5-6-2016

Imprecisely Supervised Learning

Xin Wang
University of Connecticut - Storrs, xiw11003@engineer.uconn.edu

Imprecisely Supervised Learning

Xin Wang

University of Connecticut, 2016

ABSTRACT

Modern technologies have enabled us to collect large quantities of data. The proliferation of such data has facilitated knowledge discovery using machine learning techniques. However, it has also imposed great challenges for human annotators to label the massive data, which then offers proper supervision to learning algorithms. How we can improve learning accuracy with limited or imprecise supervision has recently become a much focused research subject. In this dissertation, we design effective algorithms based on modern machine learning theories to address the following two problems of: (1) learning from inconsistent labels collected from multiple annotators with varying expertise; and (2) knowledge transfer among related tasks to build more accurate predictive models. The proposed solutions will be evaluated not only on benchmark datasets but also in real-world scenarios from across disciplines.

In the first direction, we develop bi-convex optimization algorithms to address annotation ambiguity from inconsistent labels. We extend the well-known support vector machine (SVM) algorithm and optimize SVM classifiers with respect to a weighted consensus of different labelers' labels. The weights in the consensus are also automatically learned by our learning formulation. A variety of bi-convex programs are derived corresponding to different assumptions on the labeler competencies. Adding another layer of annotation ambiguity is that a labeler's label may be associated with a set of data points rather than each individual one. We then further generalize the formulation to deal with the multiple points' labels. Empirical results on benchmark datasets with synthetic labelers and real-life crowdsourced

labels demonstrate the superior performance of our methods over the state of the art.

Along the second direction, we develop new Multitask Learning (MTL) algorithms. MTL improves the generalization of the estimated models for multiple related learning tasks by capturing and exploiting the task relationships. We investigate a general framework of multiplicative multi-task feature learning which decomposes each task's model parameters into a multiplication of two components. One component is used across all tasks and the other is task specific. Several previous methods have been proposed as special cases of our framework. We prove that this framework is mathematically equivalent to the widely used multi-task feature learning methods that are based on a joint regularization of all model parameters, but with a more general form of regularizers. Two new learning formulations are proposed by varying the parameters in the proposed framework. We further study the method to learn task grouping with multiplicative feature sharing patterns in each group of tasks. We cluster tasks into one cluster if they select the same subset of features. We formulate an optimization problem to jointly optimize the models and grouping structure of the tasks. Empirical studies have revealed the advantages of our formulations by comparing with the state of the art.

Imprecisely Supervised Learning

Xin Wang

Master of Science, Dalian University of Technology, China, 2004

Bachelor of Engineering, Dalian University of Technology, China, 2008

A Dissertation

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Doctor of Philosophy

at the

University of Connecticut

2016

APPROVAL PAGE

Doctor of Philosophy Dissertation

Imprecisely Supervised Learning

Presented by

Xin Wang, B.E., M.S.

Major Advisor _____
Jinbo Bi

Associate Advisor _____
Alexander Russell

Associate Advisor _____
Laurent Michel

Associate Advisor _____
Fei Wang

University of Connecticut

2016

ACKNOWLEDGMENTS

My dissertation research was supported by multiple federal grants, including NSF grants IIS-1320586, DBI-1356655, and CCF-1514357, and an NIH grant R01DA037349-01A1 (PI: Jinbo Bi).

Over the past five years I have received a lot of support and encouragement from my advisors, collaborators, family members and my friends. I would like to express my sincere gratitude to all of these individuals.

Firstly, I offer my deepest appreciation and acknowledgement to my major advisor Dr. Jinbo Bi, who has supported me throughout my research work with her patience, motivation and immense knowledge. She has also been a mentor, colleague and friend of mine. Without her guidance, I would achieve nothing. I could not imagine having a better advisor for my Ph.D. study.

Besides, I would appreciate my associate advisors, Dr. Alexander Russell, Dr. Laurent Michel and Dr. Fei Wang, for their insightful comments and encouragement. Their advices and questions have inspired me to broaden my research in various perspectives. I am also thankful for Dr. Shipeng Yu, Dr. Minghu Song, Dr. Tulio Valdez, who gave me valuable guidance in my research.

My sincere thanks also go to Dr. Jiangwen Sun, Dr. Guoqing Chao, Tingyang Xu, Jin Lu, Aaron Palmer, Guannan Liang and all other collaborators. We have worked together for countless hours of discussions, derivations, programmings and proofreadings. I will remember those many sleepless nights that we were working hard in the lab and all the fun we have had in the past five years.

At last but not least, I would like to thank my family for supporting me throughout my

Ph.D. study and my life.

Contents

Chapter 1

Introduction

Machine learning traditionally assumes that supervision labels for collected samples are either known precisely (in supervised learning) or not known at all (in unsupervised learning). Supervised learning typically relies on a domain expert playing the role of a teacher to provide the necessary labels of the data. Unsupervised learning is a class of problems in which one seeks to determine how the data is organized. It is distinguished from supervised learning in that learning algorithms are given unlabeled samples only. Falling between the supervised learning and unsupervised learning is the semi-supervised learning, which typically uses a small amount of labeled data with a large amount of unlabeled data for training, aiming to improve the learning accuracy while keeping a low expense associated with the labeling process.

Recent technological innovations have enabled us to collect large quantities of data. The proliferation of such data has facilitated knowledge discovery and pattern prediction using machine learning techniques. However, it has also imposed great challenges for human annotators to label the massive data in order to offer proper supervision. Often times the

majority of collected data is either unlabeled or labeled with imprecise supervision.

How we can improve the learning accuracy with limited evidence or even imprecise supervision has recently become a much focused research subject. In this dissertation, we design effective algorithms based on modern machine learning theories to address the related problems, and evaluate the proposed solutions in real-world scenarios by collaborating with experts from across disciplines. There are two directions in my dissertation research as follows:

- Developing new optimization formulations to infer models from crowdsourcing labels that are collected inexpensively and time-efficiently, but accompany with imprecise labels.
- Developing new learning approaches to jointly tackle multiple related inference tasks that can improve the learning accuracy when only a limited amount of training data are available for each individual task.

1.1 Learning from the crowdsourcing labels

Ambiguous and inconsistent labels exist inevitably in crowdsourcing annotations, which brings a different set of machine learning problems associated with the efficient utilization, modelling and processing of imprecise supervision. Furthermore, data annotation becomes imprecise not only due to the expensive and time-consuming nature of the data labeling process, but also due to the difficulty and complexity of the practical problems themselves which hinder human annotators to acquire objective and reliable labels. The complex nature of the problem will make the analysis of imprecisely-labeled data an intensive research endeavour. Therefore, we design several effective ways to infer models from the crowdsourcing

labels provided by multiple annotators with different labeling expertise.

In particular, my study has been focused on the construction of classifiers utilizing multiple annotators' labels. This problem is challenging when the labeling accuracy and reliability of different labelers are unknown. Many existing methods target at the understanding and learning of the crowd behaviours. Those that actualy build classifiers typically impose a probabilistic model on the labeling process and then use an expectation-maximization (EM) algorithm to build logistic regression based classifiers. There has been limited effort in extending the widely-used support vector machines (SVM) to build classifier from crowd-annotated data. In this dissertation, we extend the discussion to the hinge loss commonly used by SVM, and develop bi-convex optimization based algorithms to construct classifiers and estimate the reliability of each annotator simultaneously. We modify the hinge loss by replacing true labels with the weighted combination of labelers' labels and the weights were determined by labeler reliabilities. We proposed three models, all of which followed a general principle that the labels from a more reliable labeler should contribute more to the determination of the classifier. If a labeler has a constant reliability factor, it represents an overall performance of the labeler for the task. For binary classification tasks, if a labeler has a predisposition to one class than the other, his/her reliability differs between the distinct classes, which brings a more complex reliability structure. The most complex structure assumes that a labeler's reliability varies on individual examples if the labeler is not equally competent to annotate different examples. We tested the proposed models on both benchmark and real-world biomedical datasets. The results demonstrated that our methods either outperformed or were competitive to the state of the art.

A more challenging problem is to learn classifiers from dual annotation ambiguity. Many pattern recognition problems confront two sources of annotation ambiguity where (1) crowdsourcing workers have provided multiple versions of a class label that may not be consistent

with one another, which forms multi-labeler learning; (2) and meanwhile a class label is associated with a bag of input vectors or instances rather than each individual instance and a bag is positive for a class label as long as one of its instances shows an evidence of that class, which is often referred to as multi-instance learning. Existing methods for multi-labeler learning and multi-instance learning only address one source of the labeling ambiguity. They are not trivially feasible to tackle the dual ambiguity problem. We hence develop a novel optimization framework by modifying the hinge loss to employ the weighted consensus of different labelers' labels and further generalizing the notion of loss functions to bags of multiple instances. The proposed formulation can be approximately solved by two mathematically tractable models that accommodate two types of labeling bias. The proposed algorithms were compared with several existing methods for multi-instance learning and those for multi-labeler learning that demonstrated superior performance on benchmark data sets collected for document classification, real-life crowd-sourced data sets, and a medical problem of heart wall motion analysis with diagnoses from multiple radiologists.

1.2 Jointly learning of multiple related tasks

Constructing models jointly for multiple related tasks is normally referred as the multi-task learning (MTL), which is the methodology to improve the generalization of the estimated models for multiple related learning tasks by capturing and exploiting their relationships. It has been theoretically and empirically shown to be more effective than learning each task independently. Especially when the single task learning suffers from limited sample size, multitask learning reinforces a single learning process with the transferable knowledge

learned from the related tasks. Multi-task learning has been widely applied in many fields ranging from robotics [113], natural language processing [3], computer aided diagnosis [13], computer vision [51] and so on.

Along this direction, we propose and investigate a general framework of multiplicative multitask feature learning which decomposes each task's model parameters into a multiplication of two components. One of the components is used across all tasks and the other component is task-specific. Several previous methods are special cases of the proposed framework. We study the theoretical properties of this framework when different regularization conditions are applied to the two decomposed components. We prove that this framework is mathematically equivalent to the widely used multitask feature learning methods that are based on a joint regularization of all model parameters, but with a more general form of regularizers. Further, an analytical formula is derived for the across-task component as related to the task-specific component for all these regularizers, leading to a better understanding of the shrinkage effect of different regularizers. Study of this framework motivates new multitask learning algorithms. We propose two new learning formulations by varying the parameters in the proposed framework. Empirical studies have revealed the relative advantages of the two new formulations by comparing with the state of the art, which provides instructive insights into the feature learning problem with multiple tasks.

Since the effectiveness of multitask learning relies on the degree of relatedness among the tasks, it is essential to identify the correct groups where tasks in the same group are highly related and suitable for joint learning. Hence, we further investigate the problem of grouping related tasks so that they share the same features within each group. We propose a new concept that tasks in a group may use a set of features but may have their own weightings on these features. A task's model parameter vector is still decomposed into a component-wise product of two vectors. One vector is used across the entire group of tasks to select features

for the group and the other vector uses the selected features and specifies the weightings of these features in each task's model. We thus group tasks according to whether they share the same across-task component. This approach helps to recover many sharing patterns that are difficult for other methods to discover. The decomposed components are regularized differently according to hypothesized feature sharing structure. We formulated an optimization problem that jointly identifies the task grouping structure and estimates the parameters for individual models. The decomposed components could be regularized differently according to hypothesized feature sharing structure. Empirical evaluations using both synthetic and real-world datasets demonstrated that our method outperformed several latest clustered multi-task learning methods.

Chapter 2

Bi-convex Optimization to Learn Classifiers from Multiple Annotators

Learning from multiple labelers who provide inconsistent annotations to the training data is an emerging machine learning problem. Recent technological innovations have created easily accessible crowdsourcing platforms such as Amazon Mechanical Turk (AMT)[1] and Crowdflower[2], through which tasks such as text or image annotations can be assigned to enormous online annotators at affordable prices. These crowdsourcing methods largely reduce the economic and time costs associated with massive annotating tasks. However, they have imposed great technical challenges in deriving and modeling ground truth in many cases. For instance, to diagnose cancer, although a biopsy provides ground truth, the procedure is complicated and combines with discomforts. Hence, series of X-ray images can instead be read and annotated by multiple radiologists to facilitate the diagnosis. An early work in cancer research reported the problem that different radiologists have different reliabilities

[1] https://www.mturk.com/
[2] http://www.crowdflower.com/

of recognizing a lesion in the same diagnostic image [39]. Since one expert's expertise may be biased and/or incomplete, integration of knowledge from multiple experts is necessary to build accountable and reliable cancer informatics systems. Learning from multiple annotators has become necessary and beneficial in a variety of fields. In the bioinformatics field, nature language processing (NLP) tasks have been performed by AMT workers to extract knowledge from biological and medical documents [109, 18, 31]. A recent work [69] recruited non-expert AMT workers to annotate CT images with little prior training and then had to use integration such as majority voting strategies to detect polyps (colon cancer).

The traditional learning task constructs a classifier mapping from input features to ground truth labels, which becomes difficult in the scenarios where ground truth labels are unknown and need to be estimated from multiple annotators of different expertise. Figure 2.1 illustrates the learning problem using an example of heart motion analysis. Multiple radiologists annotate a set of echocardiograms in terms of whether an image shows abnormal heart motion. The labels from these radiologists do not agree. The goal is to train a classifier that utilizes the inconsistent radiologist annotations to predict unseen images. The echocardiograms are very difficult to interpret even for the best physicians [76]. Inter-observer studies showed that even world-class experts only agreed on 80% of their diagnoses. The learning problem described in Figure 2.1 is especially difficult when radiologists' expertise and reliability are unknown.

Several methods have been proposed in the recent machine learning literature to learn models from crowds, or more precisely, from crowdsourced labels [44]. These methods typically impose a probabilistic model on the labeling process, such as Bernoulli model or Gaussian model on the true labels [107], or two-coin model for annotators [83, 82], and then use an expectation-maximization (EM) process to build logistic regression classifiers. Two recent works [47, 48] also propose convex formulations based on logistic regression, but the

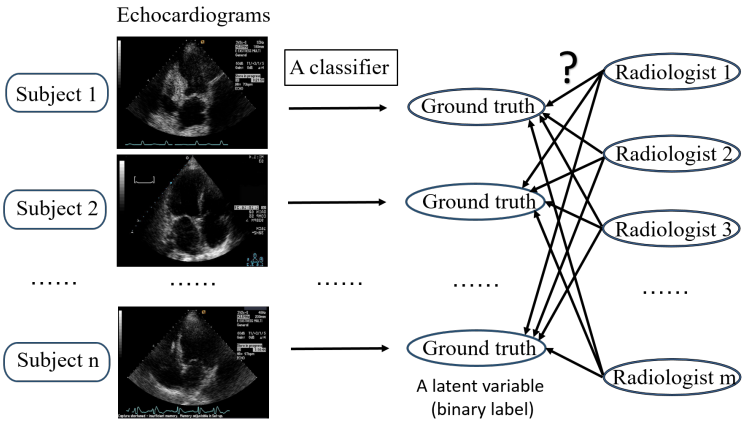


Figure 2.1: Classifier training from multiple annotators. Echocardiograms of n subjects are annotated by m radiologists, and the ground truth for each image is unknown. A classifier is constructed to map an image to its ground truth label that is estimated from the different radiologist labels.

true classifier is estimated by taking an average effect of the classifiers trained with each labeler, which may be impacted significantly by malicious labelers or spammers. There has been limited effort in extending support vector machines (SVM) to build classifiers from crowd-annotated data. It has been shown that SVM may bear some advantages over logistic regression when data follows certain distribution such as multivariate or mixture of distributions, and SVM methods may require less features than logistic regression to achieve a better or equivalent classification accuracy [75, 85, 99].

In this study, we propose a bi-convex optimization approach that performs simultaneously three tasks: (1) assess how good each labeler is, (2) estimate the true labels, and (3) build a classifier using approximate true labels estimated from the multiple labels. The key step is to modify the hinge loss used in the SVM where the unknown true labels are replaced by their estimates. In the proposed approach, we associate each labeler with a reliability factor. Three learning models, each forming a bi-convex program, are derived by making the hinge loss

reflect three different kinds of assumptions on the labeler reliability. The proposed methods follow a general principle that the labels from a more reliable labeler should contribute more to the construction of the classifier. If a labeler has a constant reliability factor, it represents an overall performance of the labeler for the task. For binary classification tasks, if a labeler has a predisposition to one class than the other, his/her reliability differs between the distinct classes, which brings a more complex reliability structure. The most complex one assumes that the labeler reliability varies on individual examples if the labeler is not equally competent to annotate different examples.

2.1 Related works

Many existing methods for *learning from crowds* focus on modeling of an annotation process and estimating error rates for the labelers independent of any classifiers. The early statistical methods [39, 23, 2] on error rate estimation for repeated but conflicting test results, and the recent work on learning crowd behaviors [92, 118, 117], are good examples. The latest work in this direction ranks annotators to identify spammers [81], uses Multinomial probabilistic models to quantify the competency of each labeler [57], and parameterizes labeler expertise or reliabilities as well as the difficulty of an annotation task to model human annotation more accurately [72, 95, 41]. Moreover, in the work of [52] and [38], reliabilities are estimated from gold standard tasks and then used in a weighted combination for new labeling tasks. Another method in [90] models the labelers using a stochastic model and select examples to teach the labelers via a greedy algorithm. These methods study the problem of optimizing the task assignment in a crowdsourcing system. We adopt the similar strategy as [52] and [38] to aggregate the inconsistent labels assigned by multiple labelers, but the labeler reliabilities

are jointly estimated with a classifier.

Recently the interest of learning from crowds has increased to directly build classifiers from multi-labeler data. Repeated labeling methods [91, 89] identify the labels that should be reacquired from some labelers in order to improve classification performance or data quality. A recent theoretical work [24], however, argues that the repeated labeling negatively impacts the relative size of the training sample. Another set of approaches [15, 21] assume the existence of prior knowledge relating the different labelers, and the prior is used to identify the samples for each labeler that are appropriate to be used in the classifier estimation. Several methods [44, 107, 83, 82, 47, 48, 106, 26], however, neither assume that labels can be reacquired, nor assume existence of any prior on labeler relations. These approaches rely on certain data distribution, such as Bernoulli model on the true binary labels or Gaussian model on the true continuous labels [107] or two-coin model on the process of how an annotator provides a label [83, 82], and then develop a posterior solution with logistic regression and use an EM algorithm to estimate the model parameters.

Among the methods that build a classifier and estimate labelers' error rates simultaneously, the models of [47, 48] and [107] are the most similar to our work. In [47, 48], a linear classifier with coefficients $\mathbf{w}_j$ is built for each individual labeler j based on his/her own annotation using logistic regression and the final classifier with a coefficient vector $\mathbf{w}$ is obtained by enforcing a regularization term, that is either $\sum_j ||\mathbf{w}_j - \mathbf{w}||^2$ in [47] or $\sum_{j,k} ||\mathbf{w}_j - \mathbf{w}_k||^2$ where j, k denotes the indexes of the classifiers constructed from an individual labeler's annotation [48]. The final classifier ($\mathbf{w}$) is hence constructed by taking an average effect of individual labeler's classifiers rather than by minimizing the final classifier's own loss on the training data. This classifier may collapse if there are many malicious labelers due to the kind of majority voting effect. In [107], it is assumed that a labeler's competence may vary when annotating different sample points, so a classifier is built for each labeler to param-

eterize his/her reliability on an example. Then the final classifier is built by modeling the reliabilities of the different labelers in a logistic regression based EM algorithm. Unlike this method, we impose no specific distributions but more general and intuitive assumptions on the labelers' reliabilities.

2.2 The Proposed Formulations

We derive the learning formulations in this section. Let $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$ comprise the n examples, where $\mathbf{x}_i \in R^d$, and is annotated with multiple versions of the label $\{y_i^1, y_i^2, \cdots, y_i^m\}$. We focus on the case of binary classification where $y_i^j \in \{-1, 1\}$, $j \in \{1, 2, ..., m\}$. Suppose that the true label of $\mathbf{x}_i$ is y_i and we consider linear models of the form $\mathbf{x}^\top \mathbf{w} + b$ where $\mathbf{w}$ is the weight vector and b is the offset to be determined for the classifier. We derive our models by modifying the hinge loss $[1 - y_i(\mathbf{x}_i^\top \mathbf{w} + b)]_+ = \max\{0, 1 - y_i(\mathbf{x}_i^\top \mathbf{w} + b)\}$ where we replace the unknown true label y_i by a linear combination of y_i^j.

The use of a linear combination of y_i^j as an approximate of y_i is rooted from a probabilistic understanding of the learning problem. For instance, in the model with constant labeler reliability derived in the next section, the essential motivation is that the true (unobserved) label y_i is a linear combination of y_i^j's that are *i.i.d.* sampled from the hidden true y_i, taking a Gaussian form $y_i^j \sim \mathcal{N}(y_i, \sigma_j^2)$ where y_i is the mean and σ_j^2 is the precision. Then the *a posteriori* distribution of y_i given all the observed labels follows $\mathcal{N}(\mu_i, \sigma_i^2)$, where $\mu_i = \sum_j \sigma_j^2 y_i^j / \sum_j \sigma_j^2$, and $\sigma_i^2 = \sum_j \sigma_j^2$. So one can see *a posteriori* mean is a weighted linear combination of all observed labels, and the weights sum to 1.

2.2.1 The model with constant labeler reliability

We approximate an example's true label y_i by a weighted combination of each labeler's labels, e.g., $y_i \simeq \sum_{j=1}^{m} r_j y_i^j$ and each labeler j is associated with a reliability factor r_j where $0 \leq r_j \leq 1$. If the reliability factors of all labelers are equal, this combination amounts to the majority voting. If we require additionally $\sum_j r_j = 1$, we approximate y_i by a convex combination of labelers' opinions. We believe these combinations may all be reasonable, and the most appropriate one may be problem-specific. If the weighted consensus of all labelers $\sum_j r_j y_i^j > 0$, the example i is more likely to be in the class of $y = 1$; or otherwise, it likely has a true label of $y = -1$.

We modify the hinge loss by replacing the true labels y_i by the weighted consensus, which yields a bi-convex function $[1 - (\sum_j r_j y_i^j)(\mathbf{x}_i^\top \mathbf{w} + b)]_+$ (convex with respect to $(\mathbf{w}, b)$ for fixed $\mathbf{r}$ and convex with respect to $\mathbf{r}$ for fixed $(\mathbf{w}, b)$). When the consistency is high among the labels given by different labelers, especially by reliable labelers, the magnitude of $\sum_j r_j y_i^j$ tends to be large regardless of its sign, showing high annotation confidence for $\mathbf{x}_i$. Minimizing the modified loss leads to a classifier that works hard to correctly classify $\mathbf{x}_i$. When the labeling consistency is low among reliable labelers for some examples, the assignment of them to either class can be a vague guess. The linear combination of labels will lead to a small value in magnitude due to the cancellation effect of the mixed $+1$ and -1 labels. The modified loss then reports a low value on such cases, which hence does not emphasize the classification performance on these examples. This justifies the validity of the modified loss.

