsymbol{\theta},$$

where $\boldsymbol{\theta} = (\beta_1|w_{1i}|^{q-1}\text{sign}(w_{1i}), \ldots, \beta_d|w_{di}|^{q-1}\text{sign}(w_{di}))^T$ and $\text{sign}(\cdot)$ denotes the sign function.

We set the derivatives of $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(k)})$ with respect to σ_i and b_i to 0 and get

$$\begin{aligned} b_i^{(k+1)} &= \frac{1}{n_i}\sum_{j=1}^{n_i}\Big[y_j^i - (\mathbf{w}_i^{(k+1)})^T\mathbf{x}_j^i\Big] \\ \sigma_i^{(k+1)} &= \sqrt{\frac{1}{n_i}\sum_{j=1}^{n_i}\Big[y_j^i - (\mathbf{w}_i^{(k+1)})^T\mathbf{x}_j^i - b_i^{(k+1)}\Big]^2}. \end{aligned}$$

For the estimation of q, we also use a gradient method. The gradient can be calculated as

$$\frac{\partial Q}{\partial q} = -\sum_{i=1}^{d}\beta_j\sum_{j=1, w_{ij}^{(k+1)}\neq 0}^{m}\big|w_{ij}^{(k+1)}\big|^q \ln\big|w_{ij}^{(k+1)}\big| + \frac{md}{q} + \frac{md}{q^2}\psi(\frac{1}{q}),$$

where $\psi(x) \equiv \frac{\partial \ln\Gamma(x)}{\partial x}$ is the digamma function.

6.4.3 Extension to Deal with Outlier Tasks and Tasks with Negative Correlation

An underlying assumption of multi-task feature selection using the $l_{1,q}$ norm is that all tasks are similar to each other and they share the same features. This assumption may not be correct in practice because there may exist outlier tasks (i.e., tasks that are not related to all other tasks) or tasks with negative correlation (i.e., tasks that are negatively correlated with some other tasks). In this section, we will discuss how to extend our probabilistic model to deal with these tasks.

We first introduce the matrix variate generalized normal distribution (Gupta & Varga, 1995) which is a generalization of the generalized normal distribution to random matrices.

Definition 6.3 *A matrix $\mathbf{Z} \in \mathbb{R}^{s\times t}$ is a matrix variate generalized normal random variable iff its p.d.f. is given as follows:*

$$p(\mathbf{Z}|\mathbf{M},\mathbf{\Sigma},\mathbf{\Omega},q) = \frac{1}{\left[2\Gamma(1+\frac{1}{q})\right]^{st} det(\mathbf{\Sigma})^t det(\mathbf{\Omega})^s} \exp\left[-\sum_{i=1}^{s}\sum_{j=1}^{t}\left|\sum_{k=1}^{s}\sum_{l=1}^{t}(\Sigma^{-1})_{ik}(Z_{kl}-M_{kl})(\Omega^{-1})_{lj}\right|^q\right],$$

where $\mathbf{\Sigma} \in \mathbb{R}^{s\times s}$ and $\mathbf{\Omega} \in \mathbb{R}^{t\times t}$ are nonsingular, $det(\cdot)$ denotes the determinant of a square matrix, A_{ij} is the (i,j)th element of matrix $\mathbf{A}$ and $(A^{-1})_{ij}$ is the (i,j)th element of the matrix inverse $\mathbf{A}^{-1}$.

We write $\mathbf{Z} \sim \mathcal{MVGN}(\mathbf{M},\mathbf{\Sigma},\mathbf{\Omega},q)$ for a matrix variate generalized normal random variable $\mathbf{Z}$. When $q = 2$, the matrix variate generalized normal distribution becomes the (ordinary) matrix variate normal distribution (Gupta & Nagar, 2000) with row covariance matrix $\mathbf{\Sigma\Sigma}^T$ and column covariance matrix $\mathbf{\Omega\Omega}^T$, which has been used before in multi-task learning (Bonilla et al., 2007; Zhang & Yeung, 2010b, 2010a). From this view, $\mathbf{\Sigma}$ is used to model the relationships between the rows of $\mathbf{Z}$ and $\mathbf{\Omega}$ is to model the relationships between the columns.

We note that the prior on $\mathbf{W}$ in Eq. (6.7) can be written as

$$\mathbf{W} \sim \mathcal{MVGN}(\mathbf{0}, \text{diag}\big((\rho_1,\ldots,\rho_d)^T\big), \mathbf{I}_m, q). \tag{6.10}$$

In this formulation, it can be seen that the columns of $\mathbf{W}$ (and hence the tasks) are independent of each other. However, the tasks are in general not independent. So we propose to use a new prior on $\mathbf{W}$:

$$\mathbf{W} \sim \mathcal{MVGN}(\mathbf{0}, \text{diag}\big((\rho_1,\ldots,\rho_d)^T\big), \mathbf{\Omega}, q), \tag{6.11}$$

where $\mathbf{\Omega}$ models the pairwise relationships between tasks.

The likelihood is still based on the normal distribution. Since in practice the relationships between tasks are not known in advance, we also need to estimate $\mathbf{\Omega}$ from data.

For parameter learning, we again use the EM algorithm to learn the model parameters. Here the model parameters are denoted as $\mathbf{\Theta} = \{\mathbf{W}, \mathbf{b}, \{\sigma_i\}, q, \mathbf{\Omega}\}$. It is easy to show that

$$\ln p(\mathbf{\Theta}|\mathbf{y}, \boldsymbol{\rho}) \propto -\sum_{i=1}^{m}\Big[\sum_{j=1}^{n_i}\frac{(y_j^i - \mathbf{w}_i^T\mathbf{x}_j^i - b_i)^2}{2\sigma_i^2} + \frac{n_i \ln \sigma_i^2}{2}\Big] - \sum_{i=1}^{d}\frac{1}{\rho_i^q}\sum_{j=1}^{m}\Big|\sum_{l=1}^{m} W_{il}(\Omega^{-1})_{lj}\Big|^q$$
$$- md\ln\Gamma(1+\frac{1}{q}) - d\ln\det(\mathbf{\Omega}).$$

Then we compute $\mathbb{E}[\frac{1}{\rho_i^q}|\mathbf{y}, \mathbf{\Theta}^{(k)}]$ as

$$\mathbb{E}\Big[\frac{1}{\rho_i^q}\Big|\mathbf{y}, \mathbf{\Theta}^{(k)}\Big] = \frac{\int_0^\infty \frac{1}{\rho_i^q} p(\rho_i)p(\mathbf{w}_{(k)}^i|\rho_i)d\rho_i}{\int_0^\infty p(\rho_i)p(\mathbf{w}_{(k)}^i|\rho_i)d\rho_i} = \frac{m}{q\sum_{j=1}^m \big|\sum_{l=1}^m W_{il}^{(k)}\big((\Omega^{(k)})^{-1}\big)_{lj}\big|^q} \equiv \alpha_i^{(k)}.$$

In the E-step, the Q-function can be formulated as

$$Q(\mathbf{\Theta}|\mathbf{\Theta}^{(k)}) = -\sum_{i=1}^{m}\Big[\sum_{j=1}^{n_i}\frac{(y_j^i - \mathbf{w}_i^T\mathbf{x}_j^i - b_i)^2}{2\sigma_i^2} + \frac{n_i \ln \sigma_i^2}{2}\Big] - \sum_{i=1}^{d}\alpha_i^{(k)}\sum_{j=1}^{m}\Big|\sum_{l=1}^{m} W_{il}(\Omega^{-1})_{lj}\Big|^q$$
$$- md\ln\Gamma(1+\frac{1}{q}) - d\ln\det(\mathbf{\Omega}).$$

In the M-step, for $\mathbf{W}$ and $\mathbf{\Omega}$, the optimization problem becomes

$$\min_{\mathbf{W},\mathbf{\Omega}} \sum_{i=1}^{m}\gamma_i^{(k)}\sum_{j=1}^{n_i}(\hat{y}_j^i - \mathbf{w}_i^T\mathbf{x}_j^i)^2 + \sum_{i=1}^{d}\alpha_i^{(k)}\sum_{j=1}^{m}\Big|\sum_{l=1}^{m} W_{il}(\Omega^{-1})_{lj}\Big|^q + d\ln\det(\mathbf{\Omega}),$$

where $\gamma_i^{(k)} = \frac{1}{2(\sigma_i^{(k)})^2}$. We define a new variable $\hat{\mathbf{W}} = \mathbf{W}\mathbf{\Omega}^{-1}$ to rewrite the above problem as

$$\min_{\hat{\mathbf{W}},\mathbf{\Omega}} F = \sum_{i=1}^{m}\gamma_i^{(k)}\sum_{j=1}^{n_i}(\hat{y}_j^i - \mathbf{e}_i^T\mathbf{\Omega}^T\hat{\mathbf{W}}^T\mathbf{x}_j^i)^2 + \sum_{i=1}^{d}\alpha_i^{(k)}\sum_{j=1}^{m}|\hat{w}_{ij}|^q + d\ln\det(\mathbf{\Omega}),$$

where $\mathbf{e}_i$ denotes the ith column of the $m \times m$ identity matrix. We use an alternating method to solve this problem. For a fixed $\mathbf{\Omega}$, the problem with respect to $\hat{\mathbf{W}}$ is a convex problem and we use conjugate gradient to solve it with the following subgradient

$$\frac{\partial F}{\partial \hat{\mathbf{W}}} = 2\sum_{i=1}^{m}\gamma_i^{(k)}\sum_{j=1}^{n_i}\Big[\mathbf{x}_j^i(\mathbf{x}_j^i)^T\hat{\mathbf{W}}\mathbf{\Omega}\mathbf{e}_i\mathbf{e}_i^T\mathbf{\Omega}^T - y_j^i\mathbf{x}_j^i\mathbf{e}_i^T\mathbf{\Omega}^T\Big] + q\mathbf{M},$$

where $\mathbf{M}$ is a $d \times m$ matrix with the (i,j)th element $\alpha_i^{(k)}|\hat{w}_{ij}|^{q-1}\text{sign}(\hat{w}_{ij})$. For a fixed $\hat{\mathbf{W}}$, we also use conjugate gradient with the following gradient

$$\frac{\partial F}{\partial \mathbf{\Omega}} = 2\sum_{i=1}^{m}\gamma_i^{(k)}\sum_{j=1}^{n_i}\Big[\hat{\mathbf{W}}^T\mathbf{x}_j^i(\mathbf{x}_j^i)^T\hat{\mathbf{W}}\mathbf{\Omega}\mathbf{e}_i\mathbf{e}_i^T - y_j^i\hat{\mathbf{W}}^T\mathbf{x}_j^i\mathbf{e}_i^T\Big] + d(\mathbf{\Omega}^T)^{-1}.$$

After obtaining the optimal $\hat{\mathbf{W}}^\star$ and $\mathbf{\Omega}^\star$, we can compute the optimal $\mathbf{W}^\star$ as $\mathbf{W}^\star = \hat{\mathbf{W}}^\star\mathbf{\Omega}^\star$. The update rules for $\{\sigma_i\}$, $\{b_i\}$ and q are similar to those in the above section.

In summary, the proposed two models both have the same normal likelihood but different matrix variate priors defined in Eqs. (6.10) and (6.11) respectively, which shows the proposed models are two concrete cases of the proposed framework in section 1.3.

6.5 Related Work

Some probabilistic multi-task feature selection methods have been proposed before (Xiong et al., 2007; Bi et al., 2008). However, they only focus on the $l_{1,2}$ norm. Moreover, they use point estimation in the ARD prior and hence, as discussed in Section 3, are susceptible to overfitting (Qi et al., 2004).

Zhang et al. (2008) proposed a latent variable model for multi-task learning by using the Laplace prior to enforce sparsity. This is equivalent to using the $l_{1,1}$ norm in our framework which, as discussed above, cannot enforce group sparsity among different features over all tasks.

6.6 Experiments

In this section, we study our methods empirically on two cancer classification applications using microarray gene expression data. We compare our methods with three related methods: multi-task feature learning (MTFL) (Argyriou et al., 2008a)[2], multi-task feature selection using $l_{1,2}$ regularization (Liu et al., 2009b)[3], and multi-task feature selection using $l_{1,\infty}$ regularization (Quattoni et al., 2009)[4].

6.6.1 Breast Cancer Classification

We first conduct empirical study on a breast cancer classification application. This application consists of three learning tasks with data collected under different platforms (Shabalin et al., 2008). The dataset for the first task, collected at the Koo Foundation Sun Yat-Sen Cancer Centre in Taipei, contains 89 samples with 8948 genes per sample. The dataset for the second task, obtained from the Netherlands Cancer Institute, contains 97 samples with 16360 genes per sample. Most of the patients in this dataset had stage I or II breast cancer. The dataset for the

[2]http://ttic.uchicago.edu/∼argyriou/code/index.html

[3]http://www.public.asu.edu/∼jye02/Software/SLEP/index.htm

[4]http://www.lsi.upc.edu/∼aquattoni/

third task, obtained using 22K Agilent oligonucleotide arrays, contains 114 samples with 12065 genes per sample. Even though these three datasets were collected under different platforms, they share 6092 common genes which are used in our experiments.

Here we abbreviate the method in Section 4.2 as PMTFS1 and that in Section 4.3 as PMTFS2. For each task, we choose 70% of the data for training and the rest for testing. We perform 10 random splits of the data and report the mean and standard derivation of the classification error over the 10 trials. The results are summarized in Table 6.1. It is clear that PMTFS1 outperforms the three previous methods, showing the effectiveness of our more general formulation with q determined automatically. Moreover, we also note that PMTFS2 is better than PMTFS1. This verifies the usefulness of exploiting the relationships between tasks in multi-task feature selection. Since our methods can estimate q automatically, we compute the mean of the estimated q values over 10 trials. The means for PMTFS1 and PMTFS2 are 2.5003 and 2.6718, respectively, which seem to imply that smaller values of q are preferred for this application. This probably explains why the performance of $\text{MTFS}_{1,\infty}$ is not good when compared with other methods.

Table 6.1: Comparison of different methods on the breast cancer classification application in terms of classification error rate (in mean$\pm$std-dev). Each column in the table represents one task.

Method	1st Task	2nd Task	3rd Task
MTFL	0.3478$\pm$0.1108	0.0364$\pm$0.0345	0.3091$\pm$0.0498
$\text{MTFS}_{1,2}$	0.3370$\pm$0.0228	0.0343$\pm$0.0134	0.2855$\pm$0.0337
$\text{MTFS}_{1,\infty}$	0.3896$\pm$0.0583	0.1136$\pm$0.0579	0.2909$\pm$0.0761
PMTFS1	0.3072$\pm$0.0234	0.0298$\pm$0.0121	0.1786$\pm$0.0245
PMTFS2	0.2870$\pm$0.0228	0.0273$\pm$0.0102	0.1455$\pm$0.0263

6.6.2 Prostate Cancer Classification

We next study a prostate cancer classification application consisting of two tasks. The Singh dataset (Singh et al., 2002) for the first task is made up of laser intensity images from each microarray. The RMA preprocessing method was used to produce gene expression values from these images. On the other hand, the Welsh dataset (Welsh et al., 2001) for the second task is already in the form of gene expression values. Even though the collection techniques for the two datasets are different, they have 12600 genes in common and are used in our experiments.

The experimental setup for this application is similar to that in the previous subsection, that is, 70% of the data of each task are used for training and the rest for testing, and 10 random splits of the data are performed. We report the mean and standard derivation of the classification error over the 10 trials in Table 6.2. As in the first set of experiments, PMTFS1 and PMTFS2 are better than the other three methods compared and PMTFS2 slightly outperforms PMTFS1. The means of the estimated q values for PMTFS1 and PMTFS2 are 2.5865 and 2.6319, respectively. So it seems that smaller values are also preferred for this application.

Table 6.2: Comparison of different methods on the prostate cancer classification application in terms of classification error rate (in mean±std-dev). Each column in the table represents one task.

Method	1st Task	2nd Task
MTFL	0.1226±0.0620	0.3500±0.0085
$\text{MTFS}_{1,2}$	0.1232±0.0270	0.3420±0.0067
$\text{MTFS}_{1,\infty}$	0.2216±0.1667	0.4200±0.1304
PMTFS1	0.1123±0.0170	0.3214±0.0053
PMTFS2	0.1032±0.0136	0.3000±0.0059

6.7 Concluding Remarks

In this chapter, we have proposed a probabilistic framework for general multi-task feature selection using the $l_{1,q}$ norm ($1 < q \leq \infty$). Our model allows the optimal value of q to be determined from data automatically. Besides considering the case in which all tasks are similar, we have also considered the more general and challenging case in which there also exist outlier tasks or tasks with negative correlation.

CHAPTER 7

MULTI-DOMAIN COLLABORATIVE FILTERING

7.1 Introduction

The amount of information available on the Internet is increasing at an astonishing rate, making the search for information a more and more challenging task. As such, recommendation plays an important role in bringing items of potential interest to our attention. Some popular examples include systems for product recommendation in Amazon.com, movie recommendation in Netflix and MovieLens, and reference recommendation in CiteULike. CF is an effective recommendation approach based on the intuitive idea that the preference of a user can be predicted by exploiting the information of other users that share similar interests. In particular, CF techniques exploit past activities of the users, such as their transaction history or product satisfaction expressed in ratings, to predict the future activities of the users. In recent years, CF-based recommendation systems have become increasingly popular because it is generally much easier to collect the past activities of users than their profiles, partially due to privacy considerations.

According to a survey on CF (Su & Khoshgoftaar, 2009), different CF techniques can be classified into three categories: memory-based methods, model-based methods, and hybrid methods. Similar to the idea of nearest neighbor classification, memory-based methods make rating prediction based on the rating behavior of other items and users with similar interests. Some representative methods are (Herlocker et al., 1999; Sarwar et al., 2001; McLaughlin & Herlocker, 2004). One limitation of memory-based methods is that they require the rating data to be dense so that the similarity values can be estimated accurately. Unfortunately, this requirement is not realistic in many applications. To achieve better prediction performance and overcome the shortcomings of memory-based CF methods, model-based CF methods have been proposed and actively studied. Model-based CF techniques use the rating data to learn a model and then use the learned model to make predictions. Many learning models have been used for CF, such as Bayesian belief networks (Breese et al., 1998), graphical models (Jin et al., 2003; Truyen et al., 2009), and dependency networks (Heckerman et al., 2000). Among all model-based CF methods, matrix factorization methods are perhaps the most popular in recent years (Billsus & Pazzani, 1998; Srebro et al., 2005; Rennie & Srebro, 2005; DeCoste, 2006; Salakhutdinov & Mnih, 2007, 2008; Lawrence & Urtasun, 2009; Yu et al., 2009, 2009). These methods assume that user and item features lie in some low-dimensional latent space and then

predictions are made based on the latent features. With the hope of further improving performance, hybrid CF techniques have been proposed to combine memory-based methods with model-based methods, or utilize additional information such as content information. Some examples are (Basu et al., 1998; Pennock et al., 2000; Popescul et al., 2001; Melville et al., 2002; Yu et al., 2003, 2004; Koren, 2008).

Even though CF methods have achieved great successes in recommendation applications, some problems which limit their performance still exist. A big challenge is the data sparsity problem (Su & Khoshgoftaar, 2009) which means that the rating matrix is extremely sparse. Our focus in this chapter is on this data sparsity problem. In particular, we consider a multi-domain CF (MCF) problem which jointly models a collection of rating prediction tasks arising from multiple domains. The MCF problem is particularly suitable for large-scale e-commerce and social networking services which often provide a diverse range of products or services. For example, different product or service categories such as books and electronics naturally constitute different domains. By exploiting the correlation between rating prediction problems in different domains, we can transfer the shared knowledge among similar domains to alleviate the data sparsity problem and therefore improve the rating prediction performance in all domains. Specifically, we propose a probabilistic framework which uses probabilistic matrix factorization (PMF) (Salakhutdinov & Mnih, 2007) to model the rating prediction problem in each domain and allows the knowledge to be adaptively transferred across different domains by automatically learning the correlation between domains. We also introduce the link function for different domains to correct their biases. Experiments conducted on several real-world applications demonstrate the effectiveness of our method.

7.2 Multi-Domain Collaborative Filtering

Let $\mathbf{X}^i \in \mathbb{R}^{m_i \times n_i}$ denote the rating matrix for the ith domain, where $i = 1, \ldots, K$. So for each domain we have m_i users and n_i items. In total we have m users in all domains. Let $\mathbf{U}^i \in \mathbb{R}^{d \times m}$ and $\mathbf{V}^i \in \mathbb{R}^{d \times n_i}$ denote the latent user and item feature matrices with each column $\mathbf{U}^i_j$ and each column $\mathbf{V}^i_k$ representing the user-specific and item-specific latent feature vectors, respectively.

We define the conditional distribution over the observed ratings on the ith domain as

$$p(\mathbf{X}^i | \mathbf{U}^i, \mathbf{V}^i, \sigma_i) = \prod_{j=1}^{m} \prod_{k=1}^{n_i} \left[\mathcal{N}(X^i_{jk} | (\mathbf{U}^i_j)^T \mathbf{V}^i_k, \sigma_i^2) \right]^{I^i_{jk}}, \tag{7.1}$$

where X^i_{jk} denotes the rating of the jth user on the kth element in $\mathbf{X}^i$, and I^i_{jk} is the indicator variable which is equal to 1 if the jth user rated the kth item in the ith domain and is 0 otherwise.

We place zero-mean spherical Gaussian priors (Tipping & Bishop, 1999) on the user features

and item features as

$$p(\mathbf{U}^i|\lambda_i) = \prod_{j=1}^{m} \mathcal{N}(\mathbf{U}_j^i|\mathbf{0}_d, \lambda_i^2 \mathbf{I}_d) \tag{7.2}$$

$$p(\mathbf{V}^i|\eta_i) = \prod_{k=1}^{n_i} \mathcal{N}(\mathbf{V}_k^i|\mathbf{0}_d, \eta_i^2 \mathbf{I}_d). \tag{7.3}$$

To learn the relationships between different domains, we place a matrix variate normal distribution (Gupta & Nagar, 2000) on $\mathbf{U} = [\text{vec}(\mathbf{U}^1), \ldots, \text{vec}(\mathbf{U}^K)]$ where $\text{vec}(\cdot)$ denotes the operator which converts a matrix into a vector in a columnwise manner:

$$p(\mathbf{U}|\mathbf{\Omega}) = \mathcal{MN}_{md\times K}(\mathbf{U}|\mathbf{0}_{md\times K}, \mathbf{I}_{md} \otimes \mathbf{\Omega}). \tag{7.4}$$

More specifically, here the row covariance matrix $\mathbf{I}_{md}$ models the relationships between user latent features and the column covariance matrix $\mathbf{\Omega}$ models the relationships between different $\mathbf{U}^i$'s. In other words, $\mathbf{\Omega}$ models the relationships between domains.

In summary, the priors on $\mathbf{U}$ and $\mathbf{V}$ are

$$\begin{aligned} p(\mathbf{U}) &= \mathcal{MN}_{md\times K}(\mathbf{U}|\mathbf{0}_{md\times K}, \mathbf{I}_{md} \otimes \mathbf{\Omega}) \prod_{i=1}^{K}\prod_{j=1}^{m} \mathcal{N}(\mathbf{U}_j^i|\mathbf{0}_d, \lambda_i^2 \mathbf{I}_d) \\ p(\mathbf{V}) &= \prod_{i=1}^{K}\prod_{k=1}^{n_i} \mathcal{N}(\mathbf{V}_k^i|\mathbf{0}_d, \eta_i^2 \mathbf{I}_d), \end{aligned}$$

where $\mathbf{V} = [\text{vec}(\mathbf{V}^1), \ldots, \text{vec}(\mathbf{V}^K)]$, and the likelihood is defined in Eq. (7.1). So it can be fit into the proposed framework in section 1.3.

7.2.1 Parameter Learning

The log-posterior over $\{\mathbf{U}^i\}$ and $\{\mathbf{V}^i\}$ is given by

$$\begin{aligned} &\ln p(\{\mathbf{U}^i\}, \{\mathbf{V}^i\}|\{\mathbf{X}^i\}, \boldsymbol{\sigma}, \boldsymbol{\lambda}, \boldsymbol{\eta}, \mathbf{\Omega}) \\ =& -\sum_{i=1}^{K} \frac{1}{2\sigma_i^2} \sum_{j=1}^{m}\sum_{k=1}^{n_i} I_{jk}^i \Big(X_{jk}^i - (\mathbf{U}_j^i)^T \mathbf{V}_k^i\Big)^2 - \sum_{i=1}^{K} \frac{1}{2\lambda_i^2} \sum_{j=1}^{m} (\mathbf{U}_j^i)^T \mathbf{U}_j^i - \sum_{i=1}^{K} \frac{1}{2\eta_i^2} \sum_{k=1}^{n_i} (\mathbf{V}_k^i)^T \mathbf{V}_k^i \\ &- \frac{1}{2}\sum_{i=1}^{K} (\ln \sigma_i^2 \sum_{j=1}^{m}\sum_{k=1}^{n_i} I_{jk}^i) - \frac{md}{2}\sum_{i=1}^{K} \ln \lambda_i^2 - \sum_{i=1}^{K} \frac{dn_i}{2} \ln \eta_i^2 - \frac{1}{2}\text{tr}(\mathbf{U}\mathbf{\Omega}^{-1}\mathbf{U}^T) - \frac{md}{2} \ln |\mathbf{\Omega}| \\ &+ \text{Const}, \end{aligned} \tag{7.5}$$

where $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_K)^T$, $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_K)^T$, and $\boldsymbol{\eta} = (\eta_1, \ldots, \eta_K)^T$. We maximize $\ln p(\{\mathbf{U}^i\}, \{\mathbf{V}^i\}|\{\mathbf{X}^i\}, \boldsymbol{\sigma}, \boldsymbol{\lambda}, \boldsymbol{\eta}, \mathbf{\Omega})$ to obtain the MAP solution of $\{\mathbf{U}^i\}$ and $\{\mathbf{V}^i\}$ and the MLE

solution of $\boldsymbol{\sigma}$, $\boldsymbol{\lambda}$, $\boldsymbol{\eta}$ and $\boldsymbol{\Omega}$. We use an alternating method to minimize $J(\{\mathbf{U}^i\}, \{\mathbf{V}^i\}, \boldsymbol{\sigma}, \boldsymbol{\lambda}, \boldsymbol{\eta}, \boldsymbol{\Omega}) = -\ln p(\{\mathbf{U}^i\}, \{\mathbf{V}^i\}|\{\mathbf{X}^i\}, \boldsymbol{\sigma}, \boldsymbol{\lambda}, \boldsymbol{\eta}, \boldsymbol{\Omega})$. In what follows, we will present each subproblem separately.

Optimizing w.r.t. $\mathbf{U}_j^i$ when the other variables are fixed

The derivative of J with respect to $\mathbf{U}_j^i$ can be calculated as

$$\frac{\partial J}{\partial \mathbf{U}_j^i} = \frac{1}{\lambda_i^2}\mathbf{U}_j^i + \frac{1}{\sigma_i^2}\sum_{k=1}^{n_i} I_{jk}^i\Big(\mathbf{V}_k^i(\mathbf{V}_k^i)^T\mathbf{U}_j^i - X_{jk}^i\mathbf{V}_k^i\Big) + \sum_{l=1}^{K}\mathbf{U}_j^l\psi_{li},$$

where $\boldsymbol{\Psi} = \boldsymbol{\Omega}^{-1}$ and ψ_{ij} is the (i,j)th element of $\boldsymbol{\Psi}$. We set the derivative to zero and obtain the analytical solution as

$$\mathbf{U}_j^i = \Big(\sigma_i^2(\frac{1}{\lambda_i^2} + \psi_{ii})\mathbf{I}_d + \sum_{k=1}^{n_i} I_{jk}^i\mathbf{V}_k^i(\mathbf{V}_k^i)^T\Big)^{-1}\Big(\sum_{k=1}^{n_i} I_{jk}^i X_{jk}^i\mathbf{V}_k^i - \sigma_i^2\sum_{l\neq i}\psi_{li}\mathbf{U}_j^l\Big). \tag{7.6}$$

Consider a special case in which different domains are uncorrelated, which means that $\boldsymbol{\Omega}$ and $\boldsymbol{\Psi}$ are diagonal matrices, i.e., $\Psi_{ij} = 0$ for $i \neq j$. Then the update solution for $\mathbf{U}_j^i$ is

$$\mathbf{U}_j^i = \Big(\sigma_i^2(\frac{1}{\lambda_i^2} + \psi_{ii})\mathbf{I}_d + \sum_{k=1}^{n_i} I_{jk}^i\mathbf{V}_k^i(\mathbf{V}_k^i)^T\Big)^{-1}\sum_{k=1}^{n_i} I_{jk}^i X_{jk}^i\mathbf{V}_k^i,$$

which degenerates to the update solution for single-domain matrix factorization.

Optimizing w.r.t. $\mathbf{V}_k^i$ when the other variables are fixed

The derivative of J with respect to $\mathbf{V}_k^i$ can be calculated as

$$\frac{\partial J}{\partial \mathbf{V}_k^i} = \frac{1}{\eta_i^2}\mathbf{V}_k^i + \frac{1}{\sigma_i^2}\sum_{j=1}^{m} I_{jk}^i\Big(\mathbf{U}_j^i(\mathbf{U}_j^i)^T\mathbf{V}_k^i - X_{jk}^i\mathbf{U}_j^i\Big).$$

We set the derivative to zero and obtain the analytical solution as

$$\mathbf{V}_k^i = \Big(\frac{\sigma_i^2}{\eta_i^2}\mathbf{I}_d + \sum_{j=1}^{m} I_{jk}^i\mathbf{U}_j^i(\mathbf{U}_j^i)^T\Big)^{-1}\sum_{j=1}^{m} I_{jk}^i X_{jk}^i\mathbf{U}_j^i. \tag{7.7}$$

Optimizing w.r.t. $\boldsymbol{\Omega}$ when the other variables are fixed

Since $\boldsymbol{\Omega}$ is defined as a covariance matrix, $\boldsymbol{\Omega}$ and $\boldsymbol{\Omega}^{-1}$ are symmetric matrices. Then the derivative of J with respect to $\boldsymbol{\Omega}^{-1}$ can be calculated as

$$\frac{\partial J}{\partial \boldsymbol{\Omega}^{-1}} = \mathbf{U}^T\mathbf{U} - md\,\boldsymbol{\Omega} - \frac{1}{2}(\mathbf{U}^T\mathbf{U} - md\,\boldsymbol{\Omega}) \odot \mathbf{I}_K,$$

where $\odot$ denotes the Hadamard product which is the matrix elementwise product. We set the derivative to zero and get

$$\mathbf{\Sigma} = \frac{1}{2}(\mathbf{\Sigma} \odot \mathbf{I}_K),$$

where $\mathbf{\Sigma} = \mathbf{U}^T\mathbf{U} - md\,\mathbf{\Omega}$. Then we have

$$\Sigma_{ii} = \Sigma_{ii}/2 \;\Rightarrow\; \Sigma_{ii} = 0$$
$$\Sigma_{ij} = 0, i \neq j,$$

where Σ_{ij} is the (i,j)th element of $\mathbf{\Sigma}$. So $\mathbf{\Sigma}$ is a zero matrix and we obtain the analytical solution for $\mathbf{\Omega}$ as

$$\mathbf{\Omega} = \frac{1}{md}\mathbf{U}^T\mathbf{U}. \tag{7.8}$$

Considering Eq. (7.8), the (i,j)th element Ω_{ij} of $\mathbf{\Omega}$, which corresponds to the covariance between the ith and jth domains, can be computed as

$$\Omega_{ij} = \frac{1}{md}\Big(\mathrm{vec}(\mathbf{U}^i)\Big)^T\mathrm{vec}(\mathbf{U}^j) = \frac{1}{md}\mathrm{tr}\Big(\mathbf{U}^i(\mathbf{U}^j)^T\Big),$$

which is the scaled dot product of $\mathrm{vec}(\mathbf{U}^i)$ and $\mathrm{vec}(\mathbf{U}^j)$. Since $\mathrm{vec}(\mathbf{U}^i)$ is modeled as latent user features in the ith domain, using the dot product to represent covariance matches our intuition.