By adding a regularization term $||\mathbf{w}||^2$ to the empirical loss, we minimize the following

optimization problem

$$\begin{array}{ll} \min\limits_{\mathbf{w},b,\mathbf{r}} & \lambda||\mathbf{w}||^2 + \sum_i [1 - (\sum_j r_j y_i^j)(\mathbf{x}_i^\top \mathbf{w} + b)]_+ \\ \text{s.t.} & \sum_j r_j = 1, \quad r_j \geq 0, \\ & i = 1, 2, ..., n, \quad j = 1, 2, ..., m. \end{array} \tag{2.1}$$

The constraints on $\mathbf{r}$ are affine, which formulate the convex combinations of labelers' opinions and enforce competition among the labelers by limiting the sum of their reliabilities to a constant 1. It is easy to verify that Problem (2.1) is a case of bi-convex optimization because the objective function is bi-convex and constraints are affine. To translate the problem into a canonical form, the modified loss is translated into constraints $(\sum_j r_j y_i^j)(\mathbf{x}_i^\top \mathbf{w} + b) \geq 1 - \xi_i$ for each example i where the slack variables $\xi_i \geq 0$, and both $\mathbf{r}$ and $(\mathbf{w}, b)$ are variables to be determined in the following optimization problem

$$\begin{array}{ll} \min\limits_{\mathbf{w},b,\boldsymbol{\xi},\mathbf{r}} & \lambda||\mathbf{w}||^2 + \sum_i \xi_i \\ \text{s.t.} & (\sum_j r_j y_i^j)(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i, \\ & \sum_j r_j = 1, \quad r_j \geq 0, \quad \xi_i \geq 0, \\ & i = 1, 2, ..., n, \quad j = 1, 2, ..., m. \end{array} \tag{2.2}$$

Problem (3.2) is also a quadratically constrained quadratic optimization problem but with one of its constraints bi-convex.

For binary classification, if the training data is balanced in the distribution of either class (e.g., close to even numbers of positive and negative examples), the proposed model with constant reliability is often sufficient to estimate a labeler's overall reliability. This is sometimes referred to as a one-mode model. The labelers with higher reliabilities are expected to be assigned with larger weights by solving Problem (3.2). However, this one-

mode model will hardly take care of the situation when labelers have different labeling accuracies with respect to the positive or negative class labels. In practice, when the problem data is very unbalanced, the true positive rate and true negative rate will be important factors to reflect the real performance. A model considering a labeler's class-dependent reliability will be needed.

2.2.2 The model with class-dependent labeler reliability

A labeler's reliability may naturally be class dependent. For instance, online annotators may have different accuracies in labeling documents with respect to different topics relying on whether they are more familiar with some topics than others. If a labeler tends to always label examples to the +1 class, his positive predictive value (PPV) (the percentage of examples labeled by the labeler as positive that are actually positive) may be low but his negative predictive value (NPV) (the percentage of examples labeled by the labeler as negative that are actually negative) can be high.

We extend the model discussed in Section 2.2.1 to class-dependent reliability factors. The model still estimates the true labels by the weighted combination of each labeler's labels. However, two parameters $\alpha_j \geq 0$ and $\beta_j \geq 0$ are needed to estimate the j-th labeler's PPV and NPV, respectively. We set the true labels $y_i \simeq \sum_{j=1}^{m} \alpha_j^{(1+y_i^j)/2} \beta_j^{(1-y_i^j)/2} y_i^j$. If labeler j gives $y_i^j = +1$, α_j should be used as the corresponding weight for y_i^j, and β_i^j is degraded to 1. If $y_i^j = -1$, β_i^j should be used in the combination. Unlike the constant reliability model, we now require $\sum_j \alpha_j = 1$ and $\sum_j \beta_j = 1$, that can enforce competition among labelers. When the two parameters are used in the modified hinge loss, we optimize the following

optimization problem for the best class-dependent reliabilities and classifier

$$\begin{aligned}
&\min_{\mathbf{w},b,\boldsymbol{\xi},\mathbf{r}} \lambda||\mathbf{w}||^2 + \sum_i [1 - (\sum_{j=1}^{m} \alpha_j^{\frac{1+y_i^j}{2}} \beta_j^{\frac{1-y_i^j}{2}} y_i^j)(\mathbf{w}^\top \mathbf{x}_i + b)]_+ \\
&\text{s.t.} \qquad \sum_j \alpha_j = 1, \quad \sum_j \beta_j = 1, \\
&\alpha_j \geq 0, \quad \beta_j \geq 0, \\
&i = 1, 2, ..., n, \quad j = 1, 2, ..., m.
\end{aligned} \tag{2.3}$$

Bi-convexity still holds for Problem (2.3) since α_j and β_j are not used at the same time for each y_i^j in the constraints given the values of y_i^j are already known. The same optimization algorithm used to solve Problem (2.1) can be applied to solve Problem (2.3). Problem (2.3) can also be translated into a canonical form by utilizing slack variables ξ to represent the hinge losses, and the resultant optimization problem is written as follows:

$$\begin{aligned}
&\min_{\mathbf{w},b,\boldsymbol{\xi},\mathbf{r}} \lambda||\mathbf{w}||^2 + \sum_i \xi_i \\
&\text{s.t.} \quad (\sum_{j=1}^{m} \alpha_j^{(1+y_i^j)/2} \beta_j^{(1-y_i^j)/2} y_i^j)(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i, \\
&\sum_j \alpha_j = 1, \quad \sum_j \beta_j = 1, \\
&\xi_i \geq 0, \quad \alpha_j \geq 0, \quad \beta_j \geq 0, \\
&i = 1, 2, ..., n, \quad j = 1, 2, ..., m.
\end{aligned} \tag{2.4}$$

This two-mode model, is similar in spirit, to the two-coin model used in [83, 82], where a labeler's expertise was also described by two factors, sensitivity and specificity. In [83, 82], labelers' expertise, true labels and the classifier were learnt with an EM algorithm based on logistic regression. However, we observe that this prior model can become numerically unstable when a large number of laberlers are present [12]. The two-coin model updates the estimated ground truth denoted by μ, a probability of the true label being +1, based on

the multiplications of sensitivities and specificities, denoted by $0 \leq \alpha_j \leq 1$ and $0 \leq \beta_j \leq 1$ for labeler j respectively (with a little mixed use of notation). The products $\Pi_{j=1}^{m}\alpha_j$ and $\Pi_{j=1}^{m}\beta_j$, assuming there are m labelers, become extremely small with large m. Consequently, μ becomes oscillating between 0 and 1 since the two products are used in the numerator and denominator of the updating formula for μ.

The proposed model is instead scalable and reliable to deal with a large number of labelers. By selecting high-quality labelers for use in the combination, redundant labelers may be excluded and sparsity has been observed in the estimated reliabilities when a large number of labelers are included. Our empirical results show that, for both the models with constant reliabilities and class-dependent reliabilities, the true labels can be sufficiently estimated from few labelers and information from other labelers might be redundant. Our models could automatically select labelers whose labels were valid to make accurate estimates of the ground truth and exclude correlated or redundant labelers.

2.2.3 The model with sample-specific labeler reliability

If a labeler is not equally competent to annotate all sample subjects, his/her reliability r will become a factor relying on individual samples $\mathbf{x}$ and hence becomes a function of $\mathbf{x}$ as $r(\mathbf{x})$. Such an issue often takes place in real life applications. Radiologists may not have the equal reliability dealing with high-quality images versus images of different kinds of noise. Few previous studies examined this practical difficulty. The methods in [107, 106] built a classifier $\mathbf{x}^\top \mathbf{w}_j + b_j$ for each labeler j based on his/her own annotation, and defined $r_j(\mathbf{x})$ as a sigmoid translation $(1 + \exp(-(\mathbf{x}^\top \mathbf{w}_j + b_j)))^{-1}$ of the linear classifier. The probability of observing y_i^j was $p(y_i^j) = (1 - r_j(\mathbf{x})^{|y_i^j - y_i|} r_j(\mathbf{x})^{1-|y_i^j - y_i|}$. Due to the use of sigmoid functions and absolute values in the exponent, it has created complex optimization problems.

We will use the functional distance from each $\mathbf{x}_i$ to the separation boundary ($\mathbf{x}^\top \mathbf{w}_j + b_j = 0$) that is computed from a labeler j's annotation to determine the labeler's confidence on labeling $\mathbf{x}_i$ relative to other labelers. The farther from the separation boundary, the more confident the labeler annotates the example. Hence, the reliability function $r_j(\mathbf{x}_i) = |\mathbf{x}_i^\top \mathbf{w}_j + b_j| / \sum_j |\mathbf{x}_i^\top \mathbf{w}_j + b_j|$. The denominator is used to compute the jth labeler's confidence relative to other labelers' confidence. The individual labelers' classifiers can be built by minimizing standard hinge loss defined as $\eta_i^j = [1 - y_i^j(\mathbf{x}_i^\top \mathbf{w}_j + b_j)]_+$. Since these classifiers are used to determine reliabilities, they should be constructed more or less accurately (i.e., close to the final classifier determined by $\mathbf{w}$), which motivates to impose an additional regularizer $R(\mathbf{w}, \mathbf{w}_j) = \sum_j ||\mathbf{w} - \mathbf{w}_j||^2$ assuming the final classifier is a more accurate estimate of the true classifier. Importantly, this regularizer will enforce individual labeler's classifiers to have similar $||\mathbf{w}_j||$. Because $r_j(\mathbf{x}_i)$ is defined through a functional distance from $\mathbf{x}_i$ to the boundary, the similarity among different $||\mathbf{w}_j||$ will render that $r_j(\mathbf{x}_i)$ is largely proportional to the geometric (Euclidean) distance from the point to the boundary.

We define the modified hinge loss $\xi_i = [1 - (\sum_j \frac{(|\mathbf{x}_i^\top \mathbf{w}_j + b_j|}{\sum_j (|\mathbf{x}_i^\top \mathbf{w}_j + b_j|} y_i^j)(\mathbf{x}_i^\top \mathbf{w} + b)]_+$ and the additional constraint for ξ_i would be $(\sum_j \frac{(|\mathbf{x}_i^\top \mathbf{w}_j + b_j|}{\sum_j (|\mathbf{x}_i^\top \mathbf{w}_j + b_j|} y_i^j)(\mathbf{x}_i^\top \mathbf{w} + b) \geq 1 - \xi_i$. The modified hinge loss appears complex. However, it can be re-organized through simple algebraic calculations by moving the denominator in $r_j(\mathbf{x}_i)$ to the right-hand side. After re-organization, this constraint becomes $\sum_j |\mathbf{x}_i^\top \mathbf{w}_j + b_j|(y_i^j((\mathbf{x}_i^\top \mathbf{w} + b) - (1 - \xi_i)) \geq 0$. The use of the absolute value of $\mathbf{x}_i^\top \mathbf{w}_j + b_j$ can complicate the optimization problem. Hence, we replace the absolute value by the upper bound $u_i^j \geq 0$ that satisfies the constraints: $-u_i^j \leq \mathbf{x}_i^\top \mathbf{w}_j + b_j \leq u_i^j$ for all $i = 1, \cdots, n$ and $j = 1, \cdots, m$. Overall, we search for the best classifier $(\mathbf{w}, b)$ and the most

accurate reliability estimate based on $(\mathbf{w}_j, b_j)$ by optimizing the following problem:

$$\begin{aligned}
\min_{\mathbf{w},b,\boldsymbol{\xi},\mathbf{w}_j,b_j,\boldsymbol{\eta},\mathbf{u}} \quad & \lambda_1||\mathbf{w}||^2 + \lambda_2 \sum_j ||\mathbf{w} - \mathbf{w}_j||^2 \\
& + \sum_i \xi_i + \sum_i \sum_j \eta_i^j + \lambda_3 \sum_i \sum_j u_i^j \\
\text{s.t.} \quad & \sum_j u_i^j (y_i^j (\mathbf{x}_i^\top \mathbf{w} + b) - (1 - \xi_i)) \geq 0, \\
& -u_i^j \leq \mathbf{x}_i^\top \mathbf{w}_j + b_j \leq u_i^j, \\
& y_i^j (\mathbf{x}_i^\top \mathbf{w}_j + b_j) \geq 1 - \eta_i^j, \\
& \xi_i \geq 0, \quad \eta_i^j \geq 0, \quad u_i^j \geq 0, \\
& i = 1, 2, ..., n, \quad j = 1, 2, ..., m.
\end{aligned} \tag{2.5}$$

Problem (2.5) has a convex objective function, a bi-affine constraint (the first constraint) and all other constraints are affine. This problem is also a bi-convex program. We can group the variables into two groups: one group is related to the final classifier including variables $\mathbf{w}, b, \boldsymbol{\xi}$, and the other group is related to individual classifiers including variables $\mathbf{w}_j, b_j, \boldsymbol{\eta}_j, \mathbf{u}$. When fixing one group of the variables, Problem (2.5) becomes a convex quadratic program in terms of the other group of variables.

Besides the regularizer that enforces the similarity between individual $\mathbf{w}_j$'s and the final classifier's $\mathbf{w}$, an additional regularizer $||\mathbf{w}_j||^2$ can be directly applied to individual $\mathbf{w}_j$. We will also evaluate an alternative formulation by revising the objective function in Problem

(2.5) to

$$\begin{aligned}\min_{\mathbf{w},b,\boldsymbol{\xi},\mathbf{w}_j,b_j,\boldsymbol{\eta},\mathbf{u}} \quad & \lambda_1||\mathbf{w}||^2 + \lambda_2 \sum_j ||\mathbf{w}-\mathbf{w}_j||^2 \\ & + \lambda_3 \sum_j ||\mathbf{w}_j||^2 + \sum_i \xi_i + \sum_i \sum_j \eta_i^j \\ & + \lambda_4 \sum_i \sum_j u_i^j \end{aligned} \tag{2.6}$$

where the same constaints in Problem (2.5) apply.

Besides the prior methods in [107, 106], the methods in [47, 48] also estimate a labeler's reliability by building a classifier $(\mathbf{w}_j, b_j)$ from the labeler's own annotations. However, the final classifier $(\mathbf{w}, b)$ was built by minimizing $\sum_j ||\mathbf{w}-\mathbf{w}_j||$, and hence $\mathbf{w}$ is estimated as the centroid of all $\mathbf{w}_j$'s, and thus suffering from significant outlier labelers.

2.3 The Optimization Algorithm

In this section, we present an effective algorithm to solve the proposed Problems (2.1), (2.3), (2.5) and (2.6) respectively. We adopt the alternating optimization approach for each problem where we alternate between solving for the final classifier and solving for the parameters related to reliability iteratively until the algorithm converges. Because the three proposed problems are all bi-convex as discussed in the last section, the subproblems formulated for solving each group of variables is convex. To solve the subproblems in (2.1) and (2.3), we used their equivalent formulations (3.2) and (2.4) by introducing slack variables. Algorithm 1 summarizes the algorithmic procedure for each of the problems. For illustration convenience, we list the procedure for each of the problems together in Algorithm 1.

The initialization choices listed in Algorithm 1 are believed to follow the most common

Algorithm 1 Alternating optimization algorithm for the proposed bi-convex programs

Input: λ's, $\mathbf{X}$, $\mathbf{y}^j$, $j = 1, \cdots, m$, and a tolerance ϵ.
Initialize: let k denote the current number of iterations, and k is initialized to 0. Let Θ denote the set of all variables needed to be optimized.
For the constant reliability model, set $\mathbf{r}^{(0)} = 1/m$. For the class-dependent reliability model, set $\boldsymbol{\alpha}^{(0)} = 1/m$ and $\boldsymbol{\beta}^{(0)} = 1/m$. For the sample-specific reliability model, set $(\mathbf{w}_j^{(0)}, b_j^{(0)})$ to the SVM classifiers that are built from each labeler's labels, $j = 1, \cdots, m$.
repeat
Step 1:
For Problem (2.1), solve Problem (3.2) for $(\mathbf{w}^{(k)}, b^{(k)})$ and $\boldsymbol{\xi}$ with fixed $\mathbf{r}^{(k-1)}$.
For Problem (2.3), solve Problem (2.4) for $(\mathbf{w}^{(k)}, b^{(k)})$ and $\boldsymbol{\xi}$ with fixed $\boldsymbol{\alpha}^{(k-1)}$ and $\boldsymbol{\beta}^{(k-1)}$.
For Problem (2.5) and (2.6), solve for $(\mathbf{w}^{(k)}, b^{(k)})$ and $\boldsymbol{\xi}$ with fixed $(\mathbf{w}_j^{(k-1)}, b_j^{(k-1)})$, $\boldsymbol{\eta}_j^{(k-1)}$, and $\mathbf{u}_j^{(k-1)}$.
Step 2:
For Problem (2.1), solve Problem (3.2) for $\mathbf{r}^{(k)}$ and $\boldsymbol{\xi}$ with fixed $(\mathbf{w}^{(k)}, b^{(k)})$.
For Problem (2.3), solve Problem (2.4) for $\boldsymbol{\alpha}^{(k)}$ and $\boldsymbol{\beta}^{(k)}$ with fixed $(\mathbf{w}^{(k)}, b^{(k)})$.
For Problem (2.5) and (2.6), solve for $(\mathbf{w}_j^{(k)}, b_j^{(k)})$, $\boldsymbol{\eta}_j^{(k)}$, and $\mathbf{u}_j^{(k)}$ with fixed $(\mathbf{w}^{(k)}, b^{(k)})$ and $\boldsymbol{\xi}^{(k)}$.
until $||\Theta^{(k)} - \Theta^{(k-1)}||^2 \leq \epsilon$.
Output: $(\mathbf{w}, b)$, and $\mathbf{r}$, (or $(\boldsymbol{\alpha}, \boldsymbol{\beta})$, or $(\mathbf{w}_j, b_j)$).

sense. For instance, without prior knowledge, we may assume that all labelers are equally competent (with equal initial $\mathbf{r}$, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$) and let the algorithm determine and update the reliability factors based on sample data. We also empirically notice that the algorithm is insensitive to initial values in the sense that it gives the same solution when we perturb the listed initial values by random white noise.

We point out a small derivation difference in solving the three problems. Problems (2.1) and (2.3) can be solved using the same split of working variables, that is the algorithm optimizes either $(\mathbf{w}, b)$ or $\mathbf{r}$ (or $(\boldsymbol{\alpha}, \boldsymbol{\beta})$) in an alternating step. The slack variables ξ_i in Problems (3.2) and (2.4) are only used to update the hinge loss in Problems (2.1) and (2.3), respectively, at each step and hence they are included in both the working groups of variables. For the sample-specific reliability model, the modified hinge loss is not bi-convex by its literal

form. We have reformulated the problem using the first constraint in Problems (2.5) and (2.6). In this situation, we group the variables ξ_i with the final classifier parameters $(\mathbf{w}, b)$.

According to the convergence analysis in [9, 33], the alternating Algorithm 1 used for solving our programs (2.1), (2.3), (2.5) and (2.6) converges to a set of fixed points which in general includes global minimizers, local minimizers and the saddle points. Due to the bi-convexity, the fixed points of our programs do not include saddle points [33].

We give a brief discussion on the complexity of Algorithm 1 using Problems (2.1) and (2.3). To optimize for $(\mathbf{w}, b)$, Algorithm 1 solves a quadratic program similar to SVM. To solve for $\mathbf{r}$ (or $(\boldsymbol{\alpha}, \boldsymbol{\beta})$), Algorithm 1 solves a linear program. Effective algorithms such as Dantzig's simplex method, or later interior point methods have been developed for these programs. In our implementation, we used the CPLEX software to solve them with choices of simplex-based methods. Rigorous bounds on the number of operations required by these methods have been established. For instance, the complexity of solving the linear program is in order of $d^2\ell$ with a constant where d is the number of variables and ℓ is the number of constraints in the program. We observe that Algorithm 1 typically stops within 10 iterations, so the overall complexity of Algorithm 1 is a constant (e.g., 10) multipying the sum of the complexity of solving the linear and quadratic programs.

2.4 Experiments

The proposed methods were tested on five benchmark datasets at first. Four of them are commonly used in evaluating machine learning algorithms. These datasets are all for binary classification and come with ground truth labels, but they are not labeled by multiple annotators. We created synthetic labelers for these datasets. The fifth benchmark dataset is

the facial expression dataset where each face shot image was labeled by multiple real online workers. Besides the benchmark datasets, we also tested the proposed methods on three real-world problems of analyzing biomedical images. The first problem was to detect breast cancer in digitized mammographic images. The other two problems aimed respectively to detect Heart Wall Motion Abnormality (HWMA) using features extracted from echocardiogrmas and to diagnose Alzheimer's disease using features extracted from magnetic resonance imaging (MRI).

Our methods were compared against four recently-published methods, all of which construct classifiers: Two-coin model [82], EMGaussian model and EMBernoulli model [107], and a convex model [47]. The classifier trained with ground truth labels was supposed to achieve the best performance whereas the majority voting approach served as our baseline. The proposed methods *Model with Constant Reliability, Model with Class-Dependent Reliability and Model with Sample-Dependent Reliability* were respectively referred to as MCR, MCDR and MSDR.

Ten-fold cross validation (CV) was used to run all algorithms on each of the datasets with the same stratified CV split. For the proposed methods and the convex method in [47], we tuned their regularization parameters within the training data in the first CV fold using another internal three-fold CV for each dataset and then fixed the parameters for the nine remaining CV folds. We selected the parameters that obtained the best performance from the range of $[10^{-3},10^{-2},\cdots,10^{3}]$.

2.4.1 Benchmark datasets

In this section, we provide the details on the experimental procedure and results obtained on the benchmark datasets.

Table 2.1: Details of the UCI benchmark datasets with synthetic labelers

Dataset	Cases(positive)	Features
Cleveland	303 (139)	13
Glass	214 (163)	9
Ionosphere	351 (126)	33
Pima	768 (268)	8

2.4.1.1 UCI datasets with synthetic annotators

We used four datasets including Cleveland, Glass, Ionosphere and Pima, which were downloaded from UCI Machine Learning Repository[3]. Table 2.1 has the details about these datasets. The datasets were preprocessed for performing binary classification. Although all these datasets had no multiple versions of labels from real labelers, they were frequently used by many previous multi-labeler learning methods, including the three methods in [83, 82, 107] that we compared in this study. The rational was to have ground truth labels and known labeler reliabilities to test against the algorithms.