Optimizing w.r.t. σ_i when the other variables are fixed

The derivative of J with respect to σ_i^2 can be calculated as

$$\frac{\partial J}{\partial \sigma_i^2} = -\frac{1}{2\sigma_i^4}\sum_{j=1}^{m}\sum_{k=1}^{n_i} I_{jk}^i\Big(X_{jk}^i - (\mathbf{U}_j^i)^T\mathbf{V}_k^i\Big)^2 + \frac{1}{2\sigma_i^2}\sum_{j=1}^{m}\sum_{k=1}^{n_i} I_{jk}^i.$$

We set the derivative to zero and obtain the analytical solution as

$$\sigma_i^2 = \frac{\sum_{j=1}^{m}\sum_{k=1}^{n_i} I_{jk}^i\Big(X_{jk}^i - (\mathbf{U}_j^i)^T\mathbf{V}_k^i\Big)^2}{\sum_{j=1}^{m}\sum_{k=1}^{n_i} I_{jk}^i}. \tag{7.9}$$

Optimizing w.r.t. λ_i when the other variables are fixed

The derivative of J with respect to λ_i^2 can be calculated as

$$\frac{\partial J}{\partial \lambda_i^2} = -\frac{1}{2\lambda_i^4}\sum_{j=1}^{m}(\mathbf{U}_j^i)^T\mathbf{U}_j^i + \frac{md}{2\lambda_i^2}.$$

We set the derivative to zero and obtain the analytical solution as

$$\lambda_i^2 = \frac{1}{md}\sum_{j=1}^{m}(\mathbf{U}_j^i)^T\mathbf{U}_j^i. \tag{7.10}$$

Optimizing w.r.t. η_i when the other variables are fixed

The derivative of J with respect to η_i^2 can be calculated as

$$\frac{\partial J}{\partial \eta_i^2} = -\frac{1}{2\eta_i^4}\sum_{k=1}^{n_i}(\mathbf{V}_k^i)^T\mathbf{V}_k^i + \frac{dn_i}{2\eta_i^2}.$$

We set the derivative to zero and obtain the analytical solution as

$$\eta_i^2 = \frac{1}{dn_i}\sum_{k=1}^{n_i}(\mathbf{V}_k^i)^T\mathbf{V}_k^i. \tag{7.11}$$

7.2.2 Discussions

To gain more insights into our method, we plug Eqs. (7.8), (7.10) and (7.11) into $J(\{\mathbf{U}^i\}, \{\mathbf{V}^i\}, \boldsymbol{\sigma}, \boldsymbol{\lambda}, \boldsymbol{\eta}, \boldsymbol{\Omega})$. By ignoring some constant terms, $J(\{\mathbf{U}^i\}, \{\mathbf{V}^i\}, \boldsymbol{\sigma}, \boldsymbol{\lambda}, \boldsymbol{\eta}, \boldsymbol{\Omega})$ can be reformulated as

$$\begin{aligned}
&J(\{\mathbf{U}^i\}, \{\mathbf{V}^i\}, \boldsymbol{\sigma}, \boldsymbol{\lambda}, \boldsymbol{\eta}, \boldsymbol{\Omega}) \\
=&\sum_{i=1}^{K}\frac{1}{2\sigma_i^2}\sum_{j=1}^{m}\sum_{k=1}^{n_i} I_{jk}^i\Big(X_{jk}^i - (\mathbf{U}_j^i)^T\mathbf{V}_k^i\Big)^2 + \frac{1}{2}\sum_{i=1}^{K}(\ln\sigma_i^2\sum_{j=1}^{m}\sum_{k=1}^{n_i} I_{jk}^i) + \frac{md}{2}\sum_{i=1}^{K}\ln\Big(\sum_{j=1}^{m}(\mathbf{U}_j^i)^T\mathbf{U}_j^i\Big) \\
&+\sum_{i=1}^{K}\frac{dn_i}{2}\ln\Big(\sum_{k=1}^{n_i}(\mathbf{V}_k^i)^T\mathbf{V}_k^i\Big) + \frac{1}{2(md)^{K-1}}\ln|\mathbf{U}^T\mathbf{U}|.
\end{aligned} \tag{7.12}$$

The first term in Eq. (7.12) measures the empirical loss on the observed ratings, the second term penalizes the complexity of $\boldsymbol{\sigma}$, the third and fifth terms penalize the complexity of $\{\mathbf{U}^i\}$, and the fourth term penalizes the complexity of $\{\mathbf{V}^i\}$.

Since

$$\ln\Big(\sum_{j=1}^{m}(\mathbf{U}_j^i)^T\mathbf{U}_j^i\Big) = \ln\mathrm{tr}\Big(\mathbf{U}^i(\mathbf{U}^i)^T\Big)$$

$$\ln\Big(\sum_{k=1}^{n_i}(\mathbf{V}_k^i)^T\mathbf{V}_k^i\Big) = \ln\mathrm{tr}\Big(\mathbf{V}^i(\mathbf{V}^i)^T\Big),$$

which are related to the trace norms of $\mathbf{U}^i$ and $\mathbf{V}^i$ (Srebro et al., 2005), the third and fourth terms in Eq. (7.12) penalize the ranks of $\mathbf{U}^i$ and $\mathbf{V}^i$, respectively (Fazel et al., 2001). Moreover, according to (Fazel et al., 2003), using the last term in Eq. (7.12) is to minimize the product of all singular values of $\mathbf{U}^i$ which is related to the rank of $\mathbf{U}^i$.

7.3 Incorporation of Link Function

In the above model, the likelihood for the ratings is Gaussian as defined in Eq. (7.1). However, since the ratings are discrete integral values, Gaussian likelihood is not very suitable and hence

using it may affect the performance of our model. Here we consider a modification of our model which first transforms the original ratings by a so-called link function and then applies the above model on the transformed ratings. In what follows, we will present this modification in detail.

The link function for the ith domain is denoted by $g_i(\cdot;\boldsymbol{\theta}_i)$ which is parameterized by $\boldsymbol{\theta}_i$. We require g_i to be monotonically increasing and mapping onto the whole real line; otherwise the probability measure will not be preserved after the transformation. The transformed ratings are denoted by latent variables $Z^i_{jk} = g_i(X^i_{jk})$. Similar to Eq. (7.1), the likelihood is defined on Z^i_{jk} as

$$p(\mathbf{Z}^i|\mathbf{U}^i,\mathbf{V}^i,\sigma_i) = \prod_{j=1}^{m}\prod_{k=1}^{n_i}\left[\mathcal{N}(Z^i_{jk}|(\mathbf{U}^i_j)^T\mathbf{V}^i_k,\sigma_i^2)\right]^{I^i_{jk}}.$$

Then, using the Jacobian transformation, we obtain the likelihood on $\mathbf{X}^i$ as

$$p(\mathbf{X}^i|\mathbf{U}^i,\mathbf{V}^i,\sigma_i) = \prod_{j=1}^{m}\prod_{k=1}^{n_i}\left[\mathcal{N}\Big(g_i(X^i_{jk})|(\mathbf{U}^i_j)^T\mathbf{V}^i_k,\sigma_i^2\Big)g_i'(X^i_{jk})\right]^{I^i_{jk}}, \tag{7.13}$$

where $g_i'(\cdot)$ denotes the derivative function of $g_i(\cdot)$. For simplicity of discussion, we assume that different domains share the same link function, i.e., $g_i(\cdot) = g_j(\cdot),\ \forall i \neq j$. We denote the common link function as $g(\cdot)$ which is parameterized by $\boldsymbol{\theta}$.

For parameter learning, we still maximize the log-posterior to get the MAP solution of $\{\mathbf{U}^i\}$ and $\{\mathbf{V}^i\}$ and the MLE solution of the model parameters including $\boldsymbol{\sigma}$, $\boldsymbol{\lambda}$, $\boldsymbol{\eta}$, $\boldsymbol{\Omega}$ and $\boldsymbol{\theta}$. In this way, both the original model parameters in the above section and the parameters of the link function are learned simultaneously under the same probabilistic framework. We still use an alternating method to optimize the objective function.

In detail, the negative log-posterior of the whole data is computed as

$$\begin{aligned}
&J_1(\{\mathbf{U}^i\},\{\mathbf{V}^i\},\boldsymbol{\sigma},\boldsymbol{\lambda},\boldsymbol{\eta},\boldsymbol{\Omega},\boldsymbol{\theta})\\
=&\sum_{i=1}^{K}\frac{1}{2\sigma_i^2}\sum_{j=1}^{m}\sum_{k=1}^{n_i}I^i_{jk}\Big(g(X^i_{jk})-(\mathbf{U}^i_j)^T\mathbf{V}^i_k\Big)^2+\sum_{i=1}^{K}\frac{1}{2\lambda_i^2}\sum_{j=1}^{m}(\mathbf{U}^i_j)^T\mathbf{U}^i_j+\sum_{i=1}^{K}\frac{1}{2\eta_i^2}\sum_{k=1}^{n_i}(\mathbf{V}^i_k)^T\mathbf{V}^i_k\\
&+\frac{1}{2}\sum_{i=1}^{K}(\ln\sigma_i^2\sum_{j=1}^{m}\sum_{k=1}^{n_i}I^i_{jk})+\frac{md}{2}\sum_{i=1}^{K}\ln\lambda_i^2+\sum_{i=1}^{K}\frac{dn_i}{2}\ln\eta_i^2+\frac{1}{2}\mathrm{tr}(\mathbf{U}\boldsymbol{\Omega}^{-1}\mathbf{U}^T)+\frac{md}{2}\ln|\boldsymbol{\Omega}|\\
&-\sum_{i=1}^{K}\sum_{j=1}^{m}\sum_{k=1}^{n_i}I^i_{jk}\ln g'(X^i_{jk})+\text{Const.}
\end{aligned}\tag{7.14}$$

The update equations for $\{\mathbf{U}^i\}$, $\{\mathbf{V}^i\}$, $\boldsymbol{\sigma}$, $\boldsymbol{\lambda}$ and $\boldsymbol{\eta}$ are similar to Eqs. (7.6)–(7.11) by replacing X^i_{jk} with $g(X^i_{jk})$. For the learning of $\boldsymbol{\theta}$, since there is no analytical update solution, we use

a gradient-based method such as the scaled conjugate gradient method[1] to update $\boldsymbol{\theta}$. More specifically, the gradient of J_1 with respect to θ_l, the lth element of $\boldsymbol{\theta}$, is computed as

$$\frac{\partial J_1}{\partial \theta_l} = \sum_{i=1}^{K} \frac{1}{\sigma_i^2} \sum_{j=1}^{m} \sum_{k=1}^{n_i} I_{jk}^i \Big(g(X_{jk}^i) - (\mathbf{U}_j^i)^T \mathbf{V}_k^i \Big) \frac{\partial g(X_{jk}^i)}{\partial \theta_l} - \sum_{i=1}^{K} \sum_{j=1}^{m} \sum_{k=1}^{n_i} I_{jk}^i \frac{\partial \ln g'(X_{jk}^i)}{\partial \theta_l}.$$

When we want to predict the (q, r)th element in $\mathbf{X}^i$, we first predict the latent variable Z_{qr}^i as

$$\tilde{Z}_{qr}^i = (\mathbf{U}_q^i)^T \mathbf{V}_r^i,$$

and then the prediction for X_{qr}^i is computed as

$$\tilde{X}_{qr}^i = g^{-1}(\tilde{Z}_{qr}^i),$$

where $g^{-1}(\cdot)$ denotes the inverse function of $g(\cdot)$. When $g(\cdot)$ has a simple form, we can easily find the form of $g^{-1}(\cdot)$; when $g^{-1}(\cdot)$ is not easy to obtain, we can use numerical methods such as the bisection method to find the unique root of the equation $g(x) = \tilde{Z}_{qr}^i$ due to the monotonic property of $g(\cdot)$.

In our experiments, we use $g(x) = a \ln(bx + c) + d$ $(a, b, c > 0, d \in \mathbb{R})$ as the link function. Here $g(x)$ is monotonically increasing and its domain is the set of all real numbers.

7.4 Related Work

There is little previous work on the MCF problem. The most related one is (Singh & Gordon, 2008) which proposes a collective matrix factorization (CMF) method for CF. In the case of MCF, the collective matrix factorization method requires a common latent user feature matrix $\mathbf{U}$ which is shared by all domains. However, in real applications in which different domains have heterogenous properties, this requirement is not very reasonable. Our model can be viewed as a generalization of the collective matrix factorization method where each domain has its own latent user feature matrix and the correlation matrix between different domains is learned to improve the performance of all rating problems in all domains. In this sense, collective matrix factorization can be viewed as a special case of our model by restricting all $\mathbf{U}^i$ to be identical. Moreover, a transfer collaborative filtering model was proposed in (Li et al., 2009) which aims at improving the performance of a rating problem with very sparse data with the help of another rating problem which has denser rating data. However, the objective of (Li et al., 2009) is different from ours. For example, the model in (Li et al., 2009) is to improve one rating problem with the help of another rating problem, but in our case, we want to improve the performance of all rating problems in all domains simultaneously. Moreover, the model in (Li et al., 2009)

[1] `http://www.kyb.tuebingen.mpg.de/bs/people/carl/code/minimize/minimize.m`

seems to work only for problems with two domains while our model can work for two or more domains in the same way.

7.5 Experiments

In this section, we report some experiments on two real-world datasets along with our analysis on the results.

7.5.1 Experimental Settings

We test the proposed methods on two public-domain recommendation datasets in which the items come from different domains or sub-domains.

Datasets

We use two commonly used datasets in our experiments including one from movie ratings and one from book ratings. In both datasets, the items can be divided into multiple heterogeneous domains.

- MovieLens[2] is a widely used movie recommendation dataset. It contains 100,000 ratings in the scale 1–5. The ratings are given by 943 users on 1,682 movies. Besides the rating information, genre information about movies is also available.
- Book-Crossing[3] is a public book ratings dataset. A subset of the data is used in our experiment, consisting of ratings on books with category information available on Amazon.com. The subset contains 56,148 ratings in the scale 1–10 and these ratings are given by 28,503 users on 9,009 books.

For the MovieLens dataset, we use the five most popular genres to define the domains, whereas for the Book-Crossing dataset, we use the five general book categories. We randomly select 80% of the rating data from each domain to form the training set and the rest for the test set. Each configuration is iterated 10 times in the experiments.

[2]http://www.grouplens.org/

[3]http://www.informatik.uni-freiburg.de/∼cziegler/BX/

Evaluation Metric

In this chapter, we use root mean squared error (RMSE) as the measure for performance evaluation:

$$\text{RMSE} = \sqrt{\frac{\sum_{i,j}(r_{ij} - \hat{r}_{ij})^2}{N}}, \tag{7.15}$$

where r_{ij} denotes the ground-truth rating of user i for item j, $\hat{r}_{ij}$ denotes the predicted rating, and the denominator N is the number of ratings tested. The smaller the RMSE score, the better the performance.

Baselines

We compare our proposed models with the following two methods:

- Independent collaborative filtering using probabilistic matrix factorization (PMF), which treats different rating prediction problems in different domains independently.
- Collective matrix factorization (CMF) model (Singh & Gordon, 2008), which handles problems involving multiple matrix factorization tasks.

In the following, we refer to our proposed method in Section 2 as MCF and the one in Section 3 as MCF-LF.

7.5.2 Experimental Results

Parameter Setting

An appealing advantage of our probabilistic model is that it has very few parameters to set. In fact, the only parameter that needs to be set is the latent dimensionality d. Figure 7.1 shows the effect of the latent dimensionality on the performance of MCF for a subset of the MovieLens dataset. We can see that the performance in terms of RMSE does not change much after d reaches 10. Therefore, we set d to 10 in the following experiments. Other parameters in PMF, CMF, MCF, MCF-LF are randomly generated.

Results

Table 7.1 shows the experimental results on the MovieLens dataset. We can see that our proposed models have the best performance. The models that take multiple domains into consideration (CMF, MCF, MCF-LF) perform better than PMF which treats different domains independently. MCF, which can learn the similarities between different rating prediction problems, performs better than CMF, demonstraing the effectiveness of exploiting the relationships between

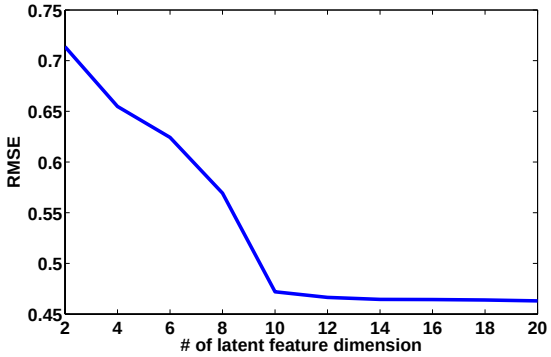


Figure 7.1: Effect of the latent feature dimensionality on the performance of rating prediction for a subset of the MovieLens data.

different domains. Comparing MCF with its variant MCF-LF which has the link function, we can conclude that the link function brings about performance improvement consistently over all domains.

Table 7.1: Comparison of different methods on the MovieLens data. Each column records the RMSE scores on one domain and the last column records the RMSE score on the total testing data. Each row records the mean RMSE of the corresponding method over 10 trials. 1st domain: 'Comedy'; 2nd domain: 'Romance'; 3rd domain: 'Drama'; 4th domain: 'Action'; 5th domain: 'Thriller'.

Method	1st domain	2nd domain	3rd domain	4th domain	5th domain	Total
PMF	0.9642	1.2104	0.9377	1.0035	1.0352	1.0092
CMF	0.8272	0.7977	0.8120	0.7945	0.7987	0.8088
MCF	0.8061	0.7914	0.7907	0.7761	0.7859	0.7913
MCF-LF	**0.8017**	**0.7644**	**0.7806**	**0.7607**	**0.7504**	**0.7755**

Table 7.2 shows the experimental results on the Book-Crossing dataset. MCF and MCF-LF are also the best among all methods compared. Different from the situation in the MovieLens dataset, the performance of CMF is worse than that of PMF, even though CMF considers multiple domains jointly. The reason can be inferred from the correlation matrix in Table 7.4. Since some domains are uncorrelated (1st and 4th domains, and 2nd and 4th domains), the assumption in CMF that different domains share the same latent user features seems not very reasonable, making the performance of CMF worse than that of PMF. However, since our methods can take the correlations between different domains into consideration, they can achieve better performance.

Analysis on Correlation Matrix

Table 7.3 shows the correlation matrix between five domains learned from the MovieLens dataset, which seems consistent with intuition. For example, the genres 'Comedy' and 'Thriller' have the smallest correlation while 'Romance' and 'Drama' have the largest one. For the genre

Table 7.2: Comparison of different methods on the Book-Crossing data. Each column records the RMSE scores on one domain and the last column records the RMSE score on the total testing data. Each row records the mean RMSE of the corresponding method over 10 trials. 1st domain: 'Mystery & Thrillers'; 2nd domain: 'Science Fiction & Fantasy'; 3rd domain: 'Science'; 4th domain: 'Business & Investing'; 5th domain: 'Religion & Spirituality'.

Method	1st domain	2nd domain	3rd domain	4th domain	5th domain	Total
PMF	0.9180	0.9795	0.8308	0.8699	0.8812	0.9269
CMF	0.9620	1.0207	0.9777	0.8465	1.0449	0.9960
MCF	0.7023	0.7046	0.7585	0.7555	0.7371	0.7158
MCF-LF	**0.5686**	**0.5791**	**0.6047**	**0.6001**	**0.5953**	**0.5811**

'Action', we can see that the other genres are ranked into order as: 'Thriller', 'Romance', 'Drama', and 'Comedy', which matches our intuition.

Table 7.4 shows the correlation matrix between five domains learned from the Book-Crossing dataset. Some relations between different domains revealed also seem intuitive. For example, the categories 'Mystery & Thrillers' and 'Business & Investing' have nearly zero correlation and the same is true for 'Science Fiction & Fantasy' and 'Business & Investing'. Also, categories 'Science' and 'Religion & Spirituality' have the largest correlation.

Table 7.3: Mean of correlation matrix learned by MCF-LF on the MovieLens data in different domains. 1st domain: 'Comedy'; 2nd domain: 'Romance'; 3rd domain: 'Drama'; 4th domain: 'Action'; 5th domain: 'Thriller'.

	1st	2nd	3rd	4th	5th
1st	1.0000	0.8837	0.8584	0.8319	0.8302
2nd	0.8837	1.0000	0.9288	0.8855	0.8805
3rd	0.8584	0.9288	1.0000	0.8647	0.8783
4th	0.8319	0.8855	0.8647	1.0000	0.9122
5th	0.8302	0.8805	0.8783	0.9122	1.0000

Table 7.4: Mean of correlation matrix learned by MCF-LF on the Book-Crossing data in different domains. 1st domain: 'Mystery & Thrillers'; 2nd domain: 'Science Fiction & Fantasy'; 3rd domain: 'Science'; 4th domain: 'Business & Investing'; 5th domain: 'Religion & Spirituality'.

	1st	2nd	3rd	4th	5th
1st	1.0000	0.6839	0.4973	0.0137	0.3887
2nd	0.6839	1.0000	0.4636	-0.0489	0.6034
3rd	0.4973	0.4636	1.0000	0.7270	0.7525
4th	0.0137	-0.0489	0.7270	1.0000	0.6682
5th	0.3887	0.6034	0.7525	0.6682	1.0000

7.6 Concluding Remarks

In this chapter, we have addressed the multi-domain collaborative filtering problem in which multiple rating prediction problems are jointly learned. We propose a probabilistic model which considers the correlation between different domains when leveraging all rating data together. Experiments conducted on several recommendation datasets demonstrate the effectiveness of our methods.

CHAPTER 8

CONCLUSIONS AND FUTURE WORK

In this chapter, we conclude the whole thesis and propose several possible directions for future pursuit.

8.1 Conclusions

In this thesis, we have proposed a series of novel multi-task learning models under a probabilistic framework with the ability to learn the task relationship for multi-task learning. Unlike most existing multi-task learning methods which make some assumptions on task relationships, our methods can be viewed as an adaptive way for multi-task learning to automatically learn them. We briefly summarize our contributions in the following:

- Multi-task relationship learning (MTRL): By utilizing a matrix variate normal distribution as a prior on the model parameters of all tasks, we propose a regularized multi-task learning method called *multi-task relationship learning* (MTRL) to learn the task relationships in the form of a covariance matrix. Similar to regularized single-task learning methods such as the ridge regression and SVM models, MTRL has a convex objective function with global optimal solution. For parameter learning, since MTRL needs to learn the model parameters of each task as well as the task relationships between tasks, an alternating optimization method is used for efficiency. Besides the symmetric multi-task learning setting, we also investigate how to use MTRL in the asymmetric setting. Then we extend MTRL to the general case where the prior becomes other matrix variate distributions, that is, matrix variate t distribution. Moreover, we also discuss an application of the idea in MTRL on the transfer metric learning problem.

- Multi-task generalized t process (MTGTP): To utilize the characteristics of Bayesian methods, we propose a *multi-task generalized t process* (MTGTP). Different from MTRL where the task covariance matrix has a parametric form and is a point estimation, MTGTP models the task covariance matrix as a random PSD matrix and places an inverse-Wishart prior on it. By utilizing a weighted-space interpretation of the multi-task GP in (Bonilla et al., 2007), we integrate out the random task covariance matrix, leading to the birth of the generalized t process. By using the generalized t process as a model prior, we adopt

the generalized t noise model as the likelihood of obtaining an analytical marginal likelihood as well as a robust model. Besides the promising empirical results reported for some applications, we also present some theoretical results for MTGTP, that is, asymptotic analysis and learning curve.

- Multi-Task High-Order Task Relationships Learning (MTHOL): We propose a method, which is based on an alternative formulation of MTRL, to learn the high-order task relationships. We propose a matrix variate distribution which is a generalization of the matrix variate normal distribution. We then analyze the properties of this new distribution. By utilizing this new matrix variate distribution as a prior on the model parameters of all tasks, we can learn the model parameters as well as the task relationships under the regularization framework. Inspired by RVM, we also discuss the kernel extension by changing the data representation. Experiments on some benchmark multi-task datasets show the effectiveness of our proposed method.

- Probabilistic Multi-Task Feature Selection (PMTFS): We propose a *probabilistic multi-task feature selection* (PMTFS) method by learning the task relationships to alleviate the assumption in existing multi-task feature selection methods that all tasks are similar. By unifying the existing multi-task feature selection method via the $l_{1,q}$ norm, we first propose a probabilistic interpretation for regularized multi-task feature selection methods. Then based on the probabilistic interpretation, we develop a probabilistic model to learn the task relationships via the matrix variate generalized normal distribution. One by-product of the proposed method is the learning of the q parameter which is not covered and also not easy to accomplish for existing multi-task feature selection methods.

- Multi-Domain Collaborative Filtering (MCF): We propose an application of multi-task learning on a collaborative filtering problem where we are given multiple collaborative filtering problems from some related domains with the hope of alleviating the data sparsity problem in each domain and improving the performance in all domains. We use a PMF model for the CF problem in each domain and learn the correlations between different domains in the form of a covariance matrix by imposing a matrix variate normal distribution as a prior on the model parameters of all PMF models. Moreover, by considering the nature of the discrete integral values in the rating matrix, the Gaussian likelihood is not very suitable. Hence we investigate a modification of our model which first transforms the original ratings by a so-called link function and then applies the above model on the transformed ratings. Empirical studies show that the incorporation of the link function leads to the performance improvement, which demonstrates the effectiveness of the link function.

8.2 Future Work

In our future work, we will pursue the following potential directions:

- A more general implementation for the proposed probabilistic framework: In this thesis, we have used normal distribution and generalized t distribution for the likelihood and matrix variate normal distribution, matrix variate t distribution and matrix variate generalized normal distribution for the prior in the proposed probabilistic framework. In our future work, we are interested in considering the methods with other choices for the likelihood and prior. Moreover, we hope to find a way to solve the optimization problem with the prior and likelihood coming from some distribution family.

- Theoretical analysis on the proposed methods: In this thesis, we have shown the effectiveness of our proposed methods empirically by experimenting them on some benchmark multi-task datasets. In our future work, we will investigate the theoretical background of our methods, that is, generalization bound and sample complexity, and hope to prove the effectiveness of our methods over single-task learning methods and the existing multi-task learning methods in the theory.

- Utilization of additional information: In many applications, there is some other additional information, that is, unlabeled data, which can also be used to alleviate the labeled data deficiency problem. Active learning and semi-supervised learning are two main ways to utilize the unlabeled data. In our previous work (Zhang & Yeung, 2009), we have studied the combination of semi-supervised learning and multi-task learning to utilize the unlabeled data in the scenario of multi-task learning. In our future work, we are interested in studying the combination of semi-supervised learning, active learning and multi-task learning to further improve the performance by using the unlabeled data in a better way.

- Application on compressed sensing: Compressed sensing aims at recovering the sparse signal $\mathbf{w}$ from a measurement vector $\mathbf{b} = \mathbf{A}\mathbf{w}$ for a given matrix $\mathbf{A}$. Compressed sensing can be extended to the joint compressed sensing model in which the signals are represented as a set of jointly sparse vectors sharing a common set of nonzero elements (Cotter et al., 2005; Chen & Huo, 2006; Sun et al., 2009). Specifically, the joint compressed sensing model considers the reconstruction of the signal represented by a matrix $\mathbf{W}$, which is given by a dictionary (or measurement matrix) $\mathbf{A}$ and multiple measurement vector $\mathbf{B}$ such that $\mathbf{B} = \mathbf{A}\mathbf{W}$. Similar to the PMTFS method introduced in Chapter 6, we can use $\|\mathbf{W}\|_{1,q}$ to enforce the joint sparsity in $\mathbf{W}$. Since there usually exists noise in the data, the optimization problem of the joint compressed sensing model can be formulated as: $\min_{\mathbf{W}} \ \lambda\|\mathbf{W}\|_{1,q} + \|\mathbf{A}\mathbf{W} - \mathbf{B}\|_2^2$. This problem is almost identical to the objective function in the PMTFS method except that the loss defines the reconstruction error rather

than the prediction error. So we can use the PMTFS model in Chapter 6 to develop a probabilistic model for the joint compressed sensing model.

Bibliography

Ahmed, A., Yu, K., Xu, W., Gong, Y., & Xing, E. P. (2008). Training hierarchical feed-forward visual recognition models using transfer learning from pseudo-tasks. In *Proceedings of the Tenth European Conference on Computer Vision*, pp. 69–82 Marseille, France.

An, Q., Wang, C., Shterev, I., Wang, E., Carin, L., & Dunson, D. B. (2008). Hierarchical kernel stick-breaking process for multi-task image analysis. In *Proceedings of the Twenty-Fifth International Conference on Machine Learning*, pp. 17–24 Helsinki, Finland.

Ando, R. K., & Zhang, T. (2005). A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, *6*, 1817–1853.

Arellano-Valle, R. B., & Bolfarine, H. (1995). On some characterization of the t-distribution. *Statistics & Probability Letters*, *25*(1), 79–85.

Argyriou, A., Evgeniou, T., & Pontil, M. (2006). Multi-task feature learning. In Schölkopf, B., Platt, J. C., & Hoffman, T. (Eds.), *Advances in Neural Information Processing Systems 19*, pp. 41–48 Vancouver, British Columbia, Canada.

Argyriou, A., Evgeniou, T., & Pontil, M. (2008a). Convex multi-task feature learning. *Machine Learning*, *73*(3), 243–272.

Argyriou, A., Maurer, A., & Pontil, M. (2008b). An algorithm for transfer learning in a heterogeneous environment. In *Proceedings of European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 71–85 Antwerp, Belgium.

Argyriou, A., Micchelli, C. A., Pontil, M., & Ying, Y. (2007). A spectral regularization framework for multi-task structure learning. In Platt, J., Koller, D., Singer, Y., & Roweis, S. (Eds.), *Advances in Neural Information Processing Systems 20*, pp. 25–32 Vancouver, British Columbia, Canada.

Argyriou, A., Micchelli, C. A., Pontil, M., & Ying, Y. (2008). A spectral regularization framework for multi-task structure learning. In Platt, J., Koller, D., Singer, Y., & Roweis, S. (Eds.), *Advances in Neural Information Processing Systems 20*, pp. 25–32 Vancouver, British Columbia, Canada.

Bakker, B., & Heskes, T. (2003). Task clustering and gating for bayesian multitask learning. *Journal of Machine Learning Research*, *4*, 83–99.

Basu, C., Hirsh, H., & Cohen, W. W. (1998). Recommendation as classification: Using social and content-based information in recommendation. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, pp. 714–720 Madison,Wis, USA.

Baxter, J. (1997). A Bayesian/information theoretic model of learning to learn via multiple task sampling. *Machine Learning*, *28*(1), 7–39.

Baxter, J. (2000). A model of inductive bias learning. *Journal of Artifical Intelligence Research*, *12*, 149–198.

Ben-David, S., Gehrke, J., & Schuller, R. (2002). A theoretical framework for learning from a pool of disparate data sources. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 443–449 Edmonton, Alberta, Canada.

Ben-David, S., & Schuller, R. (2003). Exploiting task relatedness for mulitple task learning. In *Proceedings of the Sixteenth Annual Conference on Computational Learning Theory*, pp. 567–580 Washington, DC, USA.

Bi, J., Xiong, T., Yu, S., Dundar, M., & Rao, R. B. (2008). An improved multi-task learning approach with applications in medical diagnosis. In *Proceedings of European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 117–132 Antwerp, Belgium.

Bickel, S., Bogojeska, J., Lengauer, T., & Scheffer, T. (2008). Multi-task learning for HIV therapy screening. In *Proceedings of the Twenty-Fifth International Conference on Machine Learning*, pp. 56–63 Helsinki, Finland.

Billsus, D., & Pazzani, M. J. (1998). Learning collaborative information filters. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pp. 46–54 Madison, Wisconson, USA.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer, New York.

Bonilla, E., Chai, K. M. A., & Williams, C. (2007). Multi-task Gaussian process prediction. In Platt, J., Koller, D., Singer, Y., & Roweis, S. (Eds.), *Advances in Neural Information Processing Systems 20*, pp. 153–160 Vancouver, British Columbia, Canada.

Boyd, S., & Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press, New York, NY.

Breese, J., Heckerman, D., & Kadie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, pp. 43–52 Madison, WI.

Candès, E. J., Wakin, M. B., & Boyd, S. P. (2008). Enhancing sparsity by reweighted l_1 minimization. *Journal of Fourier Analysis and Applications*, *14*(5), 877–905.

Cao, B., Liu, N. N., & Yang, Q. (2010). Transfer learning for collective link prediction in multiple heterogenous domains. In *Proceedings of the Twenty-seventh International Conference on Machine Learning*, pp. 159–166 Haifa, Israel.

Caruana, R. (1997). Multitask learning. *Machine Learning*, *28*(1), 41–75.

Chai, K. M. A., Williams, C. K. I., Klanke, S., & Vijayakumar, S. (2008). Multi-task Gaussian process learning of robot inverse dynamics. In Koller, D., Schuurmans, D., Bengio, Y., & Bottou, L. (Eds.), *Advances in Neural Information Processing Systems 21*, pp. 265–272 Vancouver, British Columbia, Canada.

Chang, H., & Yeung, D.-Y. (2004). Locally linear metric adaptation for semi-supervised clustering. In *Proceedings of the Twenty-first International Conference on Machine Learning* Banff, Alberta, Canada.

Chapelle, O., Zien, A., & Schölkopf, B. (Eds.). (2006). *Semi-Supervised Learning*. MIT Press, Boston.

Chen, J., & Huo, X. (2006). Theoretical results on sparse representations of multiple-measurement vectors. *IEEE Transactions on Signal Processing*, *54*(12), 4634–4643.

Chen, J., Liu, J., & Ye, J. (2010). Learning incoherent sparse and low-rank patterns from multiple tasks. In *Proceedings of the Sixteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1179–1188 Washington, DC, USA.

Chen, J., Tang, L., Liu, J., & Ye, J. (2009). A convex formulation for learning shared structures from multiple tasks. In *Proceedings of the Twenty-sixth International Conference on Machine Learning*, pp. 137–144 Montreal, Quebec, Canada.

Chen, J., Zhao, Z., Ye, J., & Liu, H. (2007). Nonlinear adaptive distance metric learning for clustering. In *Proceedings of the Thirteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 123–132 San Jose, California, USA.

Chung, F. R. K. (1997). *Spectral Graph Theory*. American Mathematical Society, Rhode Island.

Cohn, D., Atlas, L., & Ladner, R. (1994). Improving generalization with active learning. *Machine Learning*, *15*(2), 201–221.

Cohn, D. A., Ghahramani, Z., & Jordan, M. I. (1996). Active learning with statistical models. *Journal of Artificial Intelligence Research*, *4*, 129–145.

Cotter, S. F., Rao, B. D., Engan, K., & Kreutz-Delgado, K. (2005). Sparse solutions to linear inverse problems with multiple measurement vectors. *IEEE Transactions on Signal Processing*, *53*(7), 2477–2488.

Davis, J. V., & Dhillon, I. S. (2008). Structured metric learning for high dimensional problems. In *Proceedings of the Fourteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 195–203 Las Vegas, Nevada, USA.

Davis, J. V., Kulis, B., Jain, P., Sra, S., & Dhillon, I. S. (2007). Information-theoretic metric learning. In *Proceedings of the Twenty-Fourth International Conference on Machine Learning*, pp. 209–216 Corvalis, Oregon, USA.