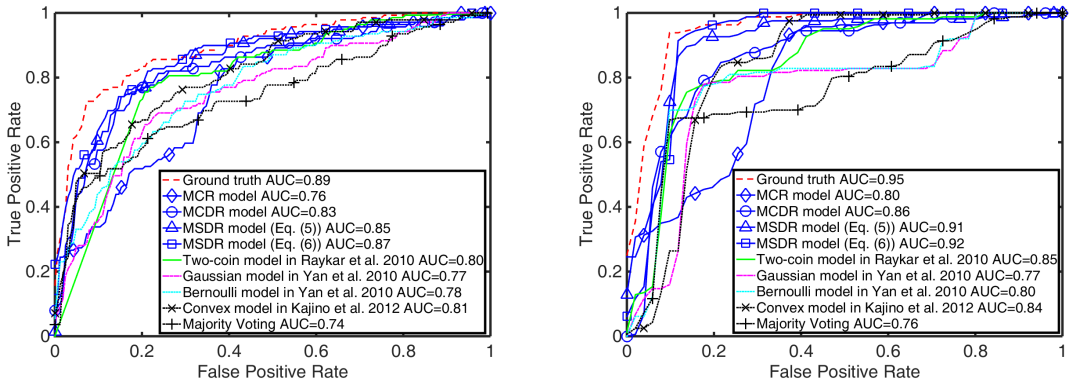
Since it was not straightforward to create synthesized labelers according to the pre-specified PPV's and NPV's, we created the labelers based on the pre-fixed sensitivities and specificities. The labelers were created following the same procedure used in [82]. We first specified two parameters for each labeler, the sensitivity α and specificity β. Five synthetic labelers were created for each of the four datasets. Their sensitivities and specificities were pre-defined as [0.6,0.6, 0.5, 0.7, 0.2] and [0.6, 0.6, 0.5, 0.2, 0.7], respectively. The third labeler's performance was close to a random guess. The first two labelers were given equal sensitivity and specificity, while the last two labelers were prejudicial in the sense that one of them had higher sensitivity and the other one had the exactly opposite parameter values.

[3]https://archive.ics.uci.edu/ml/datasets.html

Once the parameters were specified for a labeler, a random number was generated uniformly from $[0, 1]$ for each example. When the true label was $+1$ (or -1), if the random number was not bigger than the labeler's α (or β), this labeler chose the original label; or otherwise, (s)he flipped the sign of the label. After the labelers were created, their PPV's and NPV's were calculated.

In order to simulate the case where labelers have different levels of reliability on different examples, we randomly selected 50% of the data samples and ran k-means cluster analysis to group them into five subgroups. Then we made each of the five simulated labelers particularly accurate in annotating one of the subgroups, respectively, with no overlapping. Their labels coincided with the golden truth on the subgroup which they were assigned to. For the rest of data samples not belonging to the subgroup that was assigned to a labeler, the labeler will presume the same sensitivity and specificity levels used in the early experiments.

Figure 2.2 shows the Receiver Operating Characteristic (ROC) curves achieved by all the methods in comparison on the four datasets with five simulated labelers. The ROC was plotted by merging all the validation data from the 10 folds of the CV. From the ROC plots, we found that the MSDR model with Eq.(2.6) generally achieved superior performance over the other models by learning the varying expertise jointly with estimating the true labels. Among the other models, MCDR as an extension of the MCR model, which estimated labelers' weights based on the PPV and NPV, consistently achieved better performance than the MCR model that only used one parameter to capture the labeling accuracy. Compared to the method in [83, 82] which also built a two-coin model (with two parameters), the MCDR model could also achieve a slightly better performance in general.



(a) Cleveland (b) Glass

(c) Ionosphere (d) Pima

Figure 2.2: ROC curves on Cleveland, Glass, Ionosphere and Pima datasets with five simulated labelers (where two of them were simulated as good labelers, the third labeler was close to a random guess, the forth one was more accurate in labeling positive examples than negative ones and the last labeler was on the opposite of the forth one.).

2.4.1.2 Facial expression recognition dataset

The facial expression dataset was previously used to study crowdsourcing behavior [68]. The original dataset contained 585 head-shots of 20 users. For each user, images were collected in which the user could be looking at 4 directions: straight, left, right and up, and the user could present 4 different kinds of facial expression: neutral, happy, sad and angry. The images were labeled with respect to the 4 types of facial expression by totally 27 online labelers

at the Amazon Mechanical Turk. Because not all labelers labeled each image, on average, each image was labeled by 9 labelers. The previous study reported a labeling accuracy of only 63.3% using majority voting among labelers. Hence, the task of building a good feature-based classifier is very challenging.

We selected 220 images with the users looking straight, left and right without wearing sunglasses and performed experiments to classify, based on the image features, if an image contained a happy face. Twenty-four labelers were involved in labeling the 220 images. True positive labels were associated with 55 of the images, and the rest were labeled by -1. We set the missing labels to 0, which would automatically be ignored by any of the comparison methods. We segmented the region of an image containing a human face into 6×6 blocks. Local Binary Pattern (LBP) features [71] were extracted from each block and we aligned all these features together (2088 of them) to represent an image. We applied principal component analysis to reduce the dimensions to 120 that explained ≥ 95 % of the total data variance.

The area under the ROC curve (AUC) of each classifier was reported and summarized into Table 2.2 (the first column), where we used the actual labels of each image collected from online workers. Due to the difficulty of the problem itself and the significant amount of missing labels, all methods achieved modest AUC values. Our models MCDR, MSDR (both Eqs.(2.5) and (2.6)) and the Bernoulli model were among the best methods with MSDR models performing slightly better. All multi-labeler methods outperformed the majority voting baseline.

To test how the compared methods perform as the number of annotators increased, we also created more synthetic labelers following the same procedure as mentioned in Section 2.4.1.1. We set 30% of the labelers to have sensitivities and specificities around [0.6, 0.6], 30% around [0.5, 0.5] (random guess), 20% around [0.8, 0.2] while the rest 20% with [0.2,

0.8]. The results were also reported in Table 2.2 (from the second to the last columns) which clearly show that the difference in performance was magnified. The two MSDR models improved the performance by 3% to 13% over the other multi-labeler learning methods in these experiments. We also ran an experiment with 1000 synthesized labelers (results not shown in Table 2.2). We observed that the two-coin model of [82] had extremely worse performance (AUC = 0.55) than other models (e.g., the best MCDR AUC = 0.73), which shows that this model may perform poorly with a large number of annotators. Besides, the convex model of [47] did not work well either since this model suffered a lot from the synthesized annotators with lower accuracies (AUC = 0.57).

Figure 2.3 shows the average run time for an iteration of each method versus the number of annotators. All methods required longer time as the number of annotators increased. The two proposed models MCR and MCDR were more scalable since their run time curves were flatter than others and time costs were lower. It is partially because increasing the number of annotators only affects the optimization of the sub-problem, i.e., solving Problem (2.1) and (3.2) for $\mathbf{r}^{(k)}$ and $\boldsymbol{\alpha}^{(k)}, \boldsymbol{\beta}^{(k)}$ when the classifier parameters are fixed. This sub-problem is a simple linear program and easily scalable with a large number of labelers. Given the two formulations of MSDR had similar run time, Figure 2.3 reports the run time for MSDR with Eq.(2.5) only. The MSDR model was timely consuming in comparison with other models that also built individual labelers' classifiers, which may require the development of a more efficient optimization algorithm and we leave it for future work.

2.4.2 Biomedical Image Analysis

To diagnose a complex disease, a diagnostic image is often interpreted by multiple radiologists to enhance the diagnostic accuracy. In this section, we describe how the proposed methods

Table 2.2: AUC comparision on the facial expression dataset when the number of labelers increases.

Methods	24†	40‡	60‡	80‡	100‡	200‡
MCR	0.66	0.58	0.58	0.57	0.59	0.66
MCDR	0.68	**0.68**	0.60	0.60	0.62	0.70
MSDR (Eq. (5))	0.70	**0.68**	**0.68**	**0.63**	0.67	**0.73**
MSDR (Eq. (6))	**0.71**	**0.68**	0.67	**0.63**	**0.70**	**0.73**
Two-coin model	0.68	0.64	0.59	0.59	0.64	0.63
Gaussian model	0.66	0.61	0.61	0.56	0.61	0.59
Bernoulli model	0.67	0.65	0.65	**0.63**	0.61	0.66
Convex model	0.66	0.63	0.61	0.60	0.61	0.61
Majority voting	0.62	0.60	0.56	0.58	0.57	0.59

† Real annotators

‡ Synthetic annotators

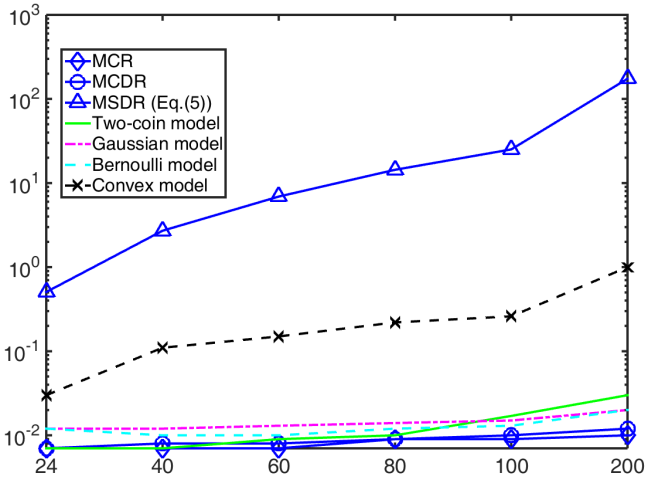


Figure 2.3: Average runtime per iteration for every method on facial expression datasets.

can help with cancer detection, heart abnormality detection and Alzheimer's disease analysis based on features that were extracted from a variety of imaging modalities, including mammographic images, ultrasound clips or MRI scans of brain. The mammographic images and

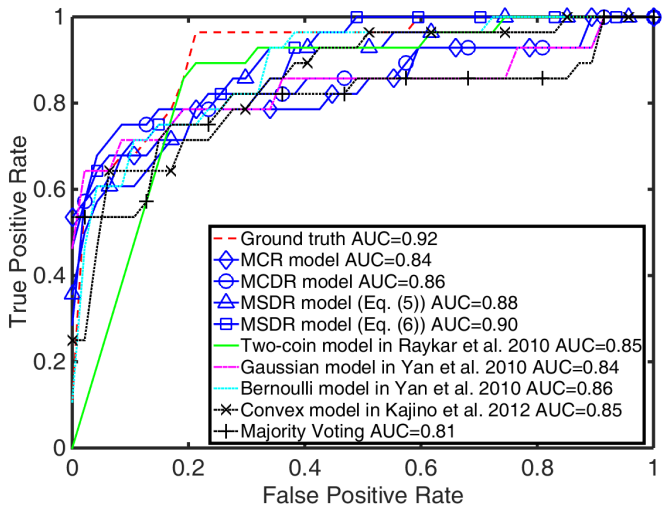
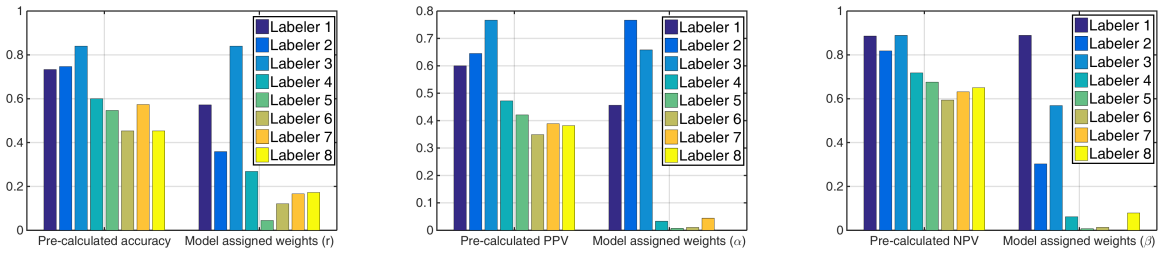


Figure 2.4: ROC comparison on the mammography dataset.

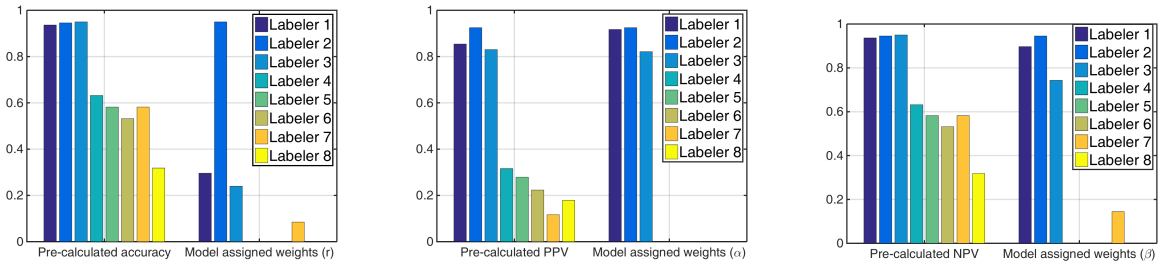
echocardiograms were annotated by multiple radiologists. The MRI dataset of Alzheimer's disease contained records for multiple visits and a doctor's annotation was supplied in each visit. The final reading of the images was also provided and served as the ground truth labels for a patient. We used the diagnoses in the different visits as multiple annotations or created synthetic labelers to provide multiple versions of annotations.

2.4.2.1 Detecting breast cancer in mammographic images

In this dataset, 75 mammograms were collected from real patients, of which the ground truth labels were obtained from biopsy which annotated whether the mammographic image contained a lesion. There were 28 positive samples (having a lesion) and 47 negative samples. Each sample image was represented by 8 attributes and was associated with the labels assigned by three radiologists. We created 5 more synthetic labelers by leveraging the ground truth labels. The labelers were synthesized with sensitivities [0.60, 0.50, 0.50, 0.20, 0.70] and specificities [0.60, 0.50, 0.50, 0.70, 0.20], which controled the accuracies of the labelers in



(a) Estimate $\mathbf{r}$ (Mammography data) (b) Estimate $\boldsymbol{\alpha}$ (Mammography data) (c) Estimate $\boldsymbol{\beta}$ (Mammography data)

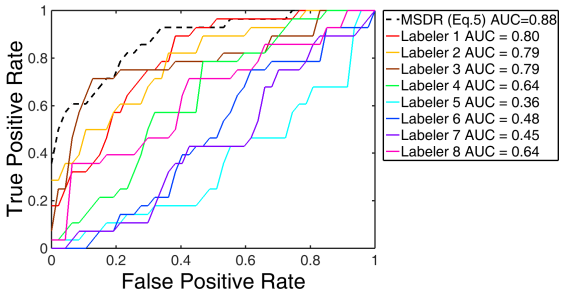


(d) Estimate $\mathbf{r}$ (HWMA data) (e) Estimate $\boldsymbol{\alpha}$ (HWMA data) (f) Estimate $\boldsymbol{\beta}$ (HWMA data)

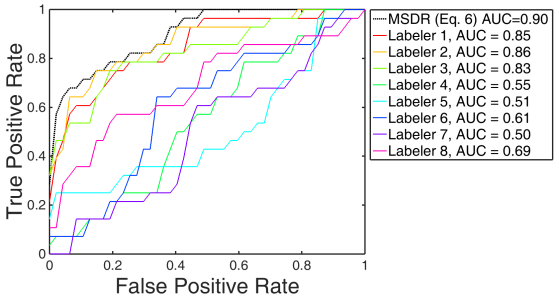
Figure 2.5: The figure shows the various parameters learned on Mammography and HWMA datasets. Sub-figures (a), (b), (c) and (g) are drawn for Mammography dataset, while (d), (e), (f) and (h) belong to HWMA dataset. Further, (a) and (d) show the estimated reliabilities by MCR against the true labeler accuracies; (b), (c) and (e), (f) show the estimated $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ by MCDR against the synthesized PPV and NPV; (g), (h), (i) and (j) show the ROC curves of the two final classifiers and each labeler's classifier obtained by MSDR.

terms of annotating either a positive or negative example.

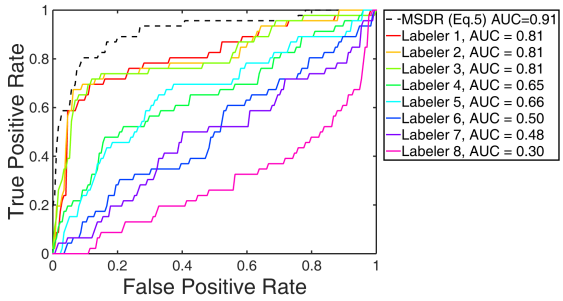
We drew the ROC curves of the classifiers constructed by the different methods in comparison together with the AUC statistic in Figure 2.4. According to the AUC values, MSDR model (Eq.(2.6) performed better than all the other multi-labeler learning algorithms. MCR achieved the lowest performance among the multi-labeler models but still outperformed the majority voting baseline. The two-coin model and MCDR performed similarly probably because both used two reliability parameters. Among the three models that used sample-specific reliabilities, our model was the best (beter than EMGaussian, EMBernoulli, and the early convex model).



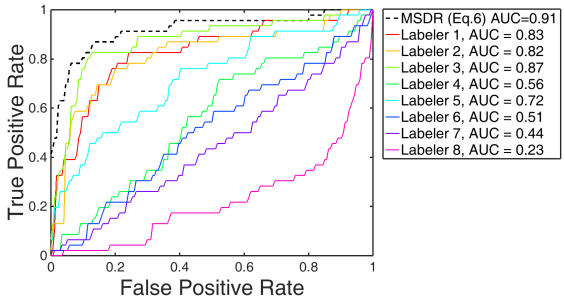
(a) $\{\mathbf{w}, b\}$ vs $\{\mathbf{w}_j, b_j\}$ (Mammographic images, MSDR (Eq. 5))



(b) $\{\mathbf{w}, b\}$ vs $\{\mathbf{w}_j, b_j\}$ (Mammographic images, MSDR (Eq. 6))



(c) $\{\mathbf{w}, b\}$ vs $\{\mathbf{w}_j, b_j\}$ (HWMA data, MSDR (Eq. 5))



(d) $\{\mathbf{w}, b\}$ vs $\{\mathbf{w}_j, b_j\}$ (HWMA data, MSDR (Eq. 6))

Figure 2.6: The figure shows the various parameters learned on Mammography and HWMA datasets. Sub-figures (a), (b), (c) and (g) are drawn for Mammography dataset, while (d), (e), (f) and (h) belong to HWMA dataset. Further, (a) and (d) show the estimated reliabilities by MCR against the true labeler accuracies; (b), (c) and (e), (f) show the estimated $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ by MCDR against the synthesized PPV and NPV; (g), (h), (i) and (j) show the ROC curves of the two final classifiers and each labeler's classifier obtained by MSDR.

Figure 2.5 (2.5a, 2.5b and 2.5c) shows the estimated reliability factors and compares them against the true labels or the simulated labeler performance. The first three labelers represent the radiologists. From Figures 2.5a, 2.5b and 2.5c, we observed that the MCR and MCDR models are able to sketch a general picture of the varying labeler expertise that is close to the true values/trend. For the MSDR model that builds a final classifier jointly with individual labelers' classifiers, the ROC plot in Figures 2.6a and 2.6b show performance for the classifiers constructed by the two MSDR models. The final classifier clearly outperformed

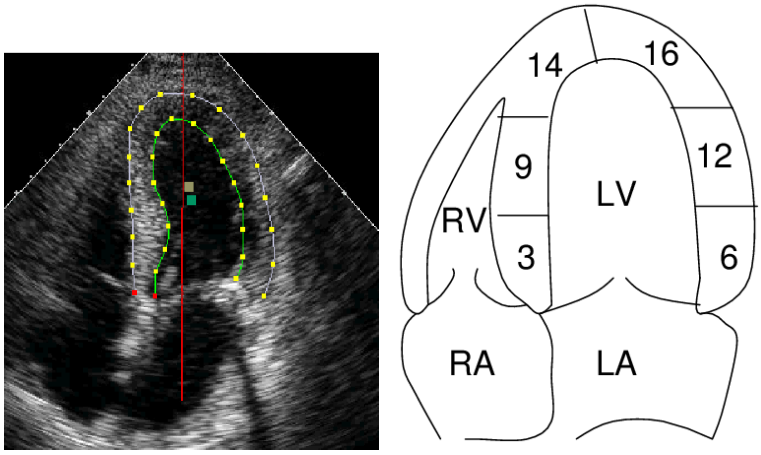


Figure 2.7: Left: an ultrasound image of Apical 4 Chamber (A4C) view; right: the 6 heart segments seen from the A4C view.

the classifiers built from any individual labeler's data. Our models created shrinkage effects that produced sparse $\mathbf{r}$ (or sparse $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$). As discussed early on, this shrinkage effect shows that true labels can be estimated from few reliable labelers for the tested datasets.

2.4.2.2 Heart wall motion analysis

The Heart Wall Motion Abnormality (HWMA) detection dataset contained the features extracted from the images of the wall motion of left ventricles in 222 heart cases. The wall of left ventricle is medically segmented into 16 segments. Figure 3.3 shows 6 of the 16 wall segments seen from the apical 4 chamber (A4C) view of an ultrasound clip. For each segment, 25 features were extracted. The feature extraction process was described in more detail in [76]. For each heart case and each segment, the ratings are provided by 5 doctors as the level of severity ranking from 1 to 5, besides, 0 would stand for the missing ratings. We assume that if the ratings are greater or equal to 2 then the label can be set as +1, which means there exists abnormality, otherwise the label is -1. Additionally, at the heart level,

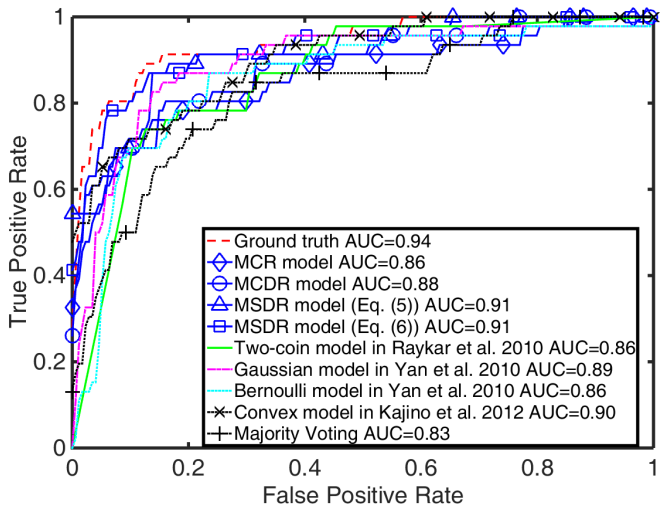


Figure 2.8: ROC comparison on HWMA dataset

if two or more segments of one heart have been claimed as abnormal, the heart-level label would be +1, which means the heart overall has abnormality, otherwise it is -1.

In this experiment, among all these 16 segments, the data extracted from segment 14 were more balanced than the other ones. We used this set of data to test our methods. Because only two cases from this dataset missed radiologists' ratings, there were total 220 examples. Since there was no ground truth available for the data, it is reasonable to make the majority voted labels from the 5 real doctors be the ground truth, and then we randomly selected three real doctors and created 5 synthetic labelers using the same settings for varying sensitivities and specificities as in Section 2.4.1.1. Figure 2.8 shows that the two proposed MSDR methods achieved the superior performance over the other methods.