DeCoste, D. (2006). Collaborative prediction using ensembles of maximum margin matrix factorizations. In *Proceedings of the Twenty-third International Conference on Machine Learning*, pp. 249–256 Pittsburgh, Pennsylvania, USA.

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

Do, C., & Ng, A. (2006). Transfer learning for text classification. In Weiss, Y., Schölkopf, B., & Platt, J. (Eds.), *Advances in Neural Information Processing Systems 18*, pp. 299–306 Vancouver, British Columbia, Canada.

Evgeniou, T., Micchelli, C. A., & Pontil, M. (2005). Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, *6*, 615–637.

Evgeniou, T., & Pontil, M. (2004). Regularized multi-task learning. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 109–117 Seattle, Washington, USA.

Fazel, M., Hindi, H., & Boyd, S. (2001). A rank minimization heuristic with application to minimum order system approximation. In *Proceedings of American Control Conference*, pp. 4734–4739 Arlington, Virginia, USA.

Fazel, M., Hindi, H., & Boyd, S. (2003). Log-det heuristic for matrix rank minimization with applications to hankel and euclidean distance matrices. In *Proceedings of American Control Conference*, pp. 2156–2162 Denver, Colorado, USA.

Figueiredo, M. A. T. (2003). Adaptive sparseness for supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *25*(9), 1150–1159.

Gelman, A., Carlin, J. B., Stern, H. S., & Rubin, D. B. (2003). *Bayesian Data Analysis* (2nd edition). Chapman & Hall.

Gestel, T. V., Suykens, J. A. K., Baesens, B., Viaene, S., Vanthienen, J., Dedene, G., Moor, B. D., & Vandewalle, J. (2004). Benchmarking least squares support vector machine classifiers. *Machine Learning*, *54*(1), 5–32.

Ghosn, J., & Bengio, Y. (1996). Multi-task learning for stock selection. In Mozer, M., Jordan, M. I., & Petsche, T. (Eds.), *Advances in Neural Information Processing Systems 9*, pp. 946–952 Denver, CO, USA.

Goodman, I. R., & Kotz, S. (1973). Multivariate θ-generalized normal distributions. *Journal of Multivariate Analysis*, *3*(2), 204–219.

Grant, M., & Boyd, S. (2008). *CVX: Matlab software for disciplined convex programming (web page and software)*. http://stanford.edu/∼boyd/cvx.

Gretton, A., Borgwardt, K. M., Rasch, M., Schölkopf, B., & Smola, A. J. (2006). A kernel method for the two-sample-problem. In Schölkopf, B., Platt, J., & Hoffman, T. (Eds.), *Advances in Neural Information Processing Systems 19*, pp. 513–520 Vancouver, British Columbia, Canada.

Gupta, A. K., & Nagar, D. K. (2000). *Matrix variate distributions*. Chapman & Hall.

Gupta, A. K., & Varga, T. (1995). Matrix variate θ-generalized normal distribution. *Transactions of The American Mathematical Society*, *347*(4), 1429–1437.

Heckerman, D., Chickering, D., Meek, C., Rounthwaite, R., & Kadie, C. (2000). Dependency networks for collaborative filtering and data visualization. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pp. 264–273 Stanford University, Stanford, California, USA.

Heisele, B., Serre, T., Pontil, M., Vetter, T., & Poggio, T. (2001). Categorization by learning and combining object parts. In Dietterich, T. G., Becker, S., & Ghahramani, Z. (Eds.), *Advances in Neural Information Processing Systems 14*, pp. 1239–1245 Vancouver, British Columbia, Canada.

Herlocker, J. L., Konstan, J. A., Borchers, A., & Riedl, J. (1999). An algorithmic framework for performing collaborative filtering. In *Proceedings of the Twenty-second Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 230–237 Berkeley, CA, USA.

Heskes, T. (1998). Solving a huge number of similar tasks: A combination of multi-task learning and a hierarchical bayesian approach. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pp. 233–241 Madison, Wisconson, USA.

Heskes, T. (2000). Empirical bayes for learning to learn. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pp. 367–374 Stanford University, Standord, CA, USA.

Jaakkola, T., Meila, M., & Jebara, T. (1999). Maximum entropy discrimination. In Solla, S. A., Leen, T. K., & Müller, K.-R. (Eds.), *Advances in Neural Information Processing Systems 12*, pp. 470–476 Denver, Colorado, USA.

Jacob, L., Bach, F., & Vert, J.-P. (2008). Clustered multi-task learning: A convex formulation. In Koller, D., Schuurmans, D., Bengio, Y., & Bottou, L. (Eds.), *Advances in Neural Information Processing Systems 21*, pp. 745–752 Vancouver, British Columbia, Canada.

Jalali, A., Ravikumar, P., Sanghavi, S., & Ruan, C. (2010). A dirty model for multi-task learning. In Lafferty, J., Williams, C. K. I., Shawe-Taylor, J., Zemel, R., & Culotta, A. (Eds.), *Advances in Neural Information Processing Systems 23*, pp. 964–972.

Jebara, T. (2004). Multi-task feature and kernel selection for SVMs. In *Proceedings of the Twenty-first International Conference on Machine Learning* Banff, Alberta, Canada.

Jin, R., Si, L., & Zhai, C. (2003). Preference-based graphic models for collaborative filtering. In *Proceedings of the Nineteenth Conference in Uncertainty in Artificial Intelligence*, pp. 329–336 Acapulco, Mexico.

Jin, R., Wang, S., & Zhou, Y. (2009). Regularized distance metric learning: Theory and algorithm. In Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C. K. I., & Culotta, A. (Eds.), *Advances in Neural Information Processing Systems 22*, pp. 862–870 Vancouver, British Columbia, Canada.

Kato, T., Kashima, H., Sugiyama, M., & Asai, K. (2007). Multi-task learning via conic programming. In Platt, J., Koller, D., Singer, Y., & Roweis, S. (Eds.), *Advances in Neural Information Processing Systems 20*, pp. 737–744 Vancouver, British Columbia, Canada.

Keerthi, S. S., & Shevade, S. K. (2003). SMO algorithm for least-squares SVM formulation. *Neural Computation*, *15*(2), 487–507.

Kienzle, W., & Chellapilla, K. (2006). Personalized handwriting recognition via biased regularization. In *Proceedings of the Twenty-Third International Conference on Machine Learning*, pp. 457–464 Pittsburgh, Pennsylvania, USA.

Koren, Y. (2008). Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the Fourteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* Las Vegas, Nevada, USA.

Lange, K., Hunter, D. R., & Yang, I. (2000). Optimization transfer using surrogate objective functions. *Journal of Computational and Graphical Statistics*, *9*(1), 1–59.

Lapedriza, A., Masip, D., & Vitrì, J. (2008). On the use of independent tasks for face recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* Anchorage, Alaska.

Lawrence, N. D., & Platt, J. C. (2004). Learning to learn with the informative vector machine. In *Proceedings of the Twenty-first International Conference on Machine Learning* Banff, Alberta, Canada.

Lawrence, N. D., Seeger, M., & Herbrich, R. (2002). Fast sparse Gaussian process methods: The informative vector machine. In Becker, S., Thrun, S., & Obermayer, K. (Eds.), *Advances in Neural Information Processing Systems 15*, pp. 609–616 Vancouver, British Columbia, Canada.

Lawrence, N. D., & Urtasun, R. (2009). Non-linear matrix factorization with Gaussian processes. In *Proceedings of the Twenty-sixth International Conference on Machine Learning*, pp. 601–608 Montreal, Quebec, Canada.

Lee, S., Zhu, J., & Xing, E. (2010). Adaptive multi-task lasso: with application to eQTL detection. In Lafferty, J., Williams, C. K. I., Shawe-Taylor, J., Zemel, R. S., & Culotta, A. (Eds.), *Advances in Neural Information Processing Systems 23*, pp. 1306–1314 Vancouver, British Columbia, Canada.

Li, B., Yang, Q., & Xue, X. (2009). Transfer learning for collaborative filtering via a rating-matrix generative model. In *Proceedings of the Twenty-sixth International Conference on Machine Learning*, pp. 617–624 Montreal, Quebec, Canada.

Liao, X., & Carin, L. (2005). Radial basis function network for multi-task learning. In Weiss, Y., Schölkopf, B., & Platt, J. (Eds.), *Advances in Neural Information Processing Systems 18*, pp. 795–802 Vancouver, British Columbia, Canada.

Liu, H., Palatucci, M., & Zhang, J. (2009a). Blockwise coordinate descent procedures for the multi-task LASSO, with applications to neural semantic basis discovery. In *Proceedings of the Twenty-sixth International Conference on Machine Learning*, pp. 649–656 Montreal, Quebec, Canada.

Liu, J., Ji, S., & Ye, J. (2009b). Multi-task feature learning via efficient $L_{2,1}$-norm minimization. In *Proceedings of the Twenty-fifth Conference on Uncertainty in Artificial Intelligence* Montreal, Canada.

Liu, Q., Xu, Q., Zheng, V. W., Xue, H., Cao, Z., & Yang, Q. (2010). Multi-task learning for cross-platform siRNA efficacy prediction: An in-silico study. *BMC Bioinformatics*, *11*(181).

Lobo, M. S., Vandenberghe, L., Boyd, S., & Lebret, H. (1998). Applications of second-order cone programming. *Linear Algebra and its Applications*, *284*, 193–228.

Mathai, A. M. (1997). *Jacobians of Matrix Transformations and Functions of Matrix Argument*. World Scientific.

Maurer, A. (2006). Bounds for linear multi-task learning. *Journal of Machine Learning Research*, *7*, 117–139.

McLaughlin, M. R., & Herlocker, J. L. (2004). A collaborative filtering algorithm and evaluation metric that accurately model the user experience. In *Proceedings of the Twenty-seventh Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 329–336 Sheffield, UK.

Melville, P., Mooney, R. J., & Nagarajan, R. (2002). Content-boosted collaborative filtering for improved recommendations. In *Proceedings of the Eighth National Conference on Artificial intelligence*, pp. 187–192 Edmonton, Alberta, Canada.

Minka, T. P., & Picard, R. W. (1997). Learning how to learn is learning with point sets. MIT Media Lab note, revised in 1999.

Obozinski, G., Taskar, B., & Jordan, M. (2006). Multi-task feature selection. Tech. rep., Department of Statistics, University of California, Berkeley.

Obozinski1, G., Taskar, B., & Jordan, M. I. (2010). Joint covariate selection and joint subspace selection for multiple classification problems. *Statistics and Computing*, *20*(2), 231–252.

Opper, M., & Vivarelli, F. (1998). General bounds on Bayes errors for regression with Gaussian processes. In Kearns, M. J., Solla, S. A., & Cohn, D. A. (Eds.), *Advances in Neural Information Processing Systems 11*, pp. 302–308 Denver, Colorado, USA.

Pan, S., & Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, *22*(10), 1345–1359.

Parameswaran, S., & Weinberger, K. (2010). Large margin multi-task metric learning. In Lafferty, J., Williams, C. K. I., Shawe-Taylor, J., Zemel, R., & Culotta, A. (Eds.), *Advances in Neural Information Processing Systems 23*, pp. 1867–1875 Vancouver, British Columbia, Canada.

Pennock, D., Horvitz, E., Lawrence, S., & Giles, C. (2000). Collaborative filtering by personality diagnosis: A hybrid memory- and model-based approach. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pp. 473–480 Stanford University, Stanford, California, USA.

Popescul, A., Ungar, L., Pennock, D., & Lawrence, S. (2001). Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, pp. 437–444 University of Washington, Seattle, Washington, USA.

Puniyani, K., Kim, S., & Xing, E. P. (2010). Multi-population GWA mapping via multi-task regularized regression. *Bioinformatics*, *26*(12), 208–216.

Qi, Y., Liu, D., Dunson, D. B., & Carin, L. (2008). Multi-task compressive sensing with Dirichlet process priors. In *Proceedings of the Twenty-Fifth International Conference on Machine Learning*, pp. 768–775 Helsinki, Finland.

Qi, Y., Minka, T. P., Picard, R. W., & Ghahramani, Z. (2004). Predictive automatic relevance determination by expectation propagation. In *Proceedings of the Twenty-first International Conference on Machine Learning* Banff, Alberta, Canada.

Quattoni, A., Carreras, X., Collins, M., & Darrell, T. (2009). An efficient projection for $l_{1,\infty}$ regularization. In *Proceedings of the Twenty-sixth International Conference on Machine Learning*, pp. 857–864 Montreal, Quebec, Canada.

Quattoni, A., Collins, M., & Darrell, T. (2008). Transfer learning for image classification with sparse prototype representations. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* Anchorage, Alaska.

Raina, R., Ng, A. Y., & Koller, D. (2006). Constructing informative priors using transfer learning. In *Proceedings of the Twenty-Third International Conference on Machine Learning*, pp. 713–720 Pittsburgh, Pennsylvania, USA.

Rasmussen, C. E., & Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. The MIT Press, Cambridge, MA, USA.

Rennie, J. D. M., & Srebro, N. (2005). Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the Twenty-second International Conference on Machine Learning*, pp. 713–719 Bonn, Germany.

Salakhutdinov, R., & Mnih, A. (2007). Probabilistic matrix factorization. In Platt, J., Koller, D., Singer, Y., & Roweis, S. (Eds.), *Advances in Neural Information Processing Systems 20*, pp. 1257–1264 Vancouver, British Columbia, Canada.

Salakhutdinov, R., & Mnih, A. (2008). Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *Proceedings of the Twenty-fifth International Conference on Machine Learning*, pp. 880–887 Helsinki, Finland.

Sarwar, B. M., Karypis, G., Konstan, J. A., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the Tenth International World Wide Web Conference*, pp. 285–295 Hong Kong.

Schwaighofer, A., Tresp, V., & Yu, K. (2004). Learning Gaussian process kernels via hierarchical bayes. In Saul, L. K., Weiss, Y., & Bottou, L. (Eds.), *Advances in Neural Information Processing Systems 17*, pp. 1209–1216 Vancouver, British Columbia, Canada.

Shabalin, A. A., Tjelmeland, H., Fan, C., Perou, C. M., & Nobel, A. B. (2008). Merging two gene-expression studies via cross-platform normalization. *Bioinformatics*, *24*(9), 1154–1160.

Silver, D. L., Poirier, R., & Currie, D. (2008). Inductive transfer with context-sensitive neural networks. *Machine Learning*, *73*(3), 313–336.

Singh, A. P., & Gordon, G. J. (2008). Relational learning via collective matrix factorization. In *Proceedings of the Fourteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 650–658 Las Vegas, Nevada, USA.

Singh, D., Febbo, P. G., Ross, K., Jackson, D. G., Manola, J., Ladd, C., Tamayo, P., Renshaw, A. A., DAmico, A. V., Richie, J. P., Lander, E. S., Loda, M., Kantoff, P. W., Golub, T. R., & R.Sellers, W. (2002). Gene expression correlates of clinical prostate cancer behavior. *Cancer Cell*, *1*(2), 203–209.

Sollich, P. (1998). Learning curves for Gaussian processes. In Kearns, M. J., Solla, S. A., & Cohn, D. A. (Eds.), *Advances in Neural Information Processing Systems 11*, pp. 344–350 Denver, Colorado, USA.

Sollich, P., & Halees, A. (2002). Learning curves for Gaussian process regression: Approximations and bounds. *Neural Computation*, *14*(6), 1393–1428.

Srebro, N., Rennie, J. D. M., & Jaakkola, T. S. (2005). Maximum-margin matrix factorization. In Saul, L. K., Weiss, Y., & Bottou, L. (Eds.), *Advances in Neural Information Processing Systems 17*, pp. 1329–1336 Vancouver, British Columbia, Canada.

Su, X., & Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*.

Sun, L., Liu, J., Chen, J., & Ye, J. (2009). Efficient recovery of jointly sparse vectors. In Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C. K. I., & Culotta, A. (Eds.), *Advances in Neural Information Processing Systems 22*, pp. 1812–1820 Vancouver, British Columbia, Canada.

Teh, Y. W., Seeger, M., & Jordan, M. I. (2005). Semiparametric latent factor models. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, pp. 333–340 Barbados.

Thrun, S. (1995). Is learning the n-th thing any easier than learning the first?. In Touretzky, D. S., Mozer, M., & Hasselmo, M. E. (Eds.), *Advances in Neural Information Processing Systems 8*, pp. 640–646 Denver, CO.

Thrun, S., & O'Sullivan, J. (1996). Discovering structure in multiple learning tasks: The TC algorithm. In *Proceedings of the Thirteenth International Conference on Machine Learning*, pp. 489–497 Bari, Italy.

Tibshirani, R. (1996). Regression shrinkage and selection via the LASSO. *Journal of the Royal Statistical Society. Series B(Methodological)*, *58*(1), 267–288.

Tipping, M. E. (2001). Sparse bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, *1*, 211–244.

Tipping, M. E., & Bishop, C. M. (1999). Probabilistic principal component analysis. *Journal of the Royal Statistic Society, B*, *61*(3), 611–622.

Torralba, A. B., Murphy, K. P., & Freeman, W. T. (2004). Sharing features: Efficient boosting procedures for multiclass object detection. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 762–769 Washington, DC, USA.

Truyen, T., Phung, D., & Venkatesh, S. (2009). Ordinal boltzmann machines for collaborative filtering. In *Proceedings of the Twenty-fifth Conference on Uncertainty in Artificial Intelligence*, pp. 548–556 Corvallis, Oregon.

Turlach, B. A., Wenables, W. N., & Wright, S. J. (2005). Simultaneous variable selection. *Technometrics*, *47*(3), 349–363.

Uhling, H. (1994). On singular wishart and singular multivariate beta distributions. *Annals of Statistics*, *22*(1), 395–405.

Weinberger, K. Q., Blitzer, J., & Saul, L. K. (2005). Distance metric learning for large margin nearest neighbor classification. In Weiss, Y., Schölkopf, B., & Platt, J. (Eds.), *Advances in*

Neural Information Processing Systems 18, pp. 1473–1480 Vancouver, British Columbia, Canada.

Weinberger, K. Q., & Saul, L. K. (2008). Fast solvers and efficient implementations for distance metric learning. In *Proceedings of the Twenty-Fifth International Conference on Machine Learning*, pp. 1160–1167 Helsinki, Finland.

Welsh, J. B., Sapinoso, L. M., Su, A. I., Kern, S. G., Wang-Rodriguez, J., Moskaluk, C. A., Frierson, F. H., Jr., & Hampton, G. M. (2001). Analysis of gene expression identifies candidate markers and pharmacological targets in prostate cancer. *Cancer Research*, *61*(16), 5974–5978.

Williams, C. K. I. (2000). On a connection between kernel PCA and metric multidimensional scaling. In Leen, T. K., Dietterich, T. G., & Tresp, V. (Eds.), *Advances in Neural Information Processing Systems 13*, pp. 675–681 Vancouver, British Columbia, Canada.

Williams, C. K. I., & Vivarelli, F. (2000). Upper and lower bounds on the learning curve for Gaussian processes. *Machine Learning*, *40*(1), 77–102.

Wipf, D., & Nagarajan, S. (2007). A new view of automatic relevance determination. In Platt, J. C., Koller, D., Singer, Y., & Roweis, S. (Eds.), *Advances in Neural Information Processing Systems 20*, pp. 1625–1632 Vancouver, British Columbia, Canada.

Wipf, D. P., & Nagarajan, S. (2010). Iterative reweighted l_1 and l_2 methods for finding sparse solutions. *Journal of Selected Topics in Signal Processing*, *4*(2), 317–329.

Wu, P., & Dietterich, T. G. (2004). Improving SVM accuracy by training on auxiliary data sources. In *Proceedings of the Twenty-first International Conference on Machine Learning* Banff, Alberta, Canada.

Xiao, B., Yang, X., Xu, Y., & Zha, H. (2009). Learning distance metric for regression by semidefinite programming with application to human age estimation. In *Proceedings of the Seventeenth ACM International Conference on Multimedia*, pp. 451–460 Beijing, China.

Xing, E. P., Ng, A. Y., Jordan, M. I., & Russell, S. J. (2002). Distance metric learning with application to clustering with side-information. In Becker, S., Thrun, S., & Obermayer, K. (Eds.), *Advances in Neural Information Processing Systems 15*, pp. 505–512 Vancouver, British Columbia, Canada.

Xiong, T., Bi, J., Rao, B., & Cherkassky, V. (2007). Probabilistic joint feature selection for multi-task learning. In *Proceedings of the Seventh SIAM International Conference on Data Mining* Minneapolis, Minnesota, USA.

Xu, Q., Pan, S., Xue, H., & Yang, Q. (2010). Multitask learning for protein subcellular location prediction. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *8*(3), 748–759.

Xue, Y., Liao, X., Carin, L., & Krishnapuram, B. (2007). Multi-task learning for classification with Dirichlet process priors. *Journal of Machine Learning Research*, *8*, 35–63.

Yeung, D.-Y., & Chang, H. (2007). A kernel approach for semisupervised metric learning. *IEEE Transactions on Neural Networks*, *18*(1), 141–149.

Yeung, D.-Y., Chang, H., & Dai, G. (2008). A scalable kernel-based semi-supervised metric learning algorithm with out-of-sample generalization ability. *Neural Computation*, *20*(11), 2839–2861.

Yeung, D.-Y., & Zhang, Y. (2009). Learning inverse dynamics by gaussian process regression under the multi-task learning framework. In Sukhatme, G. S. (Ed.), *The Path to Autonomous Robots*, pp. 131–142. Springer.

Yu, K., Lafferty, J. D., Zhu, S., & Gong, Y. (2009). Large-scale collaborative prediction using a nonparametric random effects model. In *Proceedings of the Twenty-sixth International Conference on Machine Learning*, pp. 1185–1192 Montreal, Quebec, Canada.

Yu, K., Schwaighofer, A., Tresp, V., Ma, W.-Y., & Zhang, H. (2003). Collaborative ensemble learning: Combining collaborative and content-based information filtering via hierarchical Bayes. In *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence*, pp. 616–623 Acapulco, Mexico.

Yu, K., Schwaighofer, A., Tresp, V., Xu, X., & Kriegel, H.-P. (2004). Probabilistic memory-based collaborative filtering. *IEEE Transactions on Knowledge and Data Engeering*, *16*(1), 56–69.

Yu, K., & Tresp, V. (2005). Learning to learn and collaborative filtering. In *NIPS Workshop on Inductive Transfer: 10 Years Later* Vancouver, British Columbia, Canada.

Yu, K., Tresp, V., & Schwaighofer, A. (2005). Learning Gaussian processes from multiple tasks. In *Proceedings of the Twenty-Second International Conference on Machine Learning*, pp. 1012–1019 Bonn, Germany.

Yu, K., Tresp, V., & Yu, S. (2004). A nonparametric hierarchical Bayesian framework for information filtering. In *Proceedings of the Twenty-seventh Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 353–360 Sheffield, UK.

Yu, K., Zhu, S., Lafferty, J. D., & Gong, Y. (2009). Fast nonparametric matrix factorization for large-scale collaborative filtering. In *Proceedings of the Thirty-second Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 211–218 Boston, MA, USA.

Yu, S., Tresp, V., & Yu, K. (2007). Robust multi-task learning with t-processes. In *Proceedings of the Twenty-Fourth International Conference on Machine Learning*, pp. 1103–1110 Corvalis, Oregon, USA.

Yuan, M., & Lin, Y. (2006). Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, *68*(1), 49–67.

Zha, Z.-J., Mei, T., Wang, M., Wang, Z., & Hua, X.-S. (2009). Robust distance metric learning with auxiliary knowledge. In *Proceedings of the Twenty-first International Joint Conference on Artificial Intelligence*, pp. 1327–1332 Pasadena, California, USA.

Zhan, D.-C., Li, M., Li, Y.-F., & Zhou, Z.-H. (2009). Learning instance specific distances using metric propagation. In *Proceedings of the Twenty-Sixth International Conference on Machine Learning*, pp. 1225–1232 Montreal, Quebec, Canada.

Zhang, J., Ghahramani, Z., & Yang, Y. (2005). Learning multiple related tasks using latent independent component analysis. In Weiss, Y., Schölkopf, B., & Platt, J. (Eds.), *Advances in Neural Information Processing Systems 18*, pp. 1585–1592 Vancouver, British Columbia, Canada.

Zhang, J., Ghahramani, Z., & Yang, Y. (2008). Flexible latent variable models for multi-task learning. *Machine Learning*, *73*(3), 221–242.

Zhang, K., Gray, J. W., & Parvin, B. (2010a). Sparse multitask regression for identifying common mechanism of response to therapeutic targets. *Bioinformatics*, *26*(12), 97–105.

Zhang, Y., Cao, B., & Yeung, D.-Y. (2010b). Multi-domain collaborative filtering. In *Proceedings of the Twenty-sixth Conference on Uncertainty in Artificial Intelligence*, pp. 725–732 Catalina Island, California.

Zhang, Y., & Yeung, D.-Y. (2010a). A convex formulation for learning task relationships in multi-task learning. In *Proceedings of the Twenty-sixth Conference on Uncertainty in Artificial Intelligence*, pp. 733–742 Catalina Island, California.

Zhang, Y., & Yeung, D.-Y. (2010b). Multi-task learning using generalized t process. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 964–971 Chia Laguna Resort, Sardinia, Italy.

Zhang, Y., & Yeung, D.-Y. (2010c). Multi-task warped Gaussian process for personalized age estimation. In *Proceedings of the Twenty-third IEEE Computer Society Conference on Computer Vision and Pattern Recognition* San Francisco, CA.

Zhang, Y., & Yeung, D.-Y. (2010d). Transfer metric learning by learning task relationships. In *Proceedings of the Sixteenth ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1199–1208 Washington, DC, USA.

Zhang, Y., Yeung, D.-Y., & Xu, Q. (2010). Probabilistic multi-task feature selection. In Lafferty, J., Williams, C. K. I., Shawe-Taylor, J., Zemel, R. S., & Culotta, A. (Eds.), *Advances in Neural Information Processing Systems 23*, pp. 2559–2567 Vancouver, British Columbia, Canada.

Zhang, Y., & Yeung, D.-Y. (2009). Semi-supervised multi-task regression. In *Proceedings of European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Dtabases*, pp. 617–631 Bled, Slovenia.

Zhang, Z., Wu, G., & Chang, E. Y. (2007). Semiparametric regression using student t processes. *IEEE Transactions on Neural Networks*, *18*(6), 1572–1588.

Zhu, S., Yu, K., & Gong, Y. (2007). Predictive matrix-variate t models. In Platt, J., Koller, D., Singer, Y., & Roweis, S. (Eds.), *Advances in Neural Information Processing Systems 20*, pp. 1721–1728 Vancouver, British Columbia, Canada.

APPENDIX A

DETAILS IN CHAPTER 3

A.1 SMO Algorithm for Problem (3.13)

In this section, we present an SMO algorithm to solve problem (3.13).

Recall that the dual form is formulated as follows:

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & h(\boldsymbol{\alpha}) = -\frac{1}{2}\boldsymbol{\alpha}^T\tilde{\mathbf{K}}\boldsymbol{\alpha} + \sum_{i,j}\alpha_j^i y_j^i \\ \text{s.t.} \quad & \sum_j \alpha_j^i = 0 \quad \forall i, \end{aligned} \tag{A.1}$$

where $\mathbf{K}$ is the kernel matrix of all data points from all tasks using the multi-task kernel, and $\tilde{\mathbf{K}} = \mathbf{K} + \frac{1}{2}\mathbf{\Lambda}$ where $\mathbf{\Lambda}$ is a diagonal matrix whose diagonal element is equal to n_i if the corresponding data point belongs to the ith task. So the kernel function for calculating $\tilde{\mathbf{K}}$ is $\tilde{k}_{MT}(\mathbf{x}_{j_1}^{i_1}, \mathbf{x}_{j_2}^{i_2}) = k_{MT}(\mathbf{x}_{j_1}^{i_1}, \mathbf{x}_{j_2}^{i_2}) + \frac{n_{i_1}}{2}\delta(i_1, i_2)\delta(j_1, j_2)$ where $\delta(\cdot,\cdot)$ is the Kronecker delta.

Note that for multiple tasks, there are m constraints in problem (A.1) with one for each task. For the single-task setting, however, there is only one constraint in the dual form.

We define

$$F_j^i = -\frac{\partial h}{\partial \alpha_j^i} = \boldsymbol{\alpha}^T\tilde{\mathbf{k}}_j^i - y_j^i,$$

where $\tilde{\mathbf{k}}_j^i$ is a column of $\tilde{\mathbf{K}}$ corresponding to $\mathbf{x}_j^i$. The Lagrangian of the dual form is

$$\tilde{L} = -\frac{1}{2}\boldsymbol{\alpha}^T\tilde{\mathbf{K}}\boldsymbol{\alpha} + \sum_{i,j}\alpha_j^i y_j^i + \sum_i \beta_i \sum_j \alpha_j^i. \tag{A.2}$$

The KKT conditions for the dual problem are

$$\frac{\tilde{L}}{\partial \alpha_j^i} = \beta_i - F_j^i = 0 \quad \forall i, j.$$

So the optimality conditions will hold at a given $\boldsymbol{\alpha}$ iff for all j we have $F_j^i = \beta_i$, that is, all $\{F_j^i\}_{j=1}^{n_i}$ are identical for $i = 1, \ldots, m$.

We introduce an index triple (i, j, k) to define a violation at $\boldsymbol{\alpha}$ if $F_j^i \neq F_k^i$. Thus the optimality conditions will hold at $\boldsymbol{\alpha}$ iff there does not exist any index triple that defines a violation.

Suppose (i, j, k) defines a violation at some $\boldsymbol{\alpha}$. So we can adjust α_j^i and α_k^i to achieve an increase in h while maintaining the equality constraints $\sum_j \alpha_j^i = 0$ for $i = 1, \ldots, m$. We define the following update:

$$\begin{aligned}
&\tilde{\alpha}_j^i(t) = \alpha_j^i - t; \\
&\tilde{\alpha}_k^i(t) = \alpha_k^i + t; \\
&\text{other elements in } \boldsymbol{\alpha} \text{ remain fixed.}
\end{aligned}$$

The updated $\boldsymbol{\alpha}$ is denoted by $\tilde{\boldsymbol{\alpha}}(t)$. We define $\phi(t) = h(\tilde{\boldsymbol{\alpha}}(t))$ and maximize $\phi(t)$ to find the optimal $t^\star$. Since $\phi(t)$ is a quadratic function of t, $\phi(t) = \phi(0) + t\phi'(0) + \frac{t^2}{2}\phi''(0)$. So the optimal $t^\star$ can be calculated as

$$t^\star = -\frac{\phi'(0)}{\phi''(0)}. \tag{A.3}$$

It is easy to show that

$$\begin{aligned}
\phi'(t) &= \frac{\partial \phi(t)}{\partial t} \\
&= \frac{\partial \phi(t)}{\partial \tilde{\alpha}_j^i(t)} \frac{\partial \tilde{\alpha}_j^i(t)}{\partial t} + \frac{\partial \phi(t)}{\partial \tilde{\alpha}_k^i(t)} \frac{\partial \tilde{\alpha}_k^i(t)}{\partial t} \\
&= \tilde{F}_j^i(t) - \tilde{F}_k^i(t) \\
\phi''(t) &= \frac{\partial \phi'(t)}{\partial t} \\
&= \frac{\partial \phi'(t)}{\partial \tilde{\alpha}_j^i(t)} \frac{\partial \tilde{\alpha}_j^i(t)}{\partial t} + \frac{\partial \phi'(t)}{\partial \tilde{\alpha}_k^i(t)} \frac{\partial \tilde{\alpha}_k^i(t)}{\partial t} \\
&= 2k_{MT}(\mathbf{x}_j^i, \mathbf{x}_k^i) - k_{MT}(\mathbf{x}_j^i, \mathbf{x}_j^i) - k_{MT}(\mathbf{x}_k^i, \mathbf{x}_k^i) - n_i,
\end{aligned}$$

where $\tilde{F}_j^i(t)$ is the value of F_j^i at $\tilde{\boldsymbol{\alpha}}(t)$. So

$$t^\star = -\frac{F_j^i - F_k^i}{\eta}, \tag{A.4}$$

where $\eta = \phi''(0)$ is a constant. After updating $\boldsymbol{\alpha}$, we can update F_q^p for all p, q as well as h as:

$$(F_q^p)^{new} = F_q^p + \tilde{k}_{MT}(\mathbf{x}_j^i, \mathbf{x}_q^p)[\tilde{\alpha}_j^i(t^\star) - \alpha_j^i] + \tilde{k}_{MT}(\mathbf{x}_k^i, \mathbf{x}_q^p)[\tilde{\alpha}_k^i(t^\star) - \alpha_k^i] \tag{A.5}$$

$$h^{new} = h^{old} + \phi(t^\star) - \phi(0) = h^{old} - \frac{\eta(t^\star)^2}{2}. \tag{A.6}$$

The SMO algorithm is an iterative method and we need to define the stopping criterion. Similar to the SMO algorithm for SVM which uses the duality gap to define the stopping criterion, we also use a similar criterion here. When given an $\boldsymbol{\alpha}$, let E denote the current primal

objective function value, h the dual objective function value, $E^\star$ the optimal primal objective function value, and $h^\star$ the optimal dual objective function value. By the Wolfe duality, we have

$$E \geq E^\star = h^\star \geq h.$$

Since $E^\star$ and $h^\star$ are unknown, we define the duality gap as $D_{gap} = E - h$. So the stopping criterion is defined as $D_{gap} \leq \epsilon h$ where ϵ is a small constant. From this stopping criterion, we can get

$$E - E^\star \leq D_{gap} \leq \epsilon h \leq \epsilon E.$$

Next we show how to calculate D_{gap} in term of $\{F_j^i\}$ and $\boldsymbol{\alpha}$. From the constraints in the primal form, we can get

$$\begin{aligned} \varepsilon_j^i &= y_j^i - (\mathbf{w}_i^T \Phi(\mathbf{x}_j^i) + b_i) \\ &= \frac{n_i \alpha_j^i}{2} - b_i - F_j^i. \end{aligned}$$

Finally, D_{gap} can be calculated as

$$\begin{aligned} D_{gap} =& E - h \\ =& \sum_{i=1}^{m} \frac{1}{n_i} \sum_{j=1}^{n_i} (\varepsilon_j^i)^2 + \frac{\lambda_1}{2} \mathrm{tr}(\mathbf{W}\mathbf{W}^T) + \frac{\lambda_2}{2} \mathrm{tr}(\mathbf{W}\boldsymbol{\Omega}^{-1}\mathbf{W}^T) - \left(-\frac{1}{2} \boldsymbol{\alpha}^T \tilde{\mathbf{K}} \boldsymbol{\alpha} + \sum_{i,j} \alpha_j^i y_j^i \right) \\ =& \sum_{i,j} \left[\alpha_j^i (F_j^i - \frac{n_i \alpha_j^i}{4}) + \frac{1}{n_i} (\varepsilon_j^i)^2 \right]. \end{aligned}$$

In the above calculation, we need to determine $\{b_i\}$. Here we choose $\{b_i\}$ to minimize D_{gap} at the given $\boldsymbol{\alpha}$, which is equivalent to minimizing $\sum_{i,j} (\varepsilon_j^i)^2$. So b_i can be calculated as

$$b_i = \frac{1}{n_i} \sum_{j=1}^{n_i} \left(\frac{n_i \alpha_j^i}{2} - F_j^i \right). \tag{A.7}$$

The whole procedure for the SMO algorithm is summarized in Table A.1 below.