Similarly, we also illustrated the reliability factors reported by the proposed models MCR and MCDR, which were included in Figures 2.5 (2.5d, 2.5e and 2.5f). The real radiologists were shown as the first three annotators. We observed that the proposed models excluded most of the synthetic annotators whose labels were not in good quality. The labels from

three radiologists were already sufficient to train the classifier well. Figures 2.6c and 2.6d show the classifiers trained by the MSDR model and each annotator's labels, where we can see that the three radiologists had similar labeling expertise and they were much better than synthetic labelers. The MSDR model combined the expertise of good labelers and thus achieved the best performance.

2.4.2.3 MRI-based Alzheimer's disease analysis

We tested the proposed models on the Alzheimer's Disease Neuroimaging Initiative (ADNI) dataset[4]. In the ADNI project, the collected data such as MRI and PET images of participants are used as the predictors to predict the progression of disease. The data we used contained 3063 MRI images taken from 882 participants including Alzheimer's disease (AD) patients, mild cognitive impairment (MCI) subjects and elderly controls. The participants were included in two ADNI study phases, ADNI GO and ADNI_2. Figure 2.9 shows an example of a participant's brain MRI image in the axial view. A participant had multiple MRI scans collected as he/she had several follow-up visits and the MRI scans were taken at each visit.

We used each MRI image as an example and constructed the classifier to predict the diagnosis of AD or MCI based on the features extracted from MRI images. Among all the 3063 MRI images, there were 833 normal cases (labeled by -1) whereas the remaining images are for AD or MCI patients (labeled by +1). Each MRI image was preprocessed by FreeSurfer[5] and represented by 307 features. The features can be categorized into 5 types: cortical thickness average, cortical thickness standard deviation, volume of cortical parcellation, volume of white matter parcellation and surface area.

[4]The ADNI website: http://adni.loni.usc.edu/

[5]http://adni.loni.usc.edu/methods/mri-analysis/

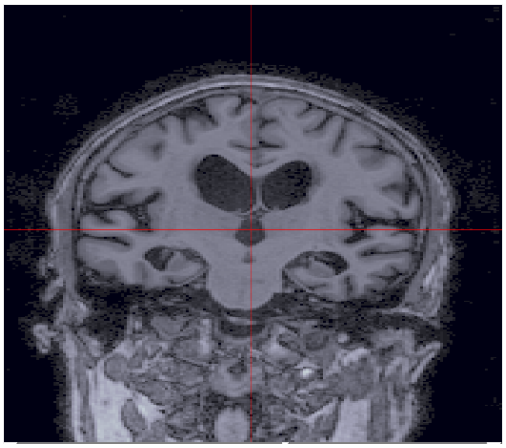


Figure 2.9: Example image of a MRI scan along axial view

In our first set of experiments, we extracted the data for 147 patients who completed (all) four visits at the month 3, 6, 12 and 24, respectively, and then we used the diagnoses for the first three visits as annotated labels so we had three different versions of the label. We used the diagnoses for the forth visit as the ground truth as it gave the latest stage of AD and MCI. Among all the compared methods, the classifier trained with the ground truth served the oracle model with the best performance of an AUC value of 0.64. The classifier trained with majority voted labels served as the baseline (AUC=0.58). The other methods achieved similar performance in general. The MCDR and MSDR models performed slightly better than others. (However, the difference became more significant when we increased the number of labelers as shown below). The three annotations at the month 3, 6, 12) served as good labelers with accuracies of [0.9, 0.93, 0.97] where the last labeler was the best. The relative labeling accuracy was reflected in the estimated reliabilities by the MCR model. For instance, $\mathbf{r}$=[0.2, 0, 0.8] indicated that the last labeler itself plays significant role in predicting the final diagnosis. We observed the similar labeler selection in the MCDR model given $\boldsymbol{\alpha}$=[0, 0.1, 0.9] and $\boldsymbol{\beta}$=[0, 0, 1].

Table 2.3: AUC comparision on the ADNI dataset when the number of synthetic labelers increases.

Methods	40	60	80	100	500	1000
MCR	0.63	0.68	0.71	0.74	0.73	0.78
MCDR	**0.66**	**0.72**	**0.74**	**0.76**	**0.76**	**0.81**
Two-coin model	**0.66**	0.70	0.70	0.72	0.71	0.70
Gaussian model	0.61	0.69	0.65	0.67	0.74	0.76
Bernoulli model	0.61	0.63	0.64	0.69	0.75	0.78
Convex model	0.65	0.66	0.65	0.65	0.68	0.70
Majority voting	0.61	0.63	0.63	0.60	0.65	0.67

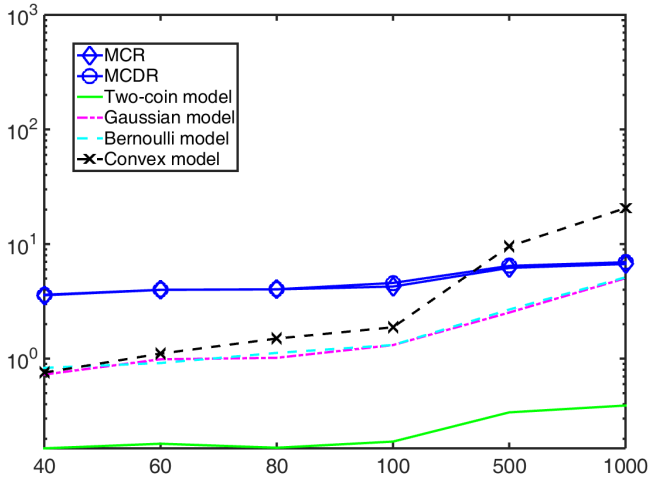


Figure 2.10: Average runtime per iteration for every method on ADNI dataset. The x-axis indicates the number of labelers used in the experiments.

In the second set of experiments, we created [40, 60, 80, 100, 500, 1000] synthetic labelers in the same way as the description in the experiments with the facial expression dataset. Because the MCR and MCDR models were more scalable to large datasets than the MSDR models, we further tested the MCR and MCDR models on the ADNI dataset using all 3063 images. The AUC values for the different methods were summarized into Table 2.3. The

results in the table show that the MCDR model had achieved a superior performance when we increased the number of labelers. We also recorded the averaged run time of one iteration for these methods. The comparison of the run time in Figure 2.10 shows that the two-coin model required the lowest run time across all the experiments with different numbers of labelers. When the number of labelers was relatively small, the MCR and MCDR had larger time costs than the other models. However, the two proposed models were more scalable to the number of labelers as we can see the run time curves were more flat. The convex model of [47] became more time consuming than the MCR and MCDR models when the number of labeler increased to 500 and 1000.

2.5 Conclusion

We have studied the multi-labeler learning problem that constructs classifiers from crowdsourcing labels and proposed three novel and unique formulations that all form bi-convex programs. By approximating the true labels with a weighted consensus of all labelers' opinions with the labeler reliabilities as the weights, we are able to modify the hinge loss function to become bi-affine with respect to the classifier parameters and the reliability factors of labelers. We employed three very general assumptions on the labeler reliability, including constant, class-dependent, and example-specific labeler reliability. The bi-convex programs can be effectively optimized by the widely-used alternating optimization algorithm, and outperform the state of the art in empirical tests.

Future extension of this work can examine the bi-convexity of the models more thoroughly, and explore some global optimization algorithms such as the one in [2] that can find a global minimizer for a bi-convex program although these algorithms are significantly more

complex. It is also worthy examining the varying reliability scores estimated by the third model, which may prove potential utility in real-world applications, for example, to group the crowdsourcing labelers according to their labeling reliabilities and behaviours. The datasets used in our experiments are relatively small. For many large datasets having inconsistent labels collected from crowdsourcing platforms such as AMT, they provide no input features but raw data examples, such as plain texts or images. Extracting meaningful features (input variables) from those datasets needs significant efforts, which goes beyond our goal of study in this work. Moreover, many of these datasets may provide no ground truth that can be used in model evaluation. Our future work will also include searching for larger datasets that are suitable for objectively and systematicly testing the proposed models.

Chapter 3

Learning Classifiers from Dual Annotation Ambiguity via A Min-Max Framework

In a variety of real-world problems, ambiguous and inconsistent annotations of data exist inevitably and bring an important set of machine learning problems associated with the efficient modeling and utilization of ambiguous supervision. Data annotation becomes ambiguous often due to both the labor-intensive and time-consuming nature in the labeling process and the difficulty of the annotation tasks themselves. The mechanism that causes labeling ambiguity varies from problem to problem, and multiple causes of ambiguity can exist in a single problem in many practical domains.

In document classification with respect to a focused topic, a document may contain multiple passages that either cover the corresponding topic or only relate to other topics. Consider a document as a bag comprising several passages as its instances. A document is often assigned to a topic category as long as one of its passages or instances is relevant

to the topic. When we try to classify a document, it would be very important to identify the specific passage in the document that corresponds to the given topic. An image can be represented as a bag of different regions and can be associated with the objects that each region dictates. This type of ambiguous annotation leads to the so-called multiple instance learning problem and usually has labeling bias in between positive and negative classes as positive labels are commonly based on evidence validation whereas negative labels indicate either true negative or lack of knowledge.

Many practical learning problems present multi-instance examples that are labeled by multiple annotators. For instances, a document can be labeled by many internet labelers in terms of whether it is relevant to a particular topic. Some labelers may recognize the passages in the document that correspond to the topic, whereas others may not, resulting in inconsistent annotations from these labelers. Moreover, negative labels, i.e., documents without a specific label, may be truly absent of a topic but can also indicate a failure of evidence search. When multiple labelers annotate if an image contains a specific object, they may perceive different regions of the image. Hence, some may give positive labels whereas others label it negative for the object, leading to a disagreement in the annotation. An example's true label becomes a latent variable, and multiple versions of its value are given. In this study, we solve the problem of constructing a classifier based on the different versions of a class label to predict if a multi-instance example is associated with the class label, and to identify the instances responsible for the class membership. Figure 3.1 illustrates this challenging problem.

To the best of our knowledge, existing multiple instance learning algorithms [63, 79, 80, 20] do not cope with the labeling inconsistency if multiple human experts have labeled the multi-instance examples. Our problem also differs from the multi-instance multi-label (MIML) learning problems [123, 45, 124], in which each bag as an example may correspond

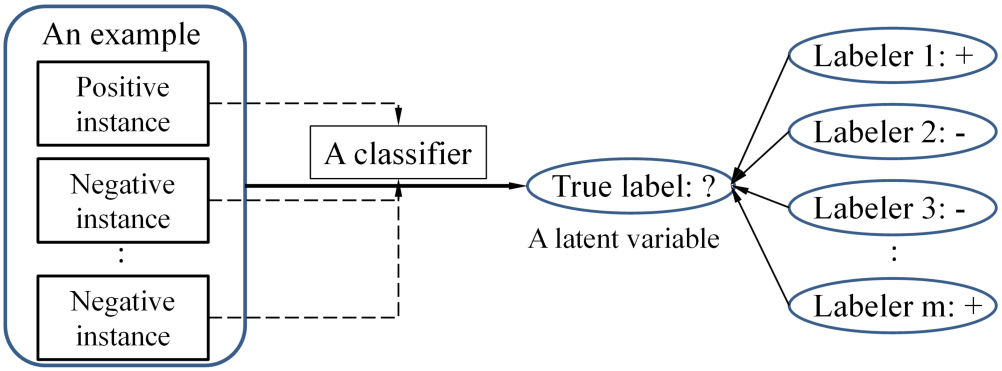


Figure 3.1: The problem of constructing a classifier from a training data set where each example is annotated by multiple labelers and the true label is unknown. This classifier is also expected to identify the instances of the example that are responsible for the class assignment (i.e., the positive instances).

to multiple labels when the example is labeled based on different concepts. These labels are all considered as accurate labels for the example. For instance, in image annotation tasks, a picture of landscapes may contain the sky, maintain or trees simultaneously, so this picture may correspond to all these labels. In contrast, for our problem, an example is labeled based on a single concept, corresponding to one class label, but multiple inconsistent versions of this class label are given rather than an accurate label (which we call the true label). Note that some versions may be *incorrect* labels. The state of the art in multi-labeler learning methods [118, 47, 82, 106, 83] can estimate a true label from the different versions of the label given by different labelers, but are not feasible to cope with examples of multiple instances, especially when different examples consist of different numbers of instances. In summary, none of the existing methods have addressed the dual ambiguity issue. Therefore, in this study we propose an approach to integrate expertise from multiple labelers and build classifiers that are able to classify bags of instances, or multi-instance examples with respect to the estimated true label and identify true positive instances for positive bags. The major contributions of this article are as follows.

- Propose a mechanism to learn the consensus label from multiple labelers by modifying the hinge loss which is commonly used in support vector machines [98].
- Extend the modified hinge loss to bags of multiple instances with a theoretical analysis to the resulted optimization problem.
- Two relaxation models are derived that properly approximate the original optimization formulation based on different assumptions on labeling bias of different labelers.
- Develop an alternating optimization algorithm to solve the two models which show superior performance in solving the dual annotation ambiguity problem.

3.1 A bi-convex program for learning classifiers from multiple annotators

In the problem of learning from multiple annotators, an input example $\mathbf{x}_i$ in training data is annotated with multiple versions $\{y_i^1, y_i^2, \cdots, y_i^m\}$ of the label y_i. Let $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$ comprise the n training examples, where $\mathbf{x}_i \in R^d$. We focus on the problem of binary classification where $y_i \in \{-1, 1\}$. The labels from different labelers $y_i^j \in \{-1, 1\}$, $j \in \{1, 2, ..., m\}$. We derive a new learning model by altering the hinge loss $\xi_i = [1 - y_i(\mathbf{x}_i^\top \mathbf{w} + b)]_+ = \max\{0, 1 - y_i(\mathbf{x}_i^\top \mathbf{w} + b)\}$ commonly used in SVMs where $\mathbf{w}$ is the weight vector and b is the offset of the linear model to be determined.

We approximate an example's golden standard y_i by a weighted combination of each labeler's labels. In other words, we estimate y_i by $\hat{y}_i = \sum_{j=1}^m r_j y_i^j$ and each labeler j is associated with a reliability factor r_j where $0 \leq r_j \leq 1$. If the reliability factors of all labelers are equal, this combination amounts to the majority voting. If we require additionally $\sum_j r_j = 1$, we approximate y_i by a convex combination of labelers' opinions. These different

ways of combinations may all be reasonable, and the most appropriate one may be problem-specific. If the weighted consensus of all labelers $\sum_j r_j y_i^j > 0$, the example i is more likely to be in the class of $y = 1$; or otherwise, it likely has a true label of $y = -1$.

We modify the hinge loss by replacing the true labels y_i, which are unknown during classifier training, by the weighted consensus. Thus,

$$\xi_i = [1 - (\sum_j r_j y_i^j)(\mathbf{x}_i^\top \mathbf{w} + b)]_+. \tag{3.1}$$

When the consistency is high among the labels given by different labelers, especially by reliable labelers, the magnitude of $\sum_j r_j y_i^j$ tends to be large regardless of its sign, showing high annotation confidence for $\mathbf{x}_i$. Minimizing the modified hinge loss Eq.(3.1) implies to penalize strongly the errors made on the examples $\mathbf{x}_i$ with highly-agreed labels. When the labeling consistency is low among reliable labelers for some examples, assigning these examples to either class can be a vague guess. The modified hinge loss, as how it is defined, will give small errors for these examples, and hence the classification performance on these ambiguous examples is not emphasized.

To regularize the empirical hinge loss, we minimize an objective function defined as $\lambda||\mathbf{w}||^2 + \sum_i [1 - (\sum_j r_j y_i^j)(\mathbf{x}_i^\top \mathbf{w} + b)]_+$ subject to the bound constraints on $0 \leq \mathbf{r} \leq 1$ where λ is a tuning parameter to balance between empirical errors and the regularization term $||\mathbf{w}||^2$. It is easy to verify that the objective function is bi-convex (i.e., convex with respect to $(\mathbf{w}, b)$ for fixed r and convex with respect to r for fixed $(\mathbf{w}, b)$) and the bound constraints give a convex feasible region. This problem forms a special case of bi-convex optimization. Even when we include the additional constraint $\sum_j r_j = 1$ for convex combinations of labelers' opinions. This constraint is affine and hence bi-affine. The resulting problem is still bi-convex. To form a canonical form of the optimization problem, the hinge loss is translated

into a constraint $(\sum_j r_j y_i^j)(\mathbf{x}_i^\top \mathbf{w} + b) \geq 1 - \xi_i$ for each example i where $\xi_i \geq 0$, and both r and $(\mathbf{w}, b)$ are now variables to be determined in the optimization problem. Overall, we search for the best $\mathbf{w}, b, \mathbf{r}$ by optimizing the following problem

$$\begin{aligned} & \min_{\mathbf{w},b,\boldsymbol{\xi},\mathbf{r}} \lambda||\mathbf{w}||^2 + \sum_i \xi_i \\ \text{s.t.} \quad & (\sum_j r_j y_i^j)(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i, \\ & \xi_i \geq 0, \quad 0 \leq r_j \leq 1, \\ & i = 1, 2, ..., n, \quad j = 1, 2, ..., m \end{aligned} \tag{3.2}$$

where we simply use the bound constraints on $\mathbf{r}$ (other constraints can be used if appropriate). Problem (3.2) is also a quadratically constrained quadratic optimization problem but with one of its constraints bi-convex. Due to the bi-convexity, efficient algorithms can be derived to approximate an optimal solution. We will discuss an algorithm based on alternating optimization in Section 3.3.

3.2 A min-max program for learning with dual annotation ambiguity

We now derive a learning formulation to address the dual labeling ambiguity issue where multiple experts or non-experts are utilized to annotate training examples, each of which consists of a varying number of instances. We extend the modified hinge loss Eq.(3.1) from the instance level to assessing the loss occurred on a bag. In the binary classification MIL, a bag is labeled positive if at least one instance in it is positive, and negative if all the instances in it are negative. The goal of a MIL problem is to distinguish positive bags

from negative bags. It is also important to infer the labels for the instances. In the dual annotation ambiguity problem, a bag B_k is labeled with m versions of the bag-level label y_k^j, $j = 1, \cdots, m$ and $k = 1, \cdots, n$. If we associate with each labeler a reliability factor r_j, the true label of B_k is estimated by $\hat{y}_k = \sum_j r_j y_k^j$. If the combined consensus of all labelers' opinions $\hat{y}_k > 0$ for a bag B_k, B_k is considered a positive bag; or otherwise, B_k is a negative bag.

We propose a min-max framework that aims to infer the labels of instances from the estimated bag-level true labels by generalizing the notion of loss functions to bags of multiple instances and minimizing the loss on bags directly. Let B contain the indices of the instances in a bag. Due to the asymmetric logic in the MIL labeling process, if a bag B is "negative", then $y_i = -1$, $\forall i \in B$, which corresponds to an "AND" logic among all of the instances in the bag. If a bag B is "positive", then $\exists i \in B$, such that $y_i = +1$, which corresponds to an "OR" logic among instances in the bag.

Now, let us pre-label all instances in a bag with the bag's estimated true label, or the consensus bag-level label of all labelers. Let ξ_{ik} be the hinge loss of the i-th instance of the k-th bag defined as $\xi_{ik} = [1 - (\sum_j r_j y_k^j)(\mathbf{x}_{ik}^\top \mathbf{w} + b)]_+$. If $\xi_{ik} = 0$, the i-th instance is correctly classified with respect to $\hat{y}_k$. If $\xi_{ik} > 0$, the i-th instance is mis-classified or classified without a proper margin. For a *negative* bag, the AND operation requires all instances in the bag to be correctly classified, which requires all hinge errors to be 0. In other words, $\max_{i \in B} \xi_{ik} = 0$. For a *positive* bag, the OR operation only requires one ξ to be 0, which amounts to $\min_{i \in B} \xi_{ik} = 0$. The min or max function conditioned on a bag's label can serve as an objective to be minimized for determining instance-level labels.

We thus construct a classifier by minimizing the integrated and regularized loss function

for the best parameters $(\mathbf{w}, b, \mathbf{r})$, i.e.,

$$\min_{\mathbf{w},b,\mathbf{r},\boldsymbol{\xi}} \lambda||\mathbf{w}||^2 + \sum_{k\in\{k:\ \hat{y}_k>0\}} \min_{i\in B_k}\{\xi_{ik}\} + \sum_{k\in\{k:\ \hat{y}_k\leq 0\}} \max_{i\in B_k}\{\xi_{ik}\} \tag{3.3}$$

where $\hat{y}_k = \sum r_j y_k^j$ is the bag-level label estimated from different labelers' labels for a bag B_k. This formulation classifies bags by calculating bag-level losses, but ultimately, it infers the labels of instances in a positive bag B_k as: $y_p = +1$, $\forall p \in \{p \in B_k | \xi_{pk} = \min_{i\in B_k}\{\xi_{ik}\}\}$, and otherwise $y_p = -1$.

Problem (3.3) is, however, mathematically intractable since (a) the index sets involved in the two summation terms rely on the estimated bag labels $\hat{y}_k$, or more precisely, the labeler reliabilities $\mathbf{r}$ that themselves are to be determined; and (b) to evaluate the objective function, it requires the evaluation of the minimum and maximum operations in the two summation terms.

We first prove that the evaluation of the minimum and maximum values in Problem (3.3) can be completely omitted once estimated true labels are given (or in other words, once the two index sets are determined). Then, we develop relaxed forms of Problem (3.3) that are tractable formulations and can effectively determine the index sets used in the two summations. Note that when the reliability of a labeler is known and fixed, the estimate of the true labels, $\hat{y}_k = \sum_j r_j y_k^j$ is determined which can be used to distinguish positive bags from negative bags. Then different treatments (min or max) will be used to compute their losses. We prove an equivalence between Problem (3.3) and the following problem when $\mathbf{r}$ is

fixed.

$$\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\mu}} \lambda||\mathbf{w}||^2 + \sum_{k\in\{k:\ \hat{y}_k>0\}} \sum_{i\in B_k} \mu_{ik}\xi_{ik} + \sum_{k\in\{k:\ \hat{y}_k\leq 0\}} \eta_k \tag{3.4}$$

$$\begin{aligned} \text{s.t.} \qquad & \hat{y}_k(\mathbf{w}^\top \mathbf{x}_{ik} + b) \geq 1 - \xi_{ik}, \quad \xi_{ik} \geq 0 \\ & \mu_{ik} \geq 0, \quad \sum_{i\in B_k} \mu_{ik} = 1, \quad \text{if } \hat{y}_k > 0, \\ & \eta_k \geq \xi_{ik}, \quad i \in B_k, \qquad \text{if } \hat{y}_k \leq 0, \\ & i \in B_k, \quad k = 1, 2, ..., n. \end{aligned}$$

where n is the total number of bags in the training set. The proof of this equivalence highlights the equivalence between the logic OR, i.e., $\min_{i\in B_k} \xi_{ik}$, and the convex combination of hinge losses ξ_{ik} in the bag B_k.

Theorem 3.1. *Any optimal solution* $(\hat{\mathbf{w}}, \hat{b})$ *of Problem (3.3) is optimal to Problem (3.4) and vice versa when* $\mathbf{r}$ *is fixed.*

Proof. We first prove that an optimal solution of Problem (3.4) has nonzero μ's only on the instances for which the classifier $\mathbf{w}^\top \mathbf{x} + b$ achieves $\min\{\xi_{ik},\ i \in B_k\},\ \forall\ k \in \{k : \hat{y}_k > 0\}$.

Let $(\hat{\mathbf{w}}, \hat{b}, \hat{\boldsymbol{\xi}}, \hat{\boldsymbol{\mu}})$ be the optimal solution of Problem (3.4). For notational convenience, denote the objective function of Problem (3.4) as $\mathcal{J}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\mu})$. Then let $\hat{\mathcal{J}}$ be the objective value attained at the optimal solution. Notice that the hinge loss $\hat{\boldsymbol{\xi}}$ is uniquely determined by $(\hat{\mathbf{w}}, \hat{b})$ as $\hat{\xi}_{ik} = \max\{0, 1 - \hat{y}_k(\mathbf{w}^\top \mathbf{x}_{ik} + b)\}$ for each instance $\mathbf{x}_{ik} \in B_k$.

If $\exists\ k_0 \in \{k : \hat{y}_k > 0\}$, and $\exists i_0 \in B_{k_0}$, such that $\hat{\mu}_{i_0k_0} > 0$ but $\hat{\xi}_{i_0k_0} \neq \min\{\xi_{ik_0}, i \in B_{k_0}\}$. Then let $\hat{\xi}_{pk_0} = \min\{\xi_{ik_0}, i \in B_{k_0}\} < \hat{\xi}_{i_0k_0}$. Then, re-set $\hat{\mu}_{pk_0} = 1$ and $\hat{\mu}_{i_0k_0} = 0$. Now, $\tilde{\mathcal{J}} = \hat{\mathcal{J}} - \hat{\mu}_{i_0k_0}\hat{\xi}_{i_0k_0} + \hat{\mu}_{pk_0}\hat{\xi}_{pk_0} < \hat{\mathcal{J}}$. This contradicts to the optimality of $(\hat{\mathbf{w}}, \hat{b}, \hat{\boldsymbol{\xi}}, \hat{\boldsymbol{\mu}})$.

By this contradiction, $\forall i, k$, if $\mu_{ik} > 0$, the corresponding ξ_{ik} has to be the minimum loss

that the classifier achieves on the k-th bag. This implies that at the optimality, the objective of Problem (3.4) is exactly equal to

$$\mathcal{J} = \lambda||\mathbf{w}||^2 + \sum_{k\in\{k:\ \hat{y}_k>0\}} \min_{i\in B_k}\{\xi_{ik}\} + \sum_{k\in\{k:\ \hat{y}_k\leq 0\}} \max_{i\in B_k}\{\xi_{ik}\}.$$

To prove the other direction, let $(\hat{\mathbf{w}}, \hat{b}, \hat{\boldsymbol{\xi}})$ be the optimal solution of Problem (3.3). We can simply define $\mu_{ik} = 1$, if ξ_{ik} achieves the smallest hinge loss over the bag B_k, or otherwise $\mu_{ik} = 0$, for all bags where $\hat{y}_k > 0$. Following the same line of thought, we can prove that $\boldsymbol{\mu}$ is optimal to Problem (3.4), and the solution $(\hat{\mathbf{w}}, \hat{b}, \hat{\boldsymbol{\xi}}, \hat{\boldsymbol{\mu}})$ is optimal to Problem (3.4). □

Problem (3.4) is also a bi-convex quadratic program where the objective function is bi-convex in the sense that it is convex with respect to $(\mathbf{w}, b)$ for fixed $\boldsymbol{\mu}$ and is convex with respect to $\boldsymbol{\mu}$ for fixed $(\mathbf{w}, b)$. All constraints of Problem (3.4) are affine and hence bi-affine.

3.3 Solving the proposed formulation

If we design an iterative algorithm to optimize the proposed formulation (3.3), the most significant challenge is how to tackle the stochastic nature of the objective function. The objective function of Problem (3.3) is stochastic (not deterministic) because the min and max functions are calculated on different sets of bags in different iterations if $\mathbf{r}$ varies in the iterations. The difficulty hence lies in the determination of r's because varying their values would alter the decision of a bag's label, and correspondingly alter the objective function. We hence develop relaxed forms of Problem (3.3) that are tractable, and approximate but effective solutions can be efficiently obtained.

In an alternating optimization process, we solve Problem (3.3) by alternating between

solving two sub-problems: one sub-problem is optimized for the best classifier characterized by $(\mathbf{w}, b)$ with a fixed choice of reliabilities $\mathbf{r}$; the other sub-problem is optimized for the best $\mathbf{r}$ after obtaining a classifier. The slack variables $\boldsymbol{\xi}$ that measure the hinge losses will need to be optimized in both sub-problems because they vary when either $(\mathbf{w}, b)$ or $\mathbf{r}$ is changed.

Sub-problem 1: building a MIL classifier when labeler reliabilities are known

If the reliability r of a labeler is known and fixed, Problem (3.4) is optimized for the best classifier $(\mathbf{w}, b)$. The parameters $\boldsymbol{\mu}$ are also correspondingly optimized in order to calculate proper bag-level hinge losses. An alternating optimization procedure can be developed to solve this problem (3.4) that alternates between solving two smaller sub-problems: one is to fix $\boldsymbol{\mu}$ in Problem (3.4) for the best $(\mathbf{w}, b)$ and the other is to fix $(\mathbf{w}, b)$ in Problem (3.4) for the best $\boldsymbol{\mu}$. The first sub-problem is a convex quadratic program similar to the standard SVM, and can hence be solved efficiently. The second sub-problem has an analytical solution and the optimal $\boldsymbol{\mu}$ can be directly obtained by searching for the smallest ξ_{ik} for each bag B_k with $\hat{y}_k > 0$ and setting the corresponding $\mu_{ik} = 1$ and other μ's to 0.

Sub-problem 2: determining a labeler's reliability when the instance-level predictions are known

If we fix the classifier parameters $(\mathbf{w}, b)$, the predicted value $\mathbf{w}^\top \mathbf{x} + b$ of every instance $\mathbf{x}$ is hence determined. The only variables in Problem (3.3) comprise the reliabilities of each labeler that ultimately determine which bag should use the min loss and which bag should

use the max loss. Converting Problem (3.3) into a canonical optimization formulation yields

$$\min_{\mathbf{r},\boldsymbol{\xi},\boldsymbol{\mu}} \sum_{k\in\{k:\ \hat{y}_k>0\}} \sum_{i\in B_k} \mu_{ik}\xi_{ik} + \sum_{k\in\{k:\ \hat{y}_k\leq 0\}} \eta_k \tag{3.5}$$

$$\text{s.t.} \quad (\sum_j r_j y_k^j)(\mathbf{w}^\top \mathbf{x}_{ik} + b) \geq 1 - \xi_{ik}, \quad \xi_{ik} \geq 0$$

$$\mu_{ik} \geq 0, \quad \sum_{i\in B_k} \mu_{ik} = 1, \quad \text{if } \hat{y}_k > 0,$$

$$\eta_k \geq \xi_{ik}, \quad i \in B_k, \qquad \text{if } \hat{y}_k \leq 0,$$

$$i \in B_k, \quad k = 1, 2, ..., n.$$

Problem (3.5) is still difficult to solve as the index sets on $\boldsymbol{\mu}$ and $\boldsymbol{\xi}$ depend on the values of $\mathbf{r}$. Two options exist to approximately solve Sub-problem 2. The first option, which we call the *all_min* model, is to estimate r's so all the bags' labels reflect at least the predicted value (by the current classifier) of one of its instances, which corresponds to applying an "OR" operation to each bag regardless the estimated bag labels.

The second option is to compute r's in such a way that we continue to apply the "OR" operation to those bags estimated to be positive (i.e., $\hat{y}_k = \sum_j r_j y_k^j > 0$) in Sub-problem 1, and apply the "AND" operation to those estimated to be negative. It implies that we will choose r's so that the new consensus bag labels can be tuned towards what the current classifier predicts. This option is a commonly used strategy in iterative algorithms, which means we fix $\hat{y}_k$ in Problem (3.5) by the one obtained in the previous iteration $\hat{y}_k^{old}$. We name this option the *selective min_max* model.

For the *all_min* option, only the single instance with the smallest hinge loss from each bag is used for optimizing r's. For the *selective min_max* option, based on the current estimated labels in Sub-problem 1, each positive bag yields one instance to be used in Sub-problem 2

whereas all instances in the negative bags are used in updating r's. This option employs the min and max losses according to the estimated bag labels used in Sub-problem 1. Hence, it will update the labelers' reliabilities to best reflect the current estimate of bag labels. These two relaxation options of Problem (3.3) lead to mathematically tractable problems that can be solved efficiently, and their solutions reflect different assumptions on the labeling bias of the labelers.

Labeling bias is commonly observed in MIL tasks, the *all_min* option takes the effect that positive labels may be more accurate than negative labels. Positive labels are commonly due to the recognition of an evidence for a class label. If a labeler annotates a bag with $y_k = +1$, it is likely that this labeler has witnessed an evidence from the bag. In contrast, a negative label may be only due to an insufficient evidence search. The *all_min* model treats any bag as potentially a positive bag if one of its labelers gives $+1$ and one of its instances satisfies $\mathbf{w}^\top \mathbf{x}_{ik} + b > 0$. It amounts to requiring an instance-level "OR" logic over all bags so to minimize only the smallest hinge loss occurred on each bag regardless of their labels. This leads us to the following optimization problem:

$$\begin{aligned} \min_{\mathbf{r},\boldsymbol{\xi}} \quad & \textstyle\sum_{k=1}^{n} \min_{i \in B_k} \xi_{ik} \\ \text{s.t.} \quad & (\textstyle\sum_j r_j y_k^j)(\mathbf{w}^\top \mathbf{x}_{ik} + b) \geq 1 - \xi_{ik}, \quad \xi_{ik} \geq 0, \\ & i \in B_k, \quad k = 1, 2, ..., n. \end{aligned} \tag{3.6}$$

However, the *all_min* model is a strong relaxation to the original formulation where for a negative bag, all its instances need to agree with the bag label. The *selective min_max* model takes the effect to leverage the classifier outputs (i.e., $\mathbf{w}^\top \mathbf{x} + b$) of all instances in a currently-estimated negative bag. An instance's predicted label is determined as $+1$ if $\mathbf{w}^\top \mathbf{x} + b > 0$ or -1 otherwise. The *selective min_max* model enforces the consensus labels,

Algorithm 2 An alternating algorithm for dual ambiguity problems

Input: $\mathbf{X}$ with bag index sets, all y_k^j's and λ
Output: $\mathbf{w}$, b and $\mathbf{r}$
1. Initialize $\mathbf{r}$ = a constant (evenly assigned to labelers).
2. Determine the bag labels by the weighted consensus $\sum_j r_j y_k^j$ based on the current values of r's.
3. Compute μ by finding the smallest ξ for each positive bag and setting the corresponding $\mu = 1$ and other $\mu = 0$. (In the initial step, set μ to $1/|B_k|$ for each positive bag.)
4. Solve Problem (3.4) with fixed μ for the best $(\mathbf{w}, b)$.
5. Solve the *all_min* model - Problem (3.6) (or the *selective min_max* model - Problem (3.5) with $\hat{y}_k$ replaced by previously-obtained $\hat{y}_k^{old}$) with fixed $(\mathbf{w}, b)$ for $\mathbf{r}$.

Repeat steps 2-5 until $(\mathbf{w}, b)$ reaches a fixed point.

i.e., the estimate of groundtruth, to be as consistent as possible to the predicted labels of all instances in the negative bags returned by Sub-problem 1.

3.4 The proposed alternating algorithm

Notice that there are two smaller sub-problems involved in solving Sub-problem 1. We develop an alternating optimization strategy that solves the three sub-problems (two from Sub-problem 1) in turns until reaching a fixed point. The resultant algorithm will output a classifier that can be applied to each instance and at the same time assess the labelers' reliabilities which are used to estimate the true bag-level labels. The first Sub-problem solves for $(\mathbf{w}, b)$ with a fixed $\mathbf{r}$, and the second Sub-problem optimizes with respect to $\mathbf{r}$ when the classifier $(\mathbf{w}, b)$ is fixed. Algorithm 2 depicts the details of our algorithm which is implemented separately for the *all_min* (Problem (3.6)) and *selective min_max* (Problem (3.5)) models.

Since Problem (3.3) has a stochastic objective function which varies due to the random

effect between taking the min or max loss determined by the value of the random variables $\mathbf{r}$, convergence analysis of our algorithms is difficult. We will leave it for the future to study techniques in stochastic combinatorial optimization [88, 22] to estimate the probability of the algorithm convergence in a theoretical form. Empirically, the proposed algorithms can terminate at a fixed point for both of the two approximation models within 20 iterations in all our experiments. Although the proposed Algorithm 1 solves two relaxed variants of Problem (3.3), it produces better classifiers than either regular MIL algorithms or multi-labeler learning algorithms by actively and effectively handling both the sources of ambiguity as shown in our experiments.

We briefly analyze the time complexity of Algorithm 1 by evaluating the computation cost at each iteration. Let ℓ, ℓ^+ and ℓ^- be the numbers of total instances, and the instances in the predicted positive and negative bags, respectively. Let n, n^+ and n^- be the numbers of all bags, and the positive and negative bags predicted at the current iteration, and d be the number of features for each instance. In Algorithm 1, three sub-problems are solved at each iteration. The first sub-problem finds $(\mathbf{w}, b)$ by solving Problem (3.4). Problem (3.4) is a convex quadratic program. It uses one instance in each positive bag and all instances in a negative bag to compute slack variables $\boldsymbol{\xi}$, so there are $n^+ + \ell^-$ slack variables. The problem dimension is $\tilde{d} = d + 1 + n^+ + \ell^-$. The second sub-problem finds $\boldsymbol{\mu}$ once $(\mathbf{w}, b)$ and $\mathbf{r}$ are fixed. This step has an analytical solution which only requires to scan through the instances in positive bags, so it requires a computation cost of $O(\ell^+)$. The last sub-problem is a linear program to optimize $\mathbf{r}$ and update slack variables $\boldsymbol{\xi}$, and the problem dimension is $\tilde{d} = m + n$ for the *all_min* model and $\tilde{d} = m + n^+ + \ell^-$ for the *selective min_max* model. We used the simplex method and a simplex-based active set method in the CPLEX optimization software [40] to respectively solve the linear and quadratic programs. The simplex method has the exponential worst-case complexity [53] but polynomial average-case

complexity [87, 93]. For instance, by assuming a spherically symmetric distribution on the constraint coefficents, a widely-used polynomial upper bound on the complexity of simplex was obtained as $\tilde{d}^{2.5}\tilde{n}^{\frac{1}{\tilde{d}-1}}$ where $\tilde{n}$ is the number of constraints in the program [17]. It is well known that the simplex method performs very efficiently in practice, which is the case shown in our empirical study. Given Algorithm 2 typically stops after 20 iterations, its time complexity would approximately be a constant times the order of complexity of simplex.

3.5 Evaluation

We implemented Algorithm 2 in Matlab where Problems (3.4), (3.5), and (3.6) were solved by calling CPLEX optimization solvers. We tested the proposed approach against the state of the art on several benchmark data sets from the natural language processing (NLP) domain, real-world crowdsourced data sets generated from human facial expression images and a medical problem that used echocardiograms for heart wall motion analysis (HWMA). First, we validated if an algorithm that deals with two sources of labeling ambiguity would improve multiple instance learning by better integrating experts' varying expertise. Second, we validated if our algorithm that enables multi-labeler learning methods to deal with examples represented by sets of instances will improve the study of some real-life crowdsourced data. Third, we investigated the algorithmic behavior and scalability of the proposed approach with respect to large quantities of labelers, which simulated the effects of large-scale crowdsourcing.

Since existing MIL methods are unable to deal with more than one version of class labels assigned to an example, following a common practice, we used majority voted labels in these methods to train classifiers. Existing MLL methods cannot easily tackle the situation that

different examples have different numbers of instances. We preprocessed the original data so that examples were represented using vectors of the same length. In our experiments, this was achieved by appropriately merging features from the different instances, so all methods were compared on the basis of the same amount of data/information for fair comparison.

3.5.1 Evaluation data sets

The first set of evaluation data was collected for document categorization, and was widely used for evaluating MIL methods [4]. The second set of data contained facial expression images that were annotated by multiple online labelers. The third data set contained HWMA features that were extracted from ultrasound videos, and was used to diagnose if a human heart had abnormal motion on its left ventricular wall by multiple radiologists. All data sets were used to compare the proposed approach against representative MIL methods. The facial expression image data set and the HWMA data set, both with real crowdsourced labels, were used to validate the proposed approach against existing MLL methods.

Table 3.1: Statistics of NLP data sets

Data sets	Bags	Positive Bags	Features	AVG. Ins/Bag
Trec1	400	200	46	8
Trec2	400	200	48	8
Trec3	400	200	31	8
Trec4	400	200	48	8
alt.altheism	100	50	200	54
comp.graphics	100	50	200	31
sci.med	100	50	200	30
GOcomponent	718	359	200	18
GOfunction	770	385	200	17

* AVG. Ins/Bag : rounded average instance number per bag

3.5.1.1 NLP benchmark data sets

Three sets of NLP data were used with their summary shown in Table 3.1.

TREC data sets [4]. Four TREC data sets were downloaded from the website of National Institute of Standards and Technology, http://trec.nist.gov/. They were collected from several years of selected MEDLINE articles. Each article was split into multiple passages using overlapping windows of maximal 50 words in each window. Since the TREC data was extremely sparse, we performed a principal component analysis to reduce the data dimension. We chose the number of principal components that cumulatively explained 75% of the total data variance in each data set, which produced 46, 48, 31 and 48 features for the four TREC data sets, respectively. All the four data sets had 400 bags including 200 positive bags. The four data sets contained 3224, 3344, 3246 and 3391 total instances, respectively.

Newsgroups data sets [86] were composed from 20 Newsgroups corpus. In this data set, each news post corresponded to an instance. For each of the 20 news categories, each bag was made up by a random number of posts. For positive bags, 3 % of the posts were randomly drawn from the target category and the remaining posts were from other categories. Three categories of these data sets, alt.atheism, comp.graphics and sci.med, were used in our experiments. Each of them contained 100 bags including 50 positive bags.

BioCreative data sets [79, 14] were derived from the articles published in biomedical journals based on the names of human proteins, and their relatedness to the gene ontology (GO) codes. The gene ontology consists of 3 hierarchical domains of standardized biological terms referring to cellular components, biological processes and molecular functions, and each term was mapped to a unique GO code. A <protein, article> pair was labeled with a GO code if the article contained text that linked the protein to the GO code. Examples labeled positive for a GO code consisted of documents that were labeled with that GO

code. We used two specific data sets, referred to as GOcomponent and GOfunction in our experiments.

3.5.1.2 Facial expression data sets with real crowdsourced labels

We also tested our methods on the Facial Expression data set previously used in a crowdsourcing study [68]. This data set contained 585 head-shots of 20 users. For each user, images were collected in which the user could be looking at 4 directions: straight, left, right and up, and the user could present four different kinds of facial expression in each direction: neutral, happy, sad and angry. The images were labeled with respect to the 4 types of facial expression by totally 27 online labelers at the Amazon Mechanical Turk. On average, each image received labels from 9 labelers. If a labeler did not annotate an image, we set the label to be 0, which corresponded to no evidence search for the corresponding facial expression from the specific labeler, and would not be used by any method.

We performed experiments to classify, based only on image features, if an image contained a happy face. We selected a set of 220 images with users looking straight ahead, left and right. We excluded images in which users wore sunglasses. Twenty-four labelers were involved in labeling the 220 images, of which 55 were associated with true labels ("+1") for the happy facial expression, and others were hence labeled by "−1". An early work in [68] estimated the actual expression labels using majority voting among the 9 labelers, and reported an accuracy of only 63.3% against the true labels for happy expression. Hence, this data set represents a very difficult problem. It can be even more challenging to not only estimate the true class labels but also simultaneously classify these images based on image features to the estimated true class.

Each image was represented by a collection of patches (or instances). Each patch was

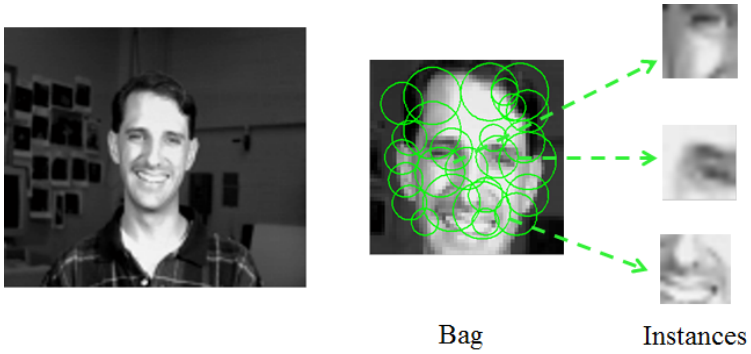


Figure 3.2: An exemplar facial expression image which is represented by a bag of multiple instances (patches) as shown in circles.

described by a vector of numerical image features. Since the original image contains large areas of background rather than the human face, we adopted the technique used in [20] to detect salient regions of an image. This has been proved to be a successful technique for MIL tasks [46, 28]. We first searched patches with a scale between 20 and 50 pixels. The largest detected region contained mostly the face area. Then, we detected salient regions only on the face areas with the scale varying from 2 to 8 pixels. This step gave us 8 to 32 salient regions (instances) for an image. Figure 3.2 presents a sample image of the detected salient regions on the human face. We resized each salient region into 40 $\times$ 40 pixels. The Local Binary Pattern (LBP) method [71] was used to extract features (58 of them) from each patch. The central location and scale of the detected salient region were also used as features. Totally, 61 features were computed for each patch or instance. In order to compare with algorithms that were only able to handle single instance examples, we divided the subregion containing mainly the human face into patches within a grid, and then LBP features were extracted for each patch. The LBP features from all patches were concatenated to form a single-instance example.

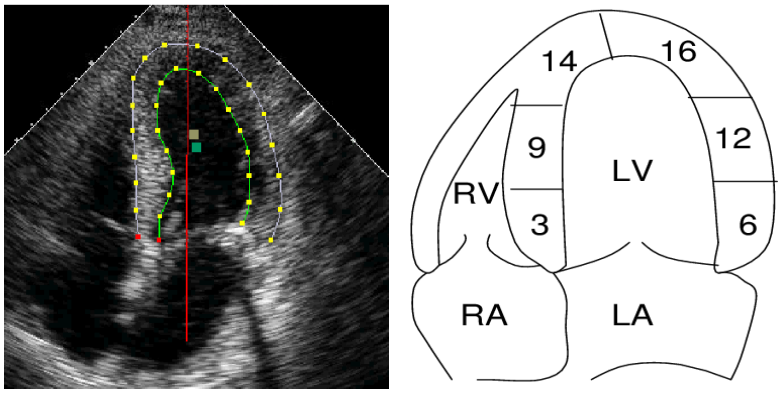


Figure 3.3: Left: an ultrasound image of Apical 4 Chamber (A4C) view; right: the 6 heart segments seen from the A4C view.

3.5.1.3 Medical image data sets with diagnoses from multiple radiologists

The goal of HWMA was to analyse and predict if a patient heart had abnormal motion based on image features extracted from two sets of ultrasound images: base-dose and peak-dose, collected in stress tests. The wall of left ventricle is medically segmented into 16 segments, corresponding to 16 instances. Figure 3.3 shows 6 of the 16 wall segments seen from the apical 4 chamber (A4C) view of an ultrasound clip. For each segment, 25 features were extracted. We also concatenated the features from each of the 16 segments to form a single-instance example. Base-dose image set contained 220 heart cases. The peak-dose set had 208 cases where 12 cases were dropped from this set due to poor image quality. The feature extraction process was described in more detail in our early works [76].

Five expert radiologists rated each segment of each heart case in terms of the severity of abnormality ranging from 1 to 5. If a rating was greater than 1, the segment was abnormal, and hence its label $y = +1$, and otherwise $y = -1$. If one of the segments was rated abnormal, the entire heart was rated abnormal. Groundtruth labels are usually difficult to acquire for HWMA. Researchers generally treat the consensus of expert readings as the

groundtruth. Hence, the majority voted segment-level labels were used in our experiments as groundtruth, based on which the bag-level groundtruth labels were induced.

3.5.2 Comparison to existing multi-instance learning algorithms

The NLP data sets were originally designed for testing MIL methods with groundtruth labels. In this dissertation, however, the goal of the study is to evaluate if dealing with dual annotation ambiguity yields better learning performance. In other words, we deal with the kind of problem where no groundtruth labels are available, and instead multiple versions of a label are given and associated with a bag of instances. The NLP datasets were chosen to use in our experiments because by means of their groundtruth labels we could objectively evaluate the accuracy of our models.

We hence simulated 20 labelers from the groundtruth labels of these NLP data sets. Each labeler's labels were created following the same procedure discussed in [82]. We first specified two parameters for each labeler, the sensitivity α and specificity β. Four of the labelers were specified to have both sensitivity and specificity close to 0.5. In other words, these labelers' performance was close to random guess. Six of the labelers were given equal sensitivity and specificity in the values of 0.6, 0.65, 0.7, 0.75, 0.8 and 0.85. The rest ten labelers were prejudicial in the sense that half of them had higher sensitivity than specificity, i.e., $[\alpha, \beta] = [0.8, 0.3]$, $[0.75, 0.3]$, $[0.75, 0.4]$, $[0.7, 0.4]$, and $[0.6, 0.4]$, and the other half had the exactly opposite parameter values. Once the parameters were specified for a labeler, a random number was generated uniformly from $[0, 1]$ for each example (a bag). When the true label was $+1$ (or -1), if the random number was not bigger than the labeler's α (or β), this labeler chose the original label; or otherwise, (s)he flipped the sign of the label.

Although HWMA data set received crowdsourced diagnoses from five radiologists, we

simulated 20 labelers in the same way as described above to increase noise level. Because the facial expression data set already came with 24 labelers' annotation, we did not simulate labelers for this data set. The experiment on facial expression data was done on three selected sets of the images where faces oriented in three different directions. Therefore 220 facial expression images were used and the performance was averaged over the three sets.

We compared our approach with four representative MIL methods listed below. These methods were implemented in PRTools [84] and its extension with MIL toolbox [94] except the MIL method, MIL-hinge, in [10] which was solved using CPLEX.

- mi-SVM [4] based on a mixed integer program, with a linear kernel
- MILBoost [100] based on AdaBoost, with 100 rounds as the maximum number of iterations
- MILES [20] based on a conversion to a single-instance example and the application of sparse SVMs
- The MIL-hinge model in [10] based on a revision to the hinge loss

These MIL methods cannot handle crowdsourced labels. To show that the ability of estimating true labels is important in the multi-instance learning setting, we used majority voted labels for the MIL methods (which is a common practice if an algorithm can only take one version of the labels), and let our methods automatically estimate the true labels. If the proposed methods perform better than the standard MIL methods, it demonstrates that the estimated labels by our models are more accurate than majority votes, and our models are better alternatives when dual ambiguity exists.

Because MIML learning is related to our learning problem, the following two representative MIML methods were also used in our comparison. (Their open-source codes were

obtained from the authors' website.) However, MIML learning solves a different problem that constructs multiple classifiers altogether by jointly considering related MIL classification problems. It is not designed for integrating crowdsourced labels to construct a single classifier for only one target label. In our experiments with the MIML methods, we treated each labeler's label as a target label in the multi-label setting. In other words, if we have 20 labelers, a MIML method will report 20 classifiers, each corresponding to a labeler (although these classifiers were jointly built). Hence, the performance of a MIML method was compared by reporting the accuracy of the best classifier among the 20 classifiers that it constructed.

- M^3MIML [112] based on a quadratic program to maximize the classification margin, with a linear kernel
- MIMLSVM [123] based on a revision to the SVM formulation, with a linear kernel

Five-fold cross validation (CV) was performed to test all of the methods. There was no tuning parameter in the MILBoost method except we set its maximum number of iterations to 100. Other methods, mi-SVM, MILES, M^3MIML, MIMLSVM, had a tuning parameter named C. The MIL-hinge method had a tuning parameter γ. The two proposed methods had a hyperparameter λ. All these parameters were tuned in the same procedure within the training data. An internal three-fold CV was performed within each training phase to select a proper hyperparameter value for each of the methods from the choices of $2^k, k = -10, -9, \cdots, 6$.

Table 3.2 shows the comparison on the averaged prediction accuracies for the bag-level labels and the standard deviation based on five separate trials of CV. The highest averaged accuracies were shown in bold fonts. As shown in Table 3.2, our methods obtained better classification accuracies than MIL methods on 10 out of the 12 experiments. On the rest

two data sets, comp.graphic and sci.med, our methods achieved comparable performance with the best results. In particular, the two MIML methods showed worse performance in general than standard MIL methods that used majority voted labels. The MIML methods employed all the 20 versions of labels and used them to learn classifiers jointly for each labeler. As majority of the labelers were simulated with low accuracies, even though we reported the best classifier's accuracy, the performance was contaminated by other labelers' performance. This result provided an evidence that a method for MIML learning would not be a solution for the dual ambiguity problem. All these results demonstrate the effectiveness of our methods and validate the hypothesis that better integration of annotators' expertise can improve multi-instance learning in a crowdsourcing scenario.

Table 3.2: Comparison on TEST accuracies (%) for predicting bag labels between our approach and MIL, MIML methods

Data sets	mi-SVM	MILBoost	MILES	MIL-hinge	M^3MIML	MIMLSVM	*all_min*	*selective min_max*
Trec1	82.0(1.0)	85.3(4.0)	87.8(1.3)	86.3(2.1)	80.0(4.2)	77.5(3.9)	92.0(2.6)	**92.5(1.3)**
Trec2	69.2(1.7)	73.0(1.8)	72.5(3.0)	66.0(1.4)	70.2(1.2)	67.5(2.2)	**73.0(1.5)**	72.0(1.5)
Trec3	70.7(0.8)	73.7(1.8)	71.8(2.1)	65.3(3.3)	70.0(1.4)	65.5(4.2)	**75.5(2.6)**	74.0(1.9)
Trec4	75.5(1.8)	76.5(1.8)	68.0(2.2)	66.0(1.2)	77.5(3.5)	67.5(2.9)	**80.0(2.5)**	79.5(0.6)
alt.atheism	64.6(2.1)	62.8(2.2)	58.0(2.4)	63.0(1.6)	63.0(1.4)	62.0(2.7)	64.0(2.9)	**69.0(2.6)**
comp.graphic	54.0(2.2)	**58.0(3.9)**	54.0(2.6)	54.0(1.1)	57.0(0.7)	56.7(2.8)	56.0(2.1)	55.0(1.9)
sci.med	62.0(3.3)	65.0(3.7)	**68.0(3.2)**	62.0(2.1)	59.3(4.3)	59.0(4.6)	**68.0(3.4)**	67.0(3.7)
GOcomponent	74.5(3.0)	78.0(1.5)	78.0(3.3)	72.1(1.1)	71.6(4.1)	70.7(2.1)	79.4(2.6)	**80.0(2.8)**
GOfunction	80.7(1.7)	77.7(2.5)	77.9(2.2)	74.8(0.6)	69.1(2.5)	65.8(2.1)	**80.9(2.2)**	76.6(2.9)
HWMA(base)	69.5(1.3)	70.9(1.5)	69.5(2.6)	70.1(2.6)	70.0(2.4)	69.1(3.4)	**74.6(0.9)**	72.7(1.5)
HWMA(peak)	72.1(1.1)	72.6(2.0)	73.4(1.9)	74.2(1.0)	68.3(4.4)	73.6(2.4)	**83.6(1.2)**	77.4(2.2)
FacialExpression	56.3(1.3)	57.0(1.3)	60.4(2.2)	54.2(1.1)	54.0(4.0)	53.8(2.2)	**61.1(2.6)**	58.6(2.1)

Although our approach handles both multi-instance examples and crowdsourced labels, it is worth investigating how the proposed MIL component works by itself, which also sheds light on what causes the performance improvement in Table 3.2. We performed additional experiments to compare the performance of the four MIL methods with that of our methods when groundtruth labels were provided. Note that when only one version of the labels, i.e. the groundtruth labels, are provided to our methods, the second Sub-problem in Algorithm 1 will be omitted because the only reliability parameter r will be set to 1 automatically. The two models, *all_min* and *selective min_max*, will be identical. In this situation, our approach is treated merely as another MIL approach. In this set of experiments, we observed that our method performed the most similarly to MIL-hinge with an average test classification accuracy 76.2% and standard deviation 3.1% over the 12 data sets. The four MIL methods, mi-SVM, MILBoost, MILES, and MIL-hinge reported average classification accuracies of 75.3%(3.2%), 75.7%(2.4%), 76.6%(3.2%), and 76.2%(3.1%), respectively, over the 12 data sets. We see that the performance of our MIL component is comparable to other state-of-the-art MIL methods. This observation confirms that the performance improvement in Table 3.2 is due to the ability of our methods to handle dual ambiguity.

3.5.3 Comparison to existing multi-labeler learning algorithms

We compared our methods with three recently-published MLL algorithms as listed below. Note that many other learning-from-crowds algorithms do not aim to build classifiers rather than study the nature of the crowd behaviors. The three MLL methods we chose all construct classifiers by integrating expert expertise and are most suitable for comparison with our methods.

- Expectation-maximization (EM) method with a two-coin model as labeler accuracy

prior [82]

- EM method with Gaussian prior on labeler accuracy [107]
- EM method with Bernoulli prior on labeler accuracy [107]

In the multi-labeler learning context, the best possible classifier would be obtained by training a classifier against the true labels if they are known. A baseline model could be the classifier trained with respect to the simple majority votes across all labelers. Hence, we also built MIL classifiers [10] using the groundtruth labels and majority voted labels as the best possible model and a baseline model.

Table 3.3: Averaged AUCs over the 5 folds of cross validation on the facial expression data set

Algorithms	Averaged AUC	Standard deviation
Groundtruth	0.63	0.04
all_min	**0.61**	0.01
selective min_max	**0.59**	0.02
Two-coin model in [82]	0.56	0.02
Gaussian model in [107]	0.55	0.01
Bernoulli model in [107]	0.55	0.01
Majority voting	0.53	0.01

In the experiments with the facial expression data set, only our methods can run on examples with varying numbers of patches as detected. To make single-instance examples, we resized the face area of each image into 120×120 pixels and split into 6×6 grid cells. Then, each image had the same number of 36 instances. The same 15 LBP features were extracted from each patch. Hence, this process transformed each image into a vector of the same length (with 540 features). Five-fold CV was performed on this data set where our methods used the 36 instances in each bag and other methods used single instances of 540 features. The averaged performance was reported. Table 3.3 shows the averaged Areas-

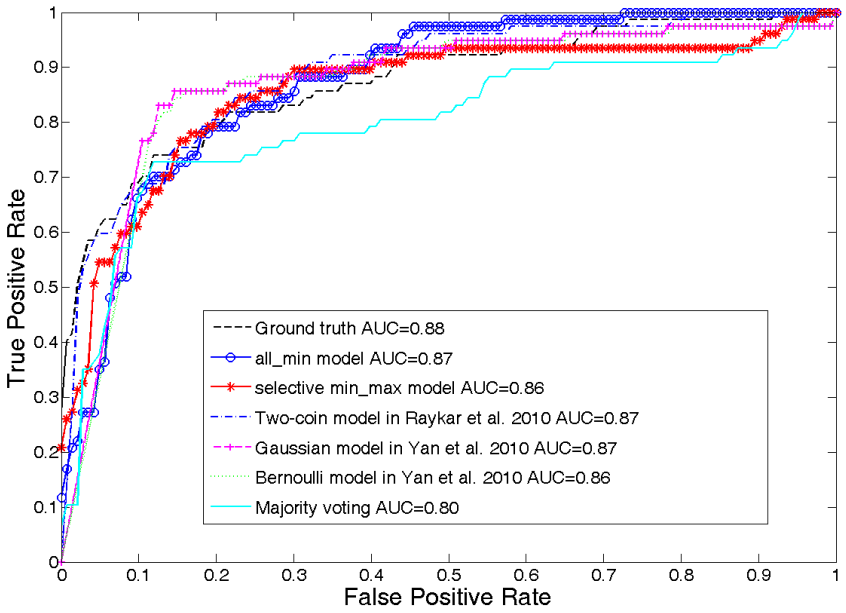


Figure 3.4: ROC comparison among different methods for the HWMA base-dose images.

Under-the-ROC-Curves (AUC) and the standard deviations over the five folds in the CV. Since the facial expression data set was extremely difficult, all methods had modest AUCs, but the proposed methods still outperformed other methods.

In the experiments with the HWMA data set, we combined the 16 segments for each heart case to form single-instance examples. Given each segment was represented by 25 image features, we obtained 400 features for each heart example. In order to more closely examine the reliability estimates of the different methods, we used 3 simulated labelers and 2 actual radiologists. The three simulated labelers had sensitivity of [0.6, 0.65, 0.7] and specificity of [0.4, 0.65, 0.7], respectively. The first simulated labeler was the least competent labeler. Based on the groundtruth labels, there were 77 and 71 positive bags, respectively, from the base-dose and peak-dose image sets. The HWMA data set was split by the five radiologists to form a test set for each dose. We draw receiver operating characteristic (ROC)

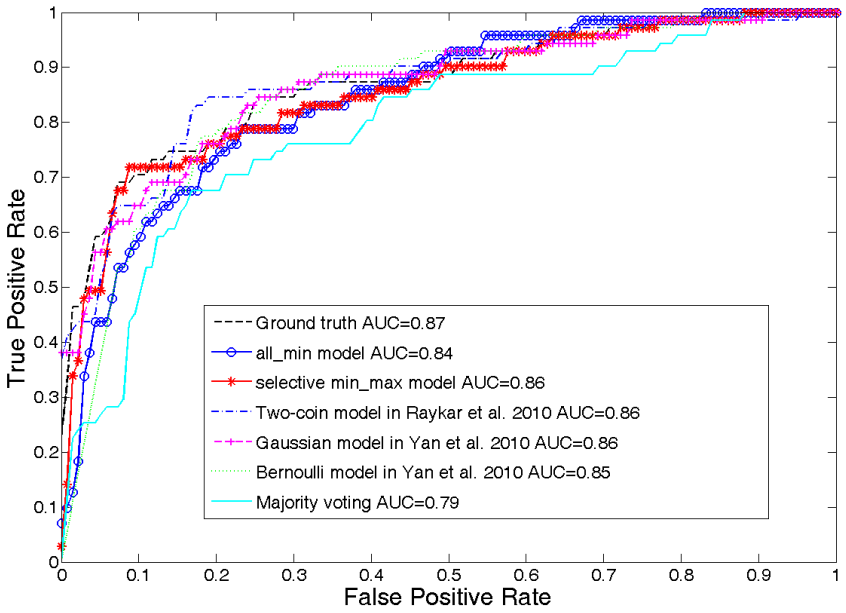


Figure 3.5: ROC comparison among different methods for the HWMA peak-dose images.

curves [27] to measure the test performance of each classifier.

Figure 3.4 and 3.5 show the ROC curves of the classifiers built by different methods on HWMA data sets. AUCs were also computed and given in the two figures. Notice that only the models trained with groundtruth, majority voted labels and two of our methods were obtained based on multi-instance examples. The other three methods ran on single-instance examples because they could not handle multi-instance examples. The empirical results on all the data sets show that the classifiers trained with groundtruth performed the best as expected. All classifiers obtained by multi-labeler learning methods performed better than the baseline model.

3.6 Conclusion

We have derived an effective approach to construct classifiers when multiple annotators with varying expertise are utilized to label bags of instances. In many practical applications, dual labeling ambiguity presents where one kind of ambiguity comes from the inconsistency of multiple labelers' labels and the other comes from the inability of labeling individual instances of an example. We first modify the hinge loss to employ the weighted consensus of different labelers' labels and then use min-max optimization to extend the loss from instance-level to bag-level, which creates a solution to the dual labeling ambiguity issue. An alternating optimization algorithm is designed to optimize the proposed formulation after relaxing it to two approximation variants. We have compared the proposed models to the state of the art multi-instance learning and multi-labeler learning methods. Empirical results on NLP benchmark data sets and two real-world crowdsourced problems have demonstrated the effectiveness of the proposed approach over existing methods and proved the need for such a technique to address the dual ambiguity problem.

There are several limitations of the current work. We have not examined other potential solvers that explore the bi-convexity property of the proposed formulation (3.2), which may motivate better relaxations to the integrative formulation (3.3). Given the stochastic nature of the formulated optimization problem (3.3) and its relaxations (e.g., the sub-problem in (3.5)), we are unable to provide a convergence proof for Algorithm 1. More extensive empirical evaluation on real-life data sets with accessible input features and a large number of labelers may better assess the strength and weakness of the proposed approach. (Notice that many crowdsourced data sets do not provide input features for classifier training). Besides the proposed *all_min* and *selective min_max* models for estimating the true labels and labeler reliabilities, alternative models may exist to further enhance the algorithm.

Chapter 4

Multiplicative Multi-task Feature Learning

Multi-task learning (MTL) improves the generalization of the estimated models for multiple related learning tasks by capturing and exploiting the task relationships. It has been theoretically and empirically shown to be more effective than learning tasks individually. Especially when single task learning suffers from limited sample size, multi-task learning reinforces a single learning process with the transferable knowledge learned from the related tasks. Multi-task learning has been widely applied in many scientific fields, such as robotics [113], natural language processing [3], computer aided diagnosis [13], and computer vision [51].

Research efforts have been devoted to various multi-task Feature Learning (MTFL) algorithms. One direction of these works directly learns the dependencies among tasks, either by modeling the correlated regression or classification noise [34], or assuming that the model parameters share a common prior [110, 56, 105, 111, 42], or by examining the tasks' covariance matrix [16, 113, 35, 108]. Another research direction relies on a basic assumption that the

different tasks may share a substructure in the feature space. In order to exploit this shared substructure, [78] project the task parameters to explore the latent common substructure. [51] form a shared low-dimensional representation of data by feature learning. More recent methods explore the latent basis that can be used to characterize the entire set of tasks and examine the potential clusters of tasks. For instance, [73] assume that subsets of features may be shared only between tasks in the same cluster. [54] allow overlapping of tasks in different groups by having several bases in common. [64] exploit a dictionary allowing sparse representations of the tasks. [29] detect if certain tasks are outliers from the majority of the tasks.

Regularization methods are widely used in MTFL to learn a shared subset of features. A common strategy is to impose a blockwise joint regularization [65, 115] on all task model parameters to shrink the effects of features across the tasks and simultaneously minimize the regression or classification loss. These methods employ the so-called $\ell_{1,p}$ matrix norm [55, 58, 70, 114, 122] that is the sum of the ℓ_p norms of the rows in a matrix. As a result, this regularizer encourages sparsity among the rows. If a row of the parameter matrix corresponds to a feature and a column represents an individual task, the $\ell_{1,p}$ regularizer intends to rule out the unrelated features across tasks by shrinking the entire rows of the matrix to zero. Typical choices for p are 2 [70, 25] and ∞ [97]. Effective algorithms have since then been developed for the $\ell_{1,2}$ [58] and $\ell_{1,\infty}$ [77] regularization. Later, the $\ell_{1,p}$ norm is generalized to include $1 < p \leq \infty$ with a probabilistic interpretation that the resultant MTFL method solves a relaxed optimization problem with a generalized normal prior for all tasks [114]. Although the matrix-norm based regularizers lead to convex learning formulations for MTFL, recent studies show that a convex regularizer may be too relaxed to approximate the ℓ_0-type regularizer for the shrinkage effects in the feature space and thus results in suboptimal performance [30]. To address this problem, non-convex regularizers, such as the capped-

ℓ_1, ℓ_1 regularizer [30], have been proposed for the multi-task joint regularization. However, using non-convex regularizers may bring up computational challenges. For instance, non-convex formulations are usually difficult to solve, and require more complicated optimization algorithms to guarantee satisfactory performance.

For the existing MTFL methods based on joint regularization, a major limitation is that it either selects a feature as relevant to all tasks or excludes it from all models, which is very restrictive in practice where tasks may share some features but may also have their own specific features that are not relevant to other tasks. To overcome this limitation, one of the most effective strategies is to decompose the model parameters into either summation [43, 19, 29] or multiplication [103, 13, 60] of two components with separate regularizers applied to the two components. One regularizer is imposed on the component taking care of the task-specific model parameters and the other one is imposed on the component for mining the cross-feature sparsity. Specifically, for the methods that decompose the parameter matrix into summation of two matrices, the dirty model in [43] employs $\ell_{1,1}$ and $\ell_{1,\infty}$ regularizers to the two components. A robust MTFL method in [19] uses the trace norm on one component for mining a low-rank structure shared by tasks and a column-wise $\ell_{1,2}$-norm on the other component for identifying task outliers. A more recent method applies the $\ell_{1,2}$-norm both row-wisely to one component and column-wisely to the other [29]. For these additive decomposition methods, it requires the corresponding entries in both components to be zero in order to exclude a feature from a task.