A.2 Derivation of Problem (3.21)

In this section, we show how to formulate problem (3.21) as a second-order cone programming (SOCP) problem.

Table A.1: SMO algorithm for Problem (A.1)

Input: training data $\{\mathbf{x}_j^i, y_j^i\}_{j=1}^{n_i}$, ϵ
Initialize $\boldsymbol{\alpha}$ as a zero vector; Initialize $\{F_j^i\}$ and h according to $\boldsymbol{\alpha}$; Repeat Find a triple (i, j, k) that defines a violation for each task; Calculate the optimal adjusted value $t^\star$ using Eq. (A.4); Update $\{F_j^i\}$ and h according to Eqs. (A.5) and (A.6); Calculate $\{b_i\}$ according to Eq. (A.7) Until $D_{gap} \leq \epsilon f$
Output: $\boldsymbol{\alpha}$ and $\mathbf{b}$.

We write $\mathbf{W}_{\tilde{m}}^T\mathbf{W}_{\tilde{m}} = \begin{pmatrix} \boldsymbol{\Psi}_{11} & \boldsymbol{\Psi}_{12} \\ \boldsymbol{\Psi}_{12}^T & \Psi_{22} \end{pmatrix}$ where $\boldsymbol{\Psi}_{11} \in \mathbb{R}^{m\times m}$, $\boldsymbol{\Psi}_{12} \in \mathbb{R}^{m\times 1}$ and $\Psi_{22} \in \mathbb{R}$. Then $\tilde{\boldsymbol{\Omega}} - t\mathbf{W}_{\tilde{m}}^T\mathbf{W}_{\tilde{m}} \succeq \mathbf{0}$ is equivalent to

$$
\begin{aligned}
&(1-\sigma)\boldsymbol{\Omega} - t\boldsymbol{\Psi}_{11} \succeq \mathbf{0} \\
&\sigma - t\Psi_{22} \geq (\boldsymbol{\omega}_{\tilde{m}} - t\boldsymbol{\Psi}_{12})^T\Big((1-\sigma)\boldsymbol{\Omega} - t\boldsymbol{\Psi}_{11}\Big)^{-1}(\boldsymbol{\omega}_{\tilde{m}} - t\boldsymbol{\Psi}_{12}),
\end{aligned}
$$

which can be reformulated as

$$
\begin{aligned}
&(1-\sigma)\mathbf{I}_m - t\boldsymbol{\Omega}^{-\frac{1}{2}}\boldsymbol{\Psi}_{11}\boldsymbol{\Omega}^{-\frac{1}{2}} \succeq \mathbf{0} \\
&\sigma - t\Psi_{22} \geq (\boldsymbol{\omega}_{\tilde{m}} - t\boldsymbol{\Psi}_{12})^T\boldsymbol{\Omega}^{-\frac{1}{2}}\Big((1-\sigma)\mathbf{I}_m - t\boldsymbol{\Omega}^{-\frac{1}{2}}\boldsymbol{\Psi}_{11}\boldsymbol{\Omega}^{-\frac{1}{2}}\Big)^{-1}\boldsymbol{\Omega}^{-\frac{1}{2}}(\boldsymbol{\omega}_{\tilde{m}} - t\boldsymbol{\Psi}_{12}),
\end{aligned}
$$

where $\boldsymbol{\Omega}^{-\frac{1}{2}}$ can be computed in advance. Let $\tilde{\boldsymbol{\Psi}}_{11} = \boldsymbol{\Omega}^{-\frac{1}{2}}\boldsymbol{\Psi}_{11}\boldsymbol{\Omega}^{-\frac{1}{2}}$, $\mathbf{U}$ and $\lambda_1, \ldots, \lambda_m$ denote the eigenvector matrix and eigenvalues of $\tilde{\boldsymbol{\Psi}}_{11}$ with $\lambda_1 \geq \ldots \geq \lambda_m \geq 0$. Then

$$
(1-\sigma)\mathbf{I}_m - t\tilde{\boldsymbol{\Psi}}_{11} \succeq \mathbf{0} \Longleftrightarrow 1 - \sigma \geq \lambda_1 t
$$

and

$$
\Big((1-\sigma)\mathbf{I}_m - t\tilde{\boldsymbol{\Psi}}_{11}\Big)^{-1} = \mathbf{U}\,\mathrm{diag}\Big(\frac{1}{1-\sigma-t\lambda_1}, \ldots, \frac{1}{1-\sigma-t\lambda_m}\Big)\,\mathbf{U}^T.
$$

Combining the above results, problem (3.20) is formulated as

$$
\begin{aligned}
\min_{\boldsymbol{\omega}_{\tilde{m}},\sigma,\mathbf{f},t} \quad & -t \\
\text{s.t.} \quad & 1 - \sigma \geq t\lambda_1 \\
& \mathbf{f} = \mathbf{U}^T\boldsymbol{\Omega}^{-\frac{1}{2}}(\boldsymbol{\omega}_{\tilde{m}} - t\boldsymbol{\Psi}_{12}) \\
& \sum_{j=1}^{m} \frac{f_j^2}{1-\sigma-t\lambda_j} \leq \sigma - t\Psi_{22} \\
& \boldsymbol{\omega}_{\tilde{m}}^T\boldsymbol{\Omega}^{-1}\boldsymbol{\omega}_{\tilde{m}} \leq \sigma - \sigma^2,
\end{aligned} \tag{A.8}
$$

where f_j is the jth element of $\mathbf{f}$. By introducing new variables h_j and r_j $(j = 1, \ldots, m)$, (A.8) is reformulated as

$$\begin{aligned} \min_{\boldsymbol{\omega}_{\tilde{m}}, \sigma, \mathbf{f}, t, \mathbf{h}, \mathbf{r}} \quad & -t \\ \text{s.t.} \quad & 1 - \sigma \geq t\lambda_1 \\ & \mathbf{f} = \mathbf{U}^T \boldsymbol{\Omega}^{-\frac{1}{2}} (\boldsymbol{\omega}_{\tilde{m}} - t\boldsymbol{\Psi}_{12}) \\ & \sum_{j=1}^{m} h_j \leq \sigma - t\Psi_{22} \\ & r_j = 1 - \sigma - t\lambda_j \; \forall j \\ & \frac{f_j^2}{r_j} \leq h_j \; \forall j \\ & \boldsymbol{\omega}_{\tilde{m}}^T \boldsymbol{\Omega}^{-1} \boldsymbol{\omega}_{\tilde{m}} \leq \sigma - \sigma^2. \end{aligned} \tag{A.9}$$

Since

$$\frac{f_j^2}{r_j} \leq h_j \; (r_j, h_j > 0) \iff \left\| \begin{pmatrix} f_j \\ \frac{r_j - h_j}{2} \end{pmatrix} \right\|_2 \leq \frac{r_j + h_j}{2}$$

and

$$\boldsymbol{\omega}_{\tilde{m}}^T \boldsymbol{\Omega}^{-1} \boldsymbol{\omega}_{\tilde{m}} \leq \sigma - \sigma^2 \iff \left\| \begin{pmatrix} \boldsymbol{\Omega}^{-\frac{1}{2}} \boldsymbol{\omega}_{\tilde{m}} \\ \frac{\sigma - 1}{2} \\ \sigma \end{pmatrix} \right\|_2 \leq \frac{\sigma + 1}{2},$$

problem (A.9) is an SOCP problem (Lobo et al., 1998) with $O(m)$ variables and $O(m)$ constraints. Then we can use a standard solver to solve problem (A.9) efficiently.

A.3 Detailed Procedure to Compare Two Bounds

In this section, we will prove that the upper bound in Eq. (3.25) is tighter than that in Eq. (3.24), which means that the following inequality holds:

$$\operatorname{tr}\big(\mathbf{M}^{-1}(\mathbf{I}_d + \mathbf{W}\boldsymbol{\Omega}^{-1}\mathbf{W}^T)\big) + \ln|\mathbf{M}| - d \leq \operatorname{tr}(\mathbf{W}\boldsymbol{\Omega}^{-1}\mathbf{W}^T), \tag{A.10}$$

where $\mathbf{M} = \mathbf{I}_d + \mathbf{W}^{(t)}(\boldsymbol{\Omega}^{(t)})^{-1}(\mathbf{W}^{(t)})^T$ or, more generally, any positive definite matrix.

To prove (A.10), we first prove the following Lemma.

Lemma A.1 *For two $d \times d$ positive definite matrices $\mathbf{A}$ and $\mathbf{B}$, the following equality holds:*

$$\operatorname{tr}(\mathbf{A}^{-1}\mathbf{B}) + \ln|\mathbf{A}| \leq \operatorname{tr}(\mathbf{B}).$$

Proof:
Consider the function $F(\mathbf{X}) = \mathrm{tr}(\mathbf{X}^{-1}\mathbf{B}) + \ln|\mathbf{X}|$. We set its derivative to zero to get

$$\frac{\partial F(\mathbf{X})}{\partial \mathbf{X}} = \mathbf{X}^{-1} - \mathbf{X}^{-1}\mathbf{B}\mathbf{X}^{-1} = 0 \;\Rightarrow\; \mathbf{X} = \mathbf{B}.$$

It is easy to prove that the maximum of $F(\mathbf{X})$ holds at $\mathbf{X} = \mathbf{B}$, which implies

$$\mathrm{tr}(\mathbf{A}^{-1}\mathbf{B}) + \ln|\mathbf{A}| = F(\mathbf{A}) \leq F(\mathbf{B}) = \ln|\mathbf{B}| + d.$$

By using Lemma 1, we can get

$$\ln|\mathbf{B}| + d \leq \mathrm{tr}(\mathbf{B}).$$

Finally, we can get

$$\mathrm{tr}(\mathbf{A}^{-1}\mathbf{B}) + \ln|\mathbf{A}| \leq \ln|\mathbf{B}| + d \leq \mathrm{tr}(\mathbf{B}),$$

which is the conclusion. □

By using Lemma 2 where we let $\mathbf{A} = \mathbf{M}$ and $\mathbf{B} = \mathbf{I}_d + \mathbf{W}\boldsymbol{\Omega}^{-1}\mathbf{W}^T$, we can prove (A.10).

A.4 Optimization Procedure for Problem (3.28)

We present here the optimization procedure for solving problem (3.28). We use an alternating method with two subproblems to be presented separately below.

Optimizing w.r.t. $\boldsymbol{\Sigma}_i$ when $\boldsymbol{\Omega}$ and $\{\boldsymbol{\Sigma}\}_{-i}$ are fixed

We first define $\tilde{\boldsymbol{\Sigma}}$ and $\boldsymbol{\Omega}^{-1}$ as

$$\begin{aligned}\tilde{\boldsymbol{\Sigma}} &= \Big(\mathrm{vec}(\boldsymbol{\Sigma}_i), \tilde{\boldsymbol{\Sigma}}_{-i}\Big)\\ \boldsymbol{\Omega}^{-1} &= \begin{pmatrix} \gamma_{ii} & \boldsymbol{\gamma}_i^T \\ \boldsymbol{\gamma}_i & \boldsymbol{\Gamma}_{-i} \end{pmatrix}.\end{aligned}$$

Then the third term in the objective function of problem (3.28) can be rewritten as

$$\begin{aligned}\frac{\lambda_2}{2}\mathrm{tr}(\tilde{\boldsymbol{\Sigma}}\boldsymbol{\Omega}^{-1}\tilde{\boldsymbol{\Sigma}}^T) =& \frac{\lambda_2}{2}\mathrm{tr}\Big((\mathrm{vec}(\boldsymbol{\Sigma}_i), \tilde{\boldsymbol{\Sigma}}_{-i}) \begin{pmatrix} \gamma_{ii} & \boldsymbol{\gamma}_i^T \\ \boldsymbol{\gamma}_i & \boldsymbol{\Gamma}_{-i} \end{pmatrix} \begin{pmatrix} \mathrm{vec}(\boldsymbol{\Sigma}_i)^T \\ \tilde{\boldsymbol{\Sigma}}_{-i}^T \end{pmatrix}\Big)\\ =& \frac{\lambda_2}{2}\Big(\gamma_{ii}\|\mathrm{vec}(\boldsymbol{\Sigma}_i)\|_2^2 + 2\boldsymbol{\gamma}_i^T\tilde{\boldsymbol{\Sigma}}_{-i}^T\mathrm{vec}(\boldsymbol{\Sigma}_i) + \mathrm{tr}(\tilde{\boldsymbol{\Sigma}}_{-i}\boldsymbol{\Gamma}_{-i}\tilde{\boldsymbol{\Sigma}}_{-i}^T)\Big)\\ =& \frac{\lambda_2}{2}\Big(\gamma_{ii}\|\boldsymbol{\Sigma}_i\|_F^2 + 2\mathrm{tr}(\mathbf{M}\boldsymbol{\Sigma}_i) + \mathrm{tr}(\tilde{\boldsymbol{\Sigma}}_{-i}\boldsymbol{\Gamma}_{-i}\tilde{\boldsymbol{\Sigma}}_{-i}^T)\Big),\end{aligned}$$

where $\|\cdot\|_2$ denotes the 2-norm of a vector and $\mathbf{M}$ is a matrix such that $\mathrm{vec}(\mathbf{M}) = \tilde{\boldsymbol{\Sigma}}_{-i}\boldsymbol{\gamma}_i$. Note that the third term in the last equation above is independent of $\boldsymbol{\Sigma}_i$. It is easy to show that $\mathbf{M}$ is

a symmetric matrix. The optimization problem with respect to $\mathbf{\Sigma}_i$ becomes

$$\begin{aligned} \min_{\mathbf{\Sigma}_i} \quad & \frac{2}{n_i(n_i-1)} \sum_{j<k} g\Big(y^i_{j,k}\left[1 - \|\mathbf{x}^i_j - \mathbf{x}^i_k\|^2_{\mathbf{\Sigma}_i}\right]\Big) + \frac{\lambda_1 + \lambda_2\gamma_{ii}}{2}\|\mathbf{\Sigma}_i\|^2_F + \lambda_2 \text{tr}(\mathbf{M}\mathbf{\Sigma}_i) \\ \text{s.t.} \quad & \mathbf{\Sigma}_i \succeq \mathbf{0}. \end{aligned} \tag{A.11}$$

It is easy to see that this problem is a convex semidefinite programming (SDP) problem since the objective function is convex with respect to $\mathbf{\Sigma}_i$ and the constraint is a PSD constraint on $\mathbf{\Sigma}_i$. Even though solving an SDP problem is computationally demanding with poor scalability, we can adopt the technique in (Weinberger & Saul, 2008) and use gradient projection to solve it.