For the methods that work with multiplicative decompositions, the parameter vector of each task is decomposed into an element-wise product of two vectors where one is used across tasks and the other is task-specific. To exclude a feature from a task, the multiplicative decomposition only requires one of the components to be zero. Existing methods of this line apply the same regularization to both of the component vectors, by either the ℓ_2-norm penalty

[13], or the sparse ℓ_1-norm (i.e., multi-level LASSO [60]). The multi-level LASSO method has been analytically compared to the dirty model [60], showing that the multiplicative decomposition creates better shrinkage on the global and task-specific parameters. The across-task component can screen out the features irrelevant to all tasks. An individual task's component can further select features from those selected by the cross-task component for use in its corresponding model. Although there are different ways to regularize the two components in the product, no systematic work has been done to analyze the algorithmic and statistical properties of the different regularizers. It is insightful to answer the questions such as how these learning formulations differ from the early methods based on blockwise joint regularization, how the optimal solutions of the two components intervene, and how the resultant solutions are compared with those of other methods that also learn both shared and task-specific features. We highlight the contributions of this dissertation as follows:

- We propose and examine a general framework of the multiplicative decomposition that enables a variety of regularizers to be applied. The general form corresponds to a family of MTFL methods, including all early methods that decompose model parameters as a product of two components [13, 60].

- Our theoretical analysis has revealed that this family of methods is actually equivalent to the joint regularization based approach but with a more general form of regularizers, including matrix-norm based and non-matrix-norm based regularizers. The non-matrix-norm based joint regularizers derived from the proposed framework have never been considered previously. If they are considered in the joint regularization form, the resultant optimization problems will be difficult to solve. However, our equivalent multiplicative MTFL framework in this case uses convex regularizers on the two components, which can be solved efficiently.

- Further analysis reveals that the optimal solution of the across-task component can be analytically computed by a formula of the optimal task-specific parameters. This analytical result facilitates a better understanding of the shrinkage effects of the different regularizers applied to the two components.

- Statistical justification is also derived for this family of formulations. It proves that the proposed framework is equivalent to maximizing a lower bound of the *maximum a posterior* solution under a probabilistic model that assumes generalized normal priors on model parameters.

- Two new MTFL formulations are derived from the proposed general framework. Unlike the existing methods [13, 60] where the same kind of vector norm is applied to both components, the shrinkage of the global and task-specific parameters differs in the new formulations. We empirically illustrate the scenarios where the two new formulations are more suitable for solving the MTFL problems.

- An efficient blockwise coordinate descent algorithm is derived suitable for solving the entire family of the methods. Given some of the methods (including the two new formulations we study) correspond to non-matrix-norm based joint regularizers, our algorithm provides a powerful alternative to solving the related difficult optimization problems, allowing us to explore the behaviors and properties of these regularizers in an effective way. Convergence analysis is thoroughly discussed.

To depict the differences between our approach and previous methods, Table 4.1 summaries various regularizers used in the joint regularization based and model decomposition based MTFL methods. There are some fundamental connections among these methods. As studied in the present work, multiplicative MTFL models can be connected with the early

Table 4.1: The regularization terms used in various MTFL methods.

	Model	Methods	Norms
Joint regularization models	$\mathbf{A}$	Evgeniou and Potil (2004)	$\ell_{1,2}$
		Turlach et al. (2005)	$\ell_{1,\infty}$
		Lee et al. (2010)	both $\ell_{1,1}$ and $\ell_{1,2}$
		Zhang et al. (2010)	$\ell_{1,p}, 1 < p \leq \infty$
		Gong et al. (2012a)	capped $\ell_{1,1}$
Decomposed models	$\mathbf{A} = \mathbf{P} + \mathbf{Q}$	Jalali et al. (2010)	$\ell_{1,1}$ on $\mathbf{P}$, $\ell_{1,\infty}$ on $\mathbf{Q}$
		Gong et al. (2012b)	$\ell_{1,2}$ on $\mathbf{P}$, $\ell_{1,2}$ on $\mathbf{Q}^T$
	$\mathbf{A} = \text{diag}(\mathbf{c}) \cdot \mathbf{B}$	**The proposed framework**	$(\ell_k)^k$ on $\mathbf{c}$, $(\ell_p)^p$ on $\mathbf{B}$
		Bi et al. (2008)	k=2, p=2
		Lozano and Swirszcz (2012)	k=1, p=1
		The new formulation 1	k=1, p=2
		The new formulation 2	k=2, p=1

blockwise joint regularization models. We also attempt to empirically compare the model behaviors between the multiplicative and additive MTFL methods, and particularly compare the applicability of the four different choices of regularization listed in Table 4.1 for multiplicative MTFL .

The rest of this chapter is organized as follows. Section 4.1 defines the mathematical notation and introduces the proposed models in detail. Section 4.2 discusses several important theoretical properties of the proposed models including equivalence analysis. Section 4.3 provides the statistical justification of the multiplicative MTFL models. In Section 4.4, we develop an efficient algorithm to solve the optimization problems in the proposed models with a convergence analysis. Section 4.5 shows the empirical results, in which simulations have been designed to examine the various feature sharing patterns for which a specific choice of regularizer may be preferred. Extensive experiments with a variety of classification and regression benchmark datasets are also described in Section 4.5. In Section 4.6, we conclude this work.

4.1 The Proposed Multiplicative MTFL

Given T tasks in total, for each task t, $t \in \{1, \cdots, T\}$, we have sample set ($\mathbf{X}_t \in \mathbb{R}^{\ell_t \times d}, \mathbf{y}_t \in \mathbb{R}^{\ell_t}$). The dataset of $\mathbf{X}_t$ has ℓ_t examples, where the i-th row corresponds to the i-th example $\mathbf{x}_i^t$ of task t, $i \in \{1, \cdots, \ell_t\}$, and each column represents a feature and there are totally d features. The vector $\mathbf{y}_t$ contains y_i^t, the label of the i-th example of task t. We consider functions of the linear form $y_t = \boldsymbol{\alpha}_t^\top \mathbf{x}_i^t$ where $\boldsymbol{\alpha}_t \in \mathbb{R}^d$, which corresponds to computing $\mathbf{X}_t \boldsymbol{\alpha}_t$ on the training data as the estimate of $\mathbf{y}_t$. We define the parameter matrix or weight matrix $\mathbf{A} = [\boldsymbol{\alpha}_1, \cdots, \boldsymbol{\alpha}_T]$ and denote the rows of $\mathbf{A}$ by $\boldsymbol{\alpha}^j$, $j \in \{1, \cdots, d\}$.

The joint regularization based MTFL method with the $\ell_{1,p}$ regularizer minimizes

$$\sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \lambda \sum_{j=1}^{d} ||\boldsymbol{\alpha}^j||_p \tag{4.1}$$

for the best $\boldsymbol{\alpha}_{t:t=1,\cdots,T}$, where $L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t)$ is the loss function of task t which computes the discrepancy between the observed $\mathbf{y}_t$ and the model output $\mathbf{X}_t \boldsymbol{\alpha}_t$, and λ is a tuning parameter to balance between the loss and the regularizer. Although any suitable loss function can be used in the formulation (4.1), convex loss functions are the common choices such as the least squares loss $\sum_{i=1}^{\ell_t} (y_i^t - \boldsymbol{\alpha}_t^\top \mathbf{x}_i^t)^2$ for regression problems or the logistic loss $\sum_{i=1}^{\ell_t} \log(1 + e^{-y_i^t (\boldsymbol{\alpha}_t^\top \mathbf{x}_i^t)})$ for classification problems. These loss functions are strictly convex with respect to the model parameters $\boldsymbol{\alpha}_t$. The ℓ_p norm is computed for each row of the matrix $\mathbf{A}$ corresponding to a feature (rather than a task) so to enforce sparsity on the features.

A family of multiplicative MTFL methods can be derived by rewriting $\boldsymbol{\alpha}_t = \text{diag}(\mathbf{c})\boldsymbol{\beta}_t$ where $\text{diag}(\mathbf{c})$ is a diagonal matrix with its diagonal elements composing a vector $\mathbf{c}$. The $\mathbf{c}$ vector is used across all tasks, indicating if a feature is useful for any of the tasks, and the vector $\boldsymbol{\beta}_t$ is only for task t. Let j index the entries in these vectors. We have $\alpha_j^t = c_j \beta_j^t$.

Typically $\mathbf{c}$ comprises binary entries that are equal to 0 or 1, but the integer constraint is often relaxed to require just non-negativity ($\mathbf{c} \geq 0$). We minimize a regularized loss function as follows for the best $\mathbf{c}$ and $\boldsymbol{\beta}_{t:t=1,\cdots,T}$:

$$\min_{\boldsymbol{\beta}_t, \mathbf{c} \geq 0} \quad \sum_{t=1}^{T} L(\mathbf{c}, \boldsymbol{\beta}_t, \mathbf{X}_t, \mathbf{y}_t) + \gamma_1 \sum_{t=1}^{T} ||\boldsymbol{\beta}_t||_p^p + \gamma_2 ||\mathbf{c}||_k^k \tag{4.2}$$

where $L(\mathbf{c}, \boldsymbol{\beta}_t, \mathbf{X}_t, \mathbf{y}_t)$ is the same loss function used in Eq.(4.1) but with $\boldsymbol{\alpha}_t$ replaced by the new vector of $(c_j \beta_j^t)_{j=1,\cdots,d}$, $||\boldsymbol{\beta}_t||_p^p = \sum_{j=1}^{d} |\beta_j^t|^p$ and $||\mathbf{c}||_k^k = \sum_{j=1}^{d} (c_j)^k$, which are the ℓ_p-norm of $\boldsymbol{\beta}_t$ to the power of p and the ℓ_k-norm of $\mathbf{c}$ to the power of k if p and k are positive integers. The tuning parameters γ_1, γ_2 are used to balance the empirical loss and regularizers. At optimality, if $c_j = 0$, the j-th variable is removed for all tasks, and the corresponding row vector $\boldsymbol{\alpha}^j = \mathbf{0}$; otherwise the j-th variable is selected for use in at least one of the $\boldsymbol{\alpha}$'s. Then, a specific $\boldsymbol{\beta}_t$ can rule out the j-th variable from task t if $\beta_j^t = 0$.

If both $p = k = 2$, Problem (4.2) becomes the formulation used in [13]. Since the ℓ_2-norm regularization is applied on both $\boldsymbol{\beta}_t$ and $\mathbf{c}$, this model does not impose strong sparsity on the model parameters. According to our empirical study, this model may be suitable for the scenarios where only a few features can be excluded from all of the tasks, and the different models (tasks) share a lot features between each other. There could exist features that, although irrelevant to most of the tasks, cannot be completely excluded only due to few tasks.

If $p = k = 1$, Problem (4.2) becomes the formulation used in [60], where the ℓ_1-norm regularization is applied on $\boldsymbol{\beta}_t$ and $\mathbf{c}$ and thus it induces very strong sparsity both on task-specific parameters and the across-task component to select the features. Compared to the model in [13], this model is more suitable for learning from the tasks with persistently sparse models. For example, many features are irrelevant to any of the tasks, and only a few of

the features selected by $\mathbf{c}$ are used by an individual task, indicating the sparse pattern in sharing the selected features among tasks.

Besides the above two existing models, any other choices of p and k will derive into new formulations for MTFL. Note that the two existing methods discussed in [13, 60] use $p = k$ in their formulations, which renders β_j^t and c_j the same amount of shrinkage. To explore other feature sharing patterns among tasks, we propose two new formulations where $p \neq k$.

Formulation 1:

If $p = 2$ but $k = 1$ in Problem (4.2), then we obtain the following optimization problem

$$\min_{\boldsymbol{\beta}_t, \mathbf{c} \geq 0} \sum_{t=1}^{T} L(\mathbf{c}, \boldsymbol{\beta}_t, \mathbf{X}_t, \mathbf{y}_t) + \gamma_1 \sum_{t=1}^{T} ||\boldsymbol{\beta}_t||_2^2 + \gamma_2 ||\mathbf{c}||_1. \tag{4.3}$$

When there exists a large subset of features irrelevant to any of the tasks, it requires a sparsity-inducing norm on $\mathbf{c}$. However, within the relevant features selected by $\mathbf{c}$, the majority of these features are shared between tasks. In other words, the features used in each task are not sparse relative to the features selected by $\mathbf{c}$, which requires a non-sparsity-inducing norm on $\boldsymbol{\beta}$. Hence, we use ℓ_1 norm on $\mathbf{c}$ and ℓ_2 norm on each $\boldsymbol{\beta}$ in our formulation (4.2).

Formulation 2:

If $p = 1$ but $k = 2$ in Problem (4.2), we obtain the following optimization problem

$$\min_{\boldsymbol{\beta}_t, \mathbf{c} \geq 0} \sum_{t=1}^{T} L(\mathbf{c}, \boldsymbol{\beta}_t, \mathbf{X}_t, \mathbf{y}_t) + \gamma_1 \sum_{t=1}^{T} ||\boldsymbol{\beta}_t||_1 + \gamma_2 ||\mathbf{c}||_2^2. \tag{4.4}$$

When the union of the features relevant to any given tasks includes many or even all features,

the ℓ_2 norm penalty on $\mathbf{c}$ may be preferred. However, only a limited number of features are shared between tasks, i.e., the features used by individual tasks are sparse with respect to the features selected as useful across tasks by $\mathbf{c}$. We can impose the ℓ_1 norm penalty on $\boldsymbol{\beta}$.

Clearly, many other choices of p and k values can be used, such as those corresponding to higher order polynomials (e.g., $\sum_{j=1}^{d} c_j^3$). Our theoretical results in the next few sections apply to all positive value choices of p and k unless otherwise specified. In our empirical studies, however, we have implemented algorithms for the two existing models and the two new models for comparison. Some other choices of regularizers may require significant re-programming of our algorithms and we will leave them for more thorough individual examinations in the future.

4.2 Theoretical Analysis

We first extend the formulation (4.1) to allow more choices of regularizers. We introduce a new notation that is an operator applied to a vector, such as $\boldsymbol{\alpha}^j$. The operator $||\boldsymbol{\alpha}^j||_p^{p/q} = \sqrt[q]{\sum_{t=1}^{T} |\alpha_j^t|^p}$, $p, q \geq 0$, which corresponds to the ℓ_p norm if $p = q$ and both are positive integers. A joint regularized MTFL approach can solve the following optimization problem with pre-specified values of p, q and λ, for the best parameters $\boldsymbol{\alpha}_{t:t=1,\cdots,T}$:

$$\min_{\boldsymbol{\alpha}_t} \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \lambda \sum_{j=1}^{d} \sqrt{||\boldsymbol{\alpha}^j||_p^{p/q}}. \tag{4.5}$$

Our main results of this study include (i) a theorem that establishes the equivalence between the models derived from solving Problem (4.2) and Problem (4.5) for properly chosen values of λ, q, k, γ_1 and γ_2; (ii) a theorem that delineates the conditions for (4.2) to impose a convex (or concave) regularizer on the model parameter matrix $\mathbf{A}$; and (iii) an analytical

solution of Problem (4.2) for $\mathbf{c}$ which shows how the sparsity of the across-task component is relative to the sparsity of task-specific components.

Theorem 4.1. (Main Result 1) *Let $\hat{\boldsymbol{\alpha}}_t$ be the optimal solution to Problem (4.5) and ($\hat{\boldsymbol{\beta}}_t$, $\hat{\mathbf{c}}$) be the optimal solution to Problem (4.2). Then $\hat{\boldsymbol{\alpha}}_t = diag(\hat{\mathbf{c}})\hat{\boldsymbol{\beta}}_t$ when $\lambda = 2\sqrt{\gamma_1^{2-\frac{p}{kq}}\gamma_2^{\frac{p}{kq}}}$ and $q = \frac{k+p}{2k}$ (or equivalently, $k = \frac{p}{2q-1}$).*

Proof. Theorem 4.1 can be proved by establishing the following two lemmas and two theorems. The two lemmas provide the basis for the proofs of the two theorems and then from the first theorem, we conclude that the solution $\hat{\boldsymbol{\alpha}}_t$ of Problem (4.5) also minimizes the following optimization problem:

$$\min_{\boldsymbol{\alpha}_t, \boldsymbol{\sigma} \geq 0} \quad \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \mu_1 \sum_{j=1}^{d} \sigma_j^{-1} ||\boldsymbol{\alpha}^j||_p^{p/q} + \mu_2 \sum_{j=1}^{d} \sigma_j, \tag{4.6}$$

and the optimal solution of Problem (4.6) also minimizes Problem (4.5) when proper values of λ, μ_1 and μ_2 are chosen. The second theorem connects Problem (4.6) to the proposed formulation (4.2). We show that the optimal $\hat{\sigma}_j$ is equal to $(\hat{c}_j)^k$, and then the optimal $\hat{\boldsymbol{\alpha}}$ can be computed as $\text{diag}(\hat{\mathbf{c}})\hat{\boldsymbol{\beta}}_t$ from the optimal $\hat{\boldsymbol{\beta}}_t$. □

Note that when $p = 2$ and $q = 1$, the intermediate problem (4.6) uses a similar regularizer to that in [67] where $\frac{|\alpha^j|^2}{\sigma_j} + \sigma_j$ is used to approximate $|\alpha^j|$ in the ℓ_1-norm regularizer. Problem (4.6) extends the discussion in [67] to include more general regularizers according to p and q.

Lemma 4.1. *For any given $\boldsymbol{\alpha}_{t:t=1,\cdots,T}$, Problem (4.6) can be optimized with respect to $\boldsymbol{\sigma}$ by the following analytical solution*

$$\sigma_j = \mu_1^{\frac{1}{2}} \mu_2^{-\frac{1}{2}} \sqrt{||\boldsymbol{\alpha}^j||_p^{p/q}}. \tag{4.7}$$

Proof. By the Cauchy-Schwarz inequality, we can derive a lower bound for the sum of the two regularizers in Problem (4.6) as follows:

$$\mu_1 \sum_{j=1}^{d} \sigma_j^{-1} ||\boldsymbol{\alpha}^j||_p^{p/q} + \mu_2 \sum_{j=1}^{d} \sigma_j \geq 2\sqrt{\mu_1 \mu_2} \sum_{j=1}^{d} \sqrt{||\boldsymbol{\alpha}^j||_p^{p/q}} \tag{4.8}$$

where the equality holds if and only if $\sigma_j = \mu_1^{\frac{1}{2}} \mu_2^{-\frac{1}{2}} \sqrt{||\boldsymbol{\alpha}^j||_p^{p/q}}$.

Using the method of proof by contradiction, suppose that $\boldsymbol{\sigma}^*$ optimizes Problem (4.6) with a given set of $\boldsymbol{\alpha}_{t:t=1,\cdots,T}$, but $\sigma_j^* \neq \mu_1^{\frac{1}{2}} \mu_2^{-\frac{1}{2}} \sqrt{||\boldsymbol{\alpha}^j||_p^{p/q}}$. Thus, $\boldsymbol{\sigma}^*$ does not make the regularization term reach its lower bound. Then, we can choose another $\tilde{\boldsymbol{\sigma}}$ where $\tilde{\sigma}_j = \mu_1^{\frac{1}{2}} \mu_2^{-\frac{1}{2}} \sqrt{||\boldsymbol{\alpha}^j||_p^{p/q}}$, so $\tilde{\boldsymbol{\sigma}}$ delivers the lower bound in Eq.(4.8). Because the lower bound (the right hand side of Eq.(4.8)) only depends on $\boldsymbol{\alpha}$, it is a constant for fixed $\boldsymbol{\alpha}_{t:t=1,\cdots,T}$. Hence, since the loss function is also fixed for given $\boldsymbol{\alpha}_{t:t=1,\cdots,T}$, $\tilde{\boldsymbol{\sigma}}$ reaches a lower objective value of Problem (4.6) than that of $\boldsymbol{\sigma}^*$, which contradicts to the optimality of $\boldsymbol{\sigma}^*$. Therefore, the optimal $\boldsymbol{\sigma}$ always takes the form of Eq.(4.7). □

Remark 4.1. Based on the proof of Lemma 4.1, we also know that the objective function of (4.5) is the lower bound of the objective function of (4.6) for any given $\boldsymbol{\alpha}_t$ (including the optimal $\hat{\boldsymbol{\alpha}}_t$) when $\lambda = 2\sqrt{\mu_1 \mu_2}$, and the lower bound can be attained if and only if $\boldsymbol{\sigma}$ is set according to the formula (4.7). Hence, we can also conclude that if $(\hat{\boldsymbol{\alpha}}_t, \hat{\boldsymbol{\sigma}})$ is the optimal solution of Problem (4.6), then $\hat{\sigma}_j = \mu_1^{\frac{1}{2}} \mu_2^{-\frac{1}{2}} \sqrt{||\hat{\boldsymbol{\alpha}}^j||_p^{p/q}}$.

Lemma 4.2. *Let $\alpha_j^t = c_j \beta_j^t$ for all t and j. Replacing $\boldsymbol{\beta}_t$ by $\boldsymbol{\alpha}_t$ in Problem (4.2) yields the following optimization problem*

$$\min_{\boldsymbol{\alpha}_t, \mathbf{c} \geq 0} \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \gamma_1 \textstyle\sum_{j=1}^{d} c_j^{-p} ||\boldsymbol{\alpha}^j||_p^p + \gamma_2 \sum_{j=1}^{d} (c_j)^k. \tag{4.9}$$

For any given $\boldsymbol{\alpha}_{t:t=1,\cdots,T}$, Problem (4.9) can be optimized with respect to $\mathbf{c}$ *by the following analytical solution*

$$c_j = (\gamma_1\gamma_2^{-1}\sum_{t=1}^{T}(\alpha_j^t)^p)^{\frac{1}{p+k}}. \tag{4.10}$$

Proof. This lemma can be proved following a similar argument in the proof of Lemma 4.1. By the Cauchy-Schwarz inequality, the sum of the two regularizers in (4.9) satisfies the following inequality

$$\gamma_1\sum_{j=1}^{d}c_j^{-p}||\boldsymbol{\alpha}^j||_p^p + \gamma_2\sum_{j=1}^{d}c_j^k \geq 2\gamma_1^{\frac{k}{p+k}}\gamma_2^{\frac{p}{p+k}}\sum_{j=1}^{d}(||\boldsymbol{\alpha}^j||_p^p)^{\frac{k}{p+k}} \tag{4.11}$$

and the equality holds if and only if $\gamma_1 c_j^{-p}||\boldsymbol{\alpha}^j||_p^p = \gamma_2 c_j^k$ (and note that all parameters γ_1, γ_2, $||\hat{\boldsymbol{\alpha}}^j||_p^p$ and $\mathbf{c}$ are non-negative), which yields the following formula

$$c_j = (\gamma_1\gamma_2^{-1}\sum_{t=1}^{T}(\alpha_j^t)^p)^{\frac{1}{p+k}}.$$

Through proof by contradiction, we know that the optimal $\mathbf{c}$ has to take the above formula.

□

Based on Lemma 4.1, we will prove that Problem (4.5) is equivalent to Problem (4.6) in the sense that an optimal solution of Problem (4.5) is also an optimal solution of Problem (4.6) and vice versa when λ, μ_1, and μ_2 satisfy certain conditions.

Theorem 4.2. *The solution sets of Problem (4.5) and Problem (4.6) are identical when* $\lambda = 2\sqrt{\mu_1\mu_2}$.

Proof. First, if $\hat{\mathbf{A}} = [\hat{\boldsymbol{\alpha}}_{t:t=1,\cdots,T}]$ minimizes Problem (4.5), we show that the pair $(\hat{\mathbf{A}}, \hat{\boldsymbol{\sigma}})$ minimizes Problem (4.6) where $\hat{\sigma}_j = \mu_1^{\frac{1}{2}}\mu_2^{-\frac{1}{2}}\sqrt{||\hat{\boldsymbol{\alpha}}^j||_p^{p/q}}$.

By Remark 4.1, the objective function of (4.5) is the lower bound of the objective function of (4.6) for any given $\mathbf{A}$ (including the optimal solution $\hat{\mathbf{A}}$). When $\hat{\sigma}_j = \mu_1^{\frac{1}{2}}\mu_2^{-\frac{1}{2}}\sqrt{||\hat{\boldsymbol{\alpha}}^j||_p^{p/q}}$, Problem (4.6) reaches the lower bound. In this case, Problem (4.5) at $\hat{\mathbf{A}}$ and Problem (4.6) at $(\hat{\mathbf{A}}, \hat{\boldsymbol{\sigma}})$ have the same objective value. Now, suppose that the pair $(\hat{\mathbf{A}}, \hat{\boldsymbol{\sigma}})$ does not minimize Problem (4.6), there exists another pair $(\tilde{\mathbf{A}}, \tilde{\boldsymbol{\sigma}}) \neq (\hat{\mathbf{A}}, \hat{\boldsymbol{\sigma}})$ that achieves a lower objective value than that of $(\hat{\mathbf{A}}, \hat{\boldsymbol{\sigma}})$. By Lemma 4.1, we have that $\tilde{\sigma}_j = \mu_1^{\frac{1}{2}}\mu_2^{-\frac{1}{2}}\sqrt{||\tilde{\boldsymbol{\alpha}}^j||_p^{p/q}}$, and then Problem (4.6), at $\tilde{\mathbf{A}}$, reaches the lower bound which is formulated as the objective of Problem (4.5) when $\lambda = 2\sqrt{\mu_1\mu_2}$. In other words, the objective values of (4.6) and (4.5) are identical at $\tilde{\mathbf{A}}$. Hence, $\tilde{\mathbf{A}}$ achieves a lower objective value than that of $\hat{\mathbf{A}}$ for Problem (4.5), contradicting to the optimality of $\hat{\mathbf{A}}$.

Second, if $(\hat{\mathbf{A}}, \hat{\boldsymbol{\sigma}})$ minimizes Problem (4.6), we show that $\hat{\mathbf{A}}$ minimizes Problem (4.5).

Suppose that $\hat{\mathbf{A}}$ does not minimize Problem (4.5), which means that there exists $\tilde{\boldsymbol{\alpha}}^j$ ($\neq \hat{\boldsymbol{\alpha}}^j$ for some j) that achieves a lower objective value than that of $\hat{\boldsymbol{\alpha}}^j$. We set $\tilde{\sigma}_j = \mu_1^{\frac{1}{2}}\mu_2^{-\frac{1}{2}}\sqrt{||\tilde{\boldsymbol{\alpha}}^j||_p^{p/q}}$. Then $(\tilde{\mathbf{A}}, \tilde{\boldsymbol{\sigma}})$ is an optimal solution of Problem (4.6) as proved in the first paragraph, and will bring the objective function of Problem (4.6) to a lower value than that of $(\hat{\mathbf{A}}, \hat{\boldsymbol{\sigma}})$, contradicting to the optimality of $(\hat{\mathbf{A}}, \hat{\boldsymbol{\sigma}})$.

Therefore, Problems (4.5) and (4.6) have identical solution sets when $\lambda = 2\sqrt{\mu_1\mu_2}$. □

In order to link Problem (4.6) to our formulation (4.2), we let $\sigma_j = (c_j)^k$, $k \in \mathbb{R}$, $k \neq 0$ and $\alpha_j^t = c_j\beta_j^t$ for all t and j, and derive an equivalent objective function of Problem (4.6) based on Lemmas 4.1 and 4.2.

Theorem 4.3. *The optimal solution* $(\hat{\mathbf{A}}, \hat{\boldsymbol{\sigma}})$ *of Problem (4.6) is equivalent to the optimal solution* $(\hat{\mathbf{B}}, \hat{\mathbf{c}})$ *of Problem (4.2) where* $\hat{\alpha}_j^t = \hat{c}_j\hat{\beta}_j^t$ *and* $\hat{\sigma}_j = (\hat{c}_j)^k$ *when* $\gamma_1 = \mu_1^{\frac{kq}{2kq-p}}\mu_2^{\frac{kq-p}{2kq-p}}$, $\gamma_2 = \mu_2$, *and* $k = \frac{p}{2q-1}$.

Proof. First, if $\hat{\alpha}_j^t$ and $\hat{\sigma}_j$ optimize Problem (4.6), we show that $\hat{c}_j = \sqrt[k]{\hat{\sigma}_j}$ and $\hat{\beta}_j^t = \hat{\alpha}_j^t / \hat{c}_j$ optimize Problem (4.2).

By a change of variables (replacing $\hat{\boldsymbol{\alpha}}_t$ and $\hat{\boldsymbol{\sigma}}$ by $\hat{\boldsymbol{\beta}}_t$ and $\hat{\mathbf{c}}$ in Problem (4.6)), we know that $\hat{\boldsymbol{\beta}}_t$ and $\hat{\mathbf{c}}$ minimize the following objective function

$$J(\hat{\beta}_j^t, \hat{c}_j) = \sum_{t=1}^{T} L(\hat{\mathbf{c}}, \hat{\boldsymbol{\beta}}_t, \mathbf{X}_t, \mathbf{y}_t) + \mu_1 \sum_{j=1}^{d} ||\hat{\boldsymbol{\beta}}^j||_p^{p/q} \hat{c}_j^{(p-kq)/q} + \mu_2 \sum_{j=1}^{d} (\hat{c}_j)^k. \tag{4.12}$$

By Lemma 4.1 and Remark 4.1, the optimal $\hat{\sigma}_j = \mu_1^{\frac{1}{2}} \mu_2^{-\frac{1}{2}} \sqrt{||\hat{\boldsymbol{\alpha}}^j||_p^{p/q}}$. Because $\hat{c}_j = \sqrt[k]{\hat{\sigma}_j}$, we derive that

$$\hat{c}_j = \left(\mu_1 \mu_2^{-1} ||\hat{\boldsymbol{\beta}}^j||_p^{p/q} \right)^{\frac{q}{2kq-p}}.$$

Substituting the formula of $\hat{c}_j$ into Eq.(4.12) yields the same objective function of Problem (4.2) after replacing μ_1 and μ_2 by γ_1 and γ_2 with the equations $\gamma_1 = \mu_1^{\frac{kq}{2kq-p}} \mu_2^{\frac{kq-p}{2kq-p}}$, $\gamma_2 = \mu_2$. Therefore, $\hat{\boldsymbol{\beta}}_t$ and $\hat{\mathbf{c}}$ optimize Problem (4.2) because otherwise, if any other solution $(\boldsymbol{\beta}, \mathbf{c})$ can further reduce the objective value of Problem (4.2), then the corresponding $\boldsymbol{\alpha}$ and $\boldsymbol{\sigma}$ will bring the objective function of Problem (4.6) to a lower value than $\hat{\boldsymbol{\alpha}}$ and $\hat{\boldsymbol{\sigma}}$.

Next, if $\hat{\beta}_j^t$ and $\hat{c}_j$ optimize Problem (4.2), we show that $\hat{\alpha}_j^t = \hat{c}_j \hat{\beta}_j^t$ and $\hat{\sigma}_j = (\hat{c}_j)^k$ optimize Problem (4.6).

Substituting $\hat{\alpha}_j^t$, $\hat{\sigma}_j$ for $\hat{\beta}_j^t$, $\hat{c}_j$ in Problem (4.2) yields an objective function

$$J(\hat{\alpha}_j^t, \hat{\sigma}_j) = \sum_{t=1}^{T} L(\hat{\boldsymbol{\alpha}}_t, \mathbf{X}_t, \mathbf{y}_t) + \gamma_1 \sum_{j=1}^{d} ||\hat{\boldsymbol{\alpha}}^j||_p^p \hat{\sigma}_j^{-(p/k)} + \gamma_2 \sum_{j=1}^{d} \hat{\sigma}_j. \tag{4.13}$$

We hence know that $\hat{\alpha}_j^t$ and $\hat{\sigma}_j$ minimize Eq.(4.13). Similar to the proof of Lemma 4.2, we

can show that the optimal $\hat{\boldsymbol{\sigma}}$ takes the form of

$$\hat{\sigma}_j = (\gamma_1\gamma_2^{-1})^{\frac{k}{k+p}}(||\hat{\boldsymbol{\alpha}}^j||_p^p)^{\frac{k}{k+p}}$$

Substituting the formula into Eq.(4.13) and setting $\gamma_1 = \mu_1^{\frac{kq}{2kq-p}}\mu_2^{\frac{kq-p}{2kq-p}}$ and $\gamma_2 = \mu_2$ transfer Eq.(4.13) to the objective function of Problem (4.6). Thus, $\hat{\alpha}_j^t$ and $\hat{\sigma}_j$ optimize Problem (4.6).

□

Now, based on the above two lemmas and two theorems, we can derive that when $\lambda = 2\sqrt{\gamma_1^{2-\frac{p}{kq}}\gamma_2^{\frac{p}{kq}}}$ and $q = \frac{k+p}{2k}$, the optimal solutions to Problems (4.2) and (4.5) are equivalent. Solving Problem (4.2) will yield an optimal solution $\hat{\boldsymbol{\alpha}}$ to Problem (4.5) and vice versa.

By the equivalence analysis, the proposed framework corresponds to a family of joint regularization methods as defined by Eq.(4.5). Assuming a convex loss function is used, this family includes some convex formulations when convex regularizers are applied to $\mathbf{A}$ in (4.5) and some other non-convex formulations when non-matrix-norm based regularizers are applied to $\mathbf{A}$. Particularly, when $q = p/2$, the regularization term on $\boldsymbol{\alpha}^j$ in (4.5) becomes the standard ℓ_p-norm. Correspondingly, when $k = p/(p-1)$ which is commonly not an integer except $p = 2$, our formulation (4.2) amounts to imposing a $\ell_{1,p}$-norm on $\mathbf{A}$. When both k and p take positive integers (except $p = k = 2$), Problem (2) is equivalent to using a non-matrix-norm regularizer in (4.5). Combinations of different p and k values will render the models different algorithmic behaviors.

Theorem 4.4. (Main Result 2) *For any positive k and p, if $kp \geq k + p$, then the formulation (2) imposes a convex regularizer on the model parameter matrix $\mathbf{A}$; or otherwise, it imposes a concave regularizer on $\mathbf{A}$.*

Proof. According to Theorem 1, the formulation (2) is equivalent to the formulation (5) that imposes the following regularizer on $\mathbf{A}$:

$$\lambda \sum_{j=1}^{d} \sqrt{||\boldsymbol{\alpha}^j||_p^{p/q}} = \lambda \sum_{j=1}^{d} (||\boldsymbol{\alpha}^j||_p)^{\frac{p}{2q}} = \lambda \sum_{j=1}^{d} (||\boldsymbol{\alpha}^j||_p)^{\frac{kp}{k+q}}. \tag{4.14}$$

where $q = \frac{k+p}{2k}$ and $\lambda = 2\sqrt{\gamma_1^{2-\frac{p}{kq}} \gamma_2^{\frac{p}{kq}}}$.

A function x^a is a convex function in terms of $x > 0$ if $a \geq 1$; or otherwise, it is concave. Hence, if $kp \geq k + p$, Eq.(4.14) is a composite function of two functions: a convex function $x^{\frac{kp}{k+p}}$ and another convex function which is the ℓ_p vector norm of $\boldsymbol{\alpha}^j$. Thus, the overall regularizer is convex. Otherwise, if $kp < k + p$, the regularizer becomes a concave function in terms of $\boldsymbol{\alpha}$'s. □

Remark 4.2. For a particular choice of $p = 2$ and $k = 2$, Problem (4.2) is formulated as

$$\min_{\boldsymbol{\beta}_t, \mathbf{c} \geq 0} \sum_{t=1}^{T} L(\mathbf{c}, \boldsymbol{\beta}_t, \mathbf{X}_t, \mathbf{y}_t) + \gamma_1 \sum_{t=1}^{T} ||\boldsymbol{\beta}_t||_2^2 + \gamma_2 ||\mathbf{c}||_2^2 \tag{4.15}$$

which is used in [13]. This problem is equivalent to the following joint regularization method as used in [70, 5].

$$\min_{\boldsymbol{\alpha}_t} \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \lambda \sum_{j=1}^{d} \sqrt{\sum_{t=1}^{T} (\alpha_j^t)^2} \tag{4.16}$$

when $\lambda = 2\sqrt{\gamma_1 \gamma_2}$. Problem (4.16) uses the so called $\ell_{1,2}$-norm to regularize the matrix $\mathbf{A}$.

Remark 4.3. For a particular choice of $p = 1$ and $k = 1$, Problem (4.2) is formulated as

$$\min_{\boldsymbol{\beta}_t, \mathbf{c} \geq 0} \sum_{t=1}^{T} L(\mathbf{c}, \boldsymbol{\beta}_t, \mathbf{X}_t, \mathbf{y}_t) + \gamma_1 \sum_{t=1}^{T} ||\boldsymbol{\beta}_t||_1 + \gamma_2 ||\mathbf{c}||_1 \tag{4.17}$$

which is used in [60]. This problem is equivalent to the following joint regularization method

$$\min_{\boldsymbol{\alpha}_t} \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \lambda \sum_{j=1}^{d} \sqrt{\sum_{t=1}^{T} |\alpha_j^t|} \tag{4.18}$$

when $\lambda = 2\sqrt{\gamma_1 \gamma_2}$. Problem (4.17) imposes stronger sparsity on $\boldsymbol{\beta}$ and $\mathbf{c}$ (and correspondingly on $\boldsymbol{\alpha}$) than (4.15), which intends to shrink more model parameters to zero. Problem (4.18) uses a concave non-matrix-norm regularizer.

Remark 4.4. For a particular choice of $p = 2$ and $k = 1$, which corresponds to the new formulation (4.3). This problem is equivalent to the following joint regularization method

$$\min_{\boldsymbol{\alpha}_t} \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \lambda \sum_{j=1}^{d} \sqrt[3]{\sum_{t=1}^{T} (\alpha_j^t)^2} \tag{4.19}$$

when $\lambda = 2\gamma_1^{\frac{1}{3}} \gamma_2^{\frac{2}{3}}$. Problem (4.19) uses a concave non-matrix-norm regularizer as well but the concavity is weaker than Problem (4.18) in the sense that the polynomial order is $\frac{2}{3}$ rather than $\frac{1}{2}$.

The proposed formulation (4.3) imposes stronger sparsity induction on the across-task component than on the task-specific component. Thus, it has stronger shrinkage effects to exclude many features for all the tasks. If the jth feature is considered as unrelated to most of the tasks, the model of $p = k = 2$ may shrink c_j to a small value but not zero, the new model (4.3) might shrink c_j to zero instead. Therefore this model would be more favorable to jointly learning from tasks where a large portion of noisy features exist that may be irrelevant or redundant to all of the tasks. Compared to the method in [60] where $p = k = 1$, the new formulation (4.3) may allow more selected features to be shared across different tasks.

Remark 4.5. For a particular choice of $p = 1$ and $k = 2$, which corresponds to the new formulation (4.4). This problem is equivalent to the following joint regularization method

$$\min_{\boldsymbol{\alpha}_t} \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \lambda \sum_{j=1}^{d} \sqrt[3]{\left(\sum_{t=1}^{T} |\alpha_j^t| \right)^2} \tag{4.20}$$

when $\lambda = 2\gamma_1^{\frac{2}{3}}\gamma_2^{\frac{1}{3}}$. Problem (4.20) uses another concave non-matrix-norm regularizer that has the similar polynomial order of $\frac{2}{3}$ to Problem (4.19). In comparison with (4.19), the cross-task quadratic terms (e.g., $|\alpha_j^1||\alpha_j^2|$, $|\alpha_j^2||\alpha_j^3|$) are allowed inside the cube root in this formulation.

The new formulation (4.4) imposes stronger sparsity induction on task-specific component than on the across-task vector. This model can be favorable in the cases where few tasks share a limited number of selected features. As shown in our empirical results, for instance, when every two or three tasks share a limited subset of selected features but no common features are used by more than three tasks, this model performs the best among the four multiplicative MTFL formulations. In comparison to the model in [60], this model allows more of the features that are only relevant to a few tasks to be selected. In comparison with the model in [13], this model can help to remove a lot of irrelevant or redundant features for individual tasks from the selected features.

We further derive another main result that characterizes the optimal across-task vector as a formula of the optimal task-specific vectors. This connection can be easily developed from the above equivalence analysis, and can help us understand the relationship and interaction between the two components.

Theorem 4.5. (Main Result 3) *Let $\hat{\boldsymbol{\beta}}_t$, $t = 1, \cdots, T$, be the optimal solutions of Problem*

(4.2), Let $\hat{\mathbf{B}} = [\hat{\boldsymbol{\beta}}_1, \cdots, \hat{\boldsymbol{\beta}}_T]$ *and* $\hat{\boldsymbol{\beta}}^j$ *denote the* j*-th row of the matrix* $\hat{\mathbf{B}}$*. Then,*

$$\hat{c}_j = (\gamma_1/\gamma_2)^{\frac{1}{k}} \sqrt[k]{\sum_{t=1}^{T} (\hat{\beta}_j^t)^p} \tag{4.21}$$

for all $j = 1, \cdots, d$*, is optimal to Problem (4.2).*

Proof. This analytical formula can be directly derived from Theorem 4.2 and Theorem 4.3. When we set $\hat{\sigma}_j = (\hat{c}_j)^k$ and $\hat{\alpha}_j^t = \hat{c}_j \hat{\beta}_j^t$ in the proof of Theorem 4.3, we obtain that $\hat{c}_j = \left(\mu_1 \mu_2^{-1} ||\hat{\boldsymbol{\beta}}^j||_p^{p/q}\right)^{\frac{q}{2kq-p}}$. In the same proof, we also show that $\mu_1 = \gamma_1^{\frac{2kq-p}{kq}} \gamma_2^{\frac{p-kq}{kq}}$ and $\mu_2 = \gamma_2$. Substituting these formulas into the formula of $\mathbf{c}$ yields $\hat{c}_j = (\gamma_1/\gamma_2)^{\frac{1}{k}} ||\hat{\boldsymbol{\beta}}^j||^{\frac{p}{2kq-p}}$. Moreover, to establish the equivalence between (4.2) and (4.5), we require $q = \frac{k+p}{2k}$, which leads to $k = 2kq - p$. Hence, $||\hat{\boldsymbol{\beta}}^j||^{\frac{p}{2kq-p}} = \sqrt[k]{\sum_{t=1}^{T} (\hat{\beta}_j^t)^p}$. We then obtain the formula (4.21). □

Remark 4.6. For particular choices of p and k, the relation between the optimal $\mathbf{c}$ and $\boldsymbol{\beta}$ can be computed according to Theorem 4.5. Table 4.2 summarizes the relation formula for the common choices when $p \in \{1, 2\}$ and $k \in \{1, 2\}$.

Table 4.2: The shrinkage effect of $\mathbf{c}$ with respect to $\boldsymbol{\beta}$ for four common choices of p and k.

p	k	$\mathbf{c}$
2	2	$\hat{c}_j = \sqrt{\gamma_1 \gamma_2^{-1}} \sqrt{\sum_{t=1}^{T} (\hat{\beta}_j^t)^2}$
1	1	$\hat{c}_j = \gamma_1 \gamma_2^{-1} \sum_{t=1}^{T} \lvert\hat{\beta}_j^t\rvert$
2	1	$\hat{c}_j = \gamma_1 \gamma_2^{-1} \sum_{t=1}^{T} (\hat{\beta}_j^t)^2$
1	2	$\hat{c}_j = \sqrt{\gamma_1 \gamma_2^{-1}} \sqrt{\sum_{t=1}^{T} \lvert\hat{\beta}_j^t\rvert}$

4.3 Probabilistic Interpretation

In this section we show that the proposed multiplicative formalism is related to the *maximum a posteriori* (MAP) solution of a probabilistic model. Let $p(\mathbf{A}|\boldsymbol{\Delta})$ be the prior distribution of the weight matrix $\mathbf{A} = [\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_T] = [\boldsymbol{\alpha}^{1\top}, \ldots, \boldsymbol{\alpha}^{d\top}]^\top \in \mathbb{R}^{d\times T}$, where $\boldsymbol{\Delta}$ denote the parameter of the prior. Then the *a posteriori* distribution of $\mathbf{A}$ can be calculated via Bayes rule as

$$p(\mathbf{A}|\mathbf{X}, \mathbf{y}, \boldsymbol{\Delta}) \propto p(\mathbf{A}|\boldsymbol{\Delta}) \prod_{t=1}^{T} p(\mathbf{y}_t|\mathbf{X}_t, \boldsymbol{\alpha}_t). \tag{4.22}$$

Denote $z \sim \mathcal{GN}(\mu, \rho, p)$ the *univariate generalized normal distribution*, with the density function

$$p(z) = \frac{1}{2\rho\Gamma(1+1/p)} \exp\left(-\frac{|z-\mu|^p}{\rho^p}\right), \tag{4.23}$$

in which $\rho > 0$, $p > 0$, and $\Gamma(\cdot)$ is the Gamma function [32]. Now let each element of $\mathbf{A}$, α_j^t, follow a generalized normal prior, $\alpha_j^t \sim \mathcal{GN}(0, \delta_j, p)$. Then with the *independent and identically distributed* assumption, the prior takes the form (also refer to [114] for a similar treatment)

$$p(\mathbf{A}|\boldsymbol{\Delta}) \propto \prod_{j=1}^{d}\prod_{t=1}^{T} \frac{1}{\delta_j} \exp\Big(-\frac{|\alpha_j^t|^p}{\delta_j^p}\Big) = \prod_{j=1}^{d} \frac{1}{\delta_j^T} \