Optimizing w.r.t. $\mathbf{\Omega}$ when $\{\mathbf{\Sigma}_i\}$ are fixed

When $\{\mathbf{\Sigma}_i\}$ are fixed, the optimization problem for finding $\mathbf{\Omega}$ becomes

$$\begin{aligned} \min_{\mathbf{\Omega}} \quad & \text{tr}(\mathbf{\Omega}^{-1}\tilde{\mathbf{\Sigma}}^T\tilde{\mathbf{\Sigma}}) \\ \text{s.t.} \quad & \mathbf{\Omega} \succeq 0 \\ & \text{tr}(\mathbf{\Omega}) = 1. \end{aligned} \tag{A.12}$$

Then we have

$$\begin{aligned} \text{tr}(\mathbf{\Omega}^{-1}\mathbf{A}) &= \text{tr}(\mathbf{\Omega}^{-1}\mathbf{A})\text{tr}(\mathbf{\Omega}) \\ &= \text{tr}((\mathbf{\Omega}^{-\frac{1}{2}}\mathbf{A}^{\frac{1}{2}})(\mathbf{A}^{\frac{1}{2}}\mathbf{\Omega}^{-\frac{1}{2}}))\text{tr}(\mathbf{\Omega}^{\frac{1}{2}}\mathbf{\Omega}^{\frac{1}{2}}) \\ &\geq (\text{tr}(\mathbf{\Omega}^{-\frac{1}{2}}\mathbf{A}^{\frac{1}{2}}\mathbf{\Omega}^{\frac{1}{2}}))^2 = (\text{tr}(\mathbf{A}^{\frac{1}{2}}))^2, \end{aligned}$$

where $\mathbf{A} = \tilde{\mathbf{\Sigma}}^T\tilde{\mathbf{\Sigma}}$. The first equality holds because of the last constraint in problem (A.12) and the last inequality holds because of the Cauchy-Schwarz inequality for the Frobenius norm. Moreover, $\text{tr}(\mathbf{\Omega}^{-1}\mathbf{A})$ attains its minimum value $(\text{tr}(\mathbf{A}^{\frac{1}{2}}))^2$ if and only if

$$\mathbf{\Omega}^{-\frac{1}{2}}\mathbf{A}^{\frac{1}{2}} = a\mathbf{\Omega}^{\frac{1}{2}}$$

for some constant a and $\text{tr}(\mathbf{\Omega}) = 1$. So we can get the following analytical solution:

$$\mathbf{\Omega} = \frac{\left(\tilde{\mathbf{\Sigma}}^T\tilde{\mathbf{\Sigma}}\right)^{\frac{1}{2}}}{\text{tr}\left(\left(\tilde{\mathbf{\Sigma}}^T\tilde{\mathbf{\Sigma}}\right)^{\frac{1}{2}}\right)}.$$

APPENDIX B

DETAILS IN CHAPTER 4

In this section, we provide the proofs for Eqs. (4.2), (4.3) and (4.6).

Before we present our proofs, we first review some relevant properties of the matrix variate normal distribution and the Wishart distribution as given in (Gupta & Nagar, 2000).

Lemma B.1 ((Gupta & Nagar, 2000), Corollary 2.3.10.1) *If* $\mathbf{X} \sim \mathcal{MN}_{q\times s}(\mathbf{M}, \mathbf{\Sigma}\otimes\mathbf{\Psi})$, $\mathbf{d} \in \mathbb{R}^q$ *and* $\mathbf{c} \in \mathbb{R}^s$, *then*

$$\mathbf{d}^T\mathbf{X}\mathbf{c} \sim \mathcal{N}(\mathbf{d}^T\mathbf{M}\mathbf{c}, (\mathbf{d}^T\mathbf{\Sigma}\mathbf{d})(\mathbf{c}^T\mathbf{\Psi}\mathbf{c})).$$

Lemma B.2 ((Gupta & Nagar, 2000), Theorem 2.3.5) *If* $\mathbf{X} \sim \mathcal{MN}_{q\times s}(\mathbf{M}, \mathbf{\Sigma} \otimes \mathbf{\Psi})$ *and* $\mathbf{A} \in \mathbb{R}^{s\times s}$, *then*

$$\mathbb{E}(\mathbf{X}\mathbf{A}\mathbf{X}^T) = tr(\mathbf{A}^T\mathbf{\Psi})\mathbf{\Sigma} + \mathbf{M}\mathbf{A}\mathbf{M}^T.$$

Lemma B.3 ((Gupta & Nagar, 2000), Theorem 3.3.16) *If* $\mathbf{S} \sim \mathcal{MN}_q(a, \mathbf{\Sigma})$ *where* $a - q - 1 > 0$, *then*

$$\mathbb{E}(\mathbf{S}^{-1}) = \frac{\mathbf{\Sigma}^{-1}}{a-q-1},$$

where $\mathbf{S}^{-1}$ *denotes the inverse of* $\mathbf{S}$.

For Eq. (4.2), using Lemma B.1 and the fact that $\mathbf{W} \sim \mathcal{MN}_{d'\times m}(\mathbf{0}_{d'\times m}, \mathbf{I}_{d'} \otimes \mathbf{\Sigma})$, we can get

$$f_j^i \overset{\text{def}}{=} \phi(\mathbf{x}_j^i)^T\mathbf{w}_i = \phi(\mathbf{x}_j^i)^T\mathbf{W}\mathbf{e}_{m,i} \sim \mathcal{N}(0, (\phi(\mathbf{x}_j^i)^T\mathbf{I}_{d'}\phi(\mathbf{x}_j^i))(\mathbf{e}_{m,i}^T\mathbf{\Sigma}\mathbf{e}_{m,i})).$$

Since $\phi(\mathbf{x}_j^i)^T\mathbf{I}_{d'}\phi(\mathbf{x}_j^i) = k(\mathbf{x}_j^i, \mathbf{x}_j^i)$ and $\mathbf{e}_{m,i}^T\mathbf{\Sigma}\mathbf{e}_{m,i} = \Sigma_{ii}$, we can get $f_j^i \sim \mathcal{N}(0, \Sigma_{ii}k(\mathbf{x}_j^i, \mathbf{x}_j^i))$.

For Eq. (4.3), we have

$$\begin{aligned}\langle f_j^i, f_s^r\rangle &= \int \phi(\mathbf{x}_j^i)^T\mathbf{W}\mathbf{e}_{m,i}\mathbf{e}_{m,r}^T\mathbf{W}^T\phi(\mathbf{x}_s^r)p(\mathbf{W})d\mathbf{W} \\ &= \phi(\mathbf{x}_j^i)^T\mathbb{E}(\mathbf{W}\mathbf{e}_{m,i}\mathbf{e}_{m,r}^T\mathbf{W}^T)\phi(\mathbf{x}_s^r),\end{aligned}$$

then using Lemma B.2 and the fact that $\mathbf{W} \sim \mathcal{MN}_{d'\times m}(\mathbf{0}_{d'\times m}, \mathbf{I}_{d'} \otimes \mathbf{\Sigma})$, we can get

$$\begin{aligned}\langle f_j^i, f_s^r\rangle &= \phi(\mathbf{x}_j^i)^T\text{tr}(\mathbf{e}_{m,r}\mathbf{e}_{m,i}^T\mathbf{\Sigma})\mathbf{I}_{d'}\phi(\mathbf{x}_s^r) \\ &= \text{tr}(\mathbf{e}_{m,r}\mathbf{e}_{m,i}^T\mathbf{\Sigma})k(\mathbf{x}_j^i, \mathbf{x}_s^r) \\ &= \mathbf{e}_{m,i}^T\mathbf{\Sigma}\mathbf{e}_{m,r}k(\mathbf{x}_j^i, \mathbf{x}_s^r) \\ &= \Sigma_{ir}k(\mathbf{x}_j^i, \mathbf{x}_s^r).\end{aligned}$$

The second last equation holds because $\mathbf{e}_{m,i}$ and $\mathbf{e}_{m,r}$ are two vectors.

For Eq. (4.6), recall that the two random variables $\mathbf{S} \sim \mathcal{W}_{d'}(\nu + d' - 1, \mathbf{I}_{d'})$ and $\mathbf{Z} \sim \mathcal{MN}_{d' \times m}(\mathbf{0}_{d' \times m}, \mathbf{I}_{d'} \otimes \mathbf{\Psi})$ are independent and $\mathbf{W} = \mathbf{S}^{-1/2}\mathbf{Z}$. Then we can get

$$
\begin{aligned}
\langle f_j^i, f_s^r \rangle &= \int \int \phi(\mathbf{x}_j^i)^T \mathbf{w}_i \mathbf{w}_r^T \phi(\mathbf{x}_s^r) p(\mathbf{w}_i) p(\mathbf{w}_r) d\mathbf{w}_i d\mathbf{w}_r \\
&= \int \phi(\mathbf{x}_j^i)^T \mathbf{W} \mathbf{e}_{m,i} \mathbf{e}_{m,r}^T \mathbf{W}^T \phi(\mathbf{x}_s^r) p(\mathbf{W}) d\mathbf{W} \\
&= \int \int \phi(\mathbf{x}_j^i)^T \mathbf{S}^{-1/2} \mathbf{Z} \mathbf{e}_{m,i} \mathbf{e}_{m,r}^T \mathbf{Z}^T \mathbf{S}^{-1/2} \phi(\mathbf{x}_s^r) p(\mathbf{Z}) p(\mathbf{S}) d\mathbf{Z} d\mathbf{S} \\
&= \int \phi(\mathbf{x}_j^i)^T \mathbf{S}^{-1/2} \mathbb{E}(\mathbf{Z} \mathbf{e}_{m,i} \mathbf{e}_{m,r}^T \mathbf{Z}^T) \mathbf{S}^{-1/2} \phi(\mathbf{x}_s^r) p(\mathbf{S}) d\mathbf{S} \\
&= \Psi_{ir} \int \phi(\mathbf{x}_j^i)^T \mathbf{S}^{-1} \phi(\mathbf{x}_s^r) p(\mathbf{S}) d\mathbf{S} \quad \text{(Using Lemma B.2)} \\
&= \Psi_{ir} \phi(\mathbf{x}_j^i)^T \mathbb{E}(\mathbf{S}^{-1}) \phi(\mathbf{x}_s^r) \\
&= \frac{\Psi_{ir} k(\mathbf{x}_j^i, \mathbf{x}_s^r)}{\nu - 2}. \qquad \text{(Using Lemma B.3)}
\end{aligned}
$$

Moreover, according to Lemma B.3, ν is required to be larger than 2.

APPENDIX C

DETAILS IN CHAPTER 6

In this section, we present the derivation of the noninformative Jeffreys prior in Eq. (6.8).

The prior on $\mathbf{W}$ is defined as follows:

$$\mathbf{w}^i \sim \mathcal{MGN}(0, \rho_i, q). \tag{C.1}$$

Then the noninformative Jeffreys prior for ρ_i can be calculated as

$$p(\rho_i) \propto \sqrt{I(\rho_i)} = \sqrt{\mathbb{E}_{\mathbf{w}^i|\rho_i}\Big[\Big(\frac{\partial \ln p(\mathbf{w}^i|\rho_i)}{\partial \rho_i}\Big)^2\Big]}. \tag{C.2}$$

Since

$$p(\mathbf{w}^i|\rho_i) = \frac{1}{\big(2\rho_i\Gamma(1+\frac{1}{q})\big)^m}\exp\Big\{-\frac{\sum_{j=1}^m |w_{ij}|^q}{\rho_i^q}\Big\},$$

we can get

$$\frac{\partial \ln p(\mathbf{w}^i|\rho_i)}{\partial \rho_i} = \frac{q\sum_{j=1}^m |w_{ij}|^q}{\rho_i^{q+1}} - \frac{m}{\rho_i}.$$

Then the Fisher information can be calculated as

$$\begin{aligned}
I(\rho_i) &= \mathbb{E}_{\mathbf{w}^i|\rho_i}\Big[\Big(\frac{\partial \ln p(\mathbf{w}^i|\rho_i)}{\partial \rho_i}\Big)^2\Big] \\
&= \mathbb{E}_{\mathbf{w}^i|\rho_i}\Big[\Big(\frac{q\sum_{j=1}^m |w_{ij}|^q}{\rho_i^{q+1}} - \frac{m}{\rho_i}\Big)^2\Big] \\
&= \frac{q^2}{\rho_i^{2q+2}}\Big\{\sum_{j=1}^m \mathbb{E}_{w_{ij}|\rho_i}\big[|w_{ij}|^{2q}\big] + \sum_{j=1}^m\sum_{k=1,k\neq j}^m \mathbb{E}_{w_{ij}|\rho_i}\big[|w_{ij}|^q\big]\mathbb{E}_{w_{ik}|\rho_i}\big[|w_{ik}|^q\big]\Big\} \\
&\quad + \frac{m^2}{\rho_i^2} - \frac{mq}{\rho_i^{q+2}}\sum_{j=1}^m \mathbb{E}_{w_{ij}|\rho_i}\big[|w_{ij}|^q\big].
\end{aligned}$$

So we need to calculate $\mathbb{E}_{w_{ij}|\rho_i}\big[|w_{ij}|^q\big]$ and $\mathbb{E}_{w_{ij}|\rho_i}\big[|w_{ij}|^{2q}\big]$ as follows:

$$\begin{aligned}
\mathbb{E}_{w_{ij}|\rho_i}\big[|w_{ij}|^q\big] &= \int_{-\infty}^{+\infty} \frac{1}{2\rho_i\Gamma(1+\frac{1}{q})} \exp\Big\{-\frac{|w_{ij}|^q}{\rho_i^q}\Big\}|w_{ij}|^q dw_{ij} \\
&= \int_0^{+\infty} \frac{1}{\rho_i\Gamma(1+\frac{1}{q})} \exp\Big\{-\frac{w_{ij}^q}{\rho_i^q}\Big\} w_{ij}^q dw_{ij} \\
&= -\int_0^{+\infty} \frac{\rho_i^{q-1}}{q\Gamma(1+\frac{1}{q})} w_{ij} d\exp\Big\{-\frac{w_{ij}^q}{\rho_i^q}\Big\} \\
&= -\frac{\rho_i^{q-1}}{q\Gamma(1+\frac{1}{q})}\Big[w_{ij}\exp\Big\{-\frac{w_{ij}^q}{\rho_i^q}\Big\}\Big|_0^{+\infty} - \int_0^{+\infty}\exp\Big\{-\frac{w_{ij}^q}{\rho_i^q}\Big\}dw_{ij}\Big] \\
&= \int_0^{+\infty} \frac{\rho_i^{q-1}}{q\Gamma(1+\frac{1}{q})} \exp\Big\{-\frac{w_{ij}^q}{\rho_i^q}\Big\} dw_{ij} \\
&= \frac{\rho_i^q}{q}\int_{-\infty}^{+\infty} \frac{1}{2\rho_i\Gamma(1+\frac{1}{q})} \exp\Big\{-\frac{|w_{ij}|^q}{\rho_i^q}\Big\} dw_{ij} \\
&= \frac{\rho_i^q}{q}
\end{aligned}$$

and

$$\begin{aligned}
\mathbb{E}_{w_{ij}|\rho_i}\big[|w_{ij}|^{2q}\big] &= \int_{-\infty}^{+\infty} \frac{1}{2\rho_i\Gamma(1+\frac{1}{q})} \exp\Big\{-\frac{|w_{ij}|^q}{\rho_i^q}\Big\}|w_{ij}|^{2q} dw_{ij} \\
&= \int_0^{+\infty} \frac{1}{\rho_i\Gamma(1+\frac{1}{q})} \exp\Big\{-\frac{w_{ij}^q}{\rho_i^q}\Big\} w_{ij}^{2q} dw_{ij} \\
&= \frac{-\rho_i^{q-1}}{q\Gamma(1+\frac{1}{q})} \int_0^{+\infty} w_{ij}^{q+1} d\exp\Big\{-\frac{w_{ij}^q}{\rho_i^q}\Big\} \\
&= \frac{-\rho_i^{q-1}}{q\Gamma(1+\frac{1}{q})}\Big[w_{ij}^{q+1}\exp\Big\{-\frac{w_{ij}^q}{\rho_i^q}\Big\}\Big|_0^{+\infty} - \int_0^{+\infty}\exp\Big\{-\frac{w_{ij}^q}{\rho_i^q}\Big\}dw_{ij}^{q+1}\Big] \\
&= \frac{(q+1)\rho_i^{q-1}}{q\Gamma(1+\frac{1}{q})} \int_0^{+\infty} w_{ij}^q \exp\Big\{-\frac{w_{ij}^q}{\rho_i^q}\Big\} dw_{ij} \\
&= \frac{(q+1)\rho_i^q}{q}\int_{-\infty}^{+\infty} \frac{1}{2\rho_i\Gamma(1+\frac{1}{q})} |w_{ij}|^q \exp\Big\{-\frac{|w_{ij}|^q}{\rho_i^q}\Big\} dw_{ij} \\
&= \frac{(q+1)\rho_i^q}{q}\mathbb{E}_{w_{ij}|\rho_i}\big[|w_{ij}|^q\big] \\
&= \frac{(q+1)\rho_i^{2q}}{q^2}.
\end{aligned}$$

So

$$\begin{aligned} I(\rho_i) &= \frac{m^2}{\rho_i^2} - \frac{m^2 q \rho_i^q}{\rho_i^{q+2} q} + \frac{q^2}{\rho_i^{2q+2}} \Big[\frac{m(q+1)\rho_i^{2q}}{q^2} + m(m-1)\frac{\rho_i^{2q}}{q^2} \Big] \\ &= \frac{m(q+m)}{\rho_i^2}. \end{aligned}$$

Finally the Jeffreys prior for ρ_i is formulated as

$$p(\rho_i) \propto \frac{1}{\rho_i}.$$

which is identical to the conventional one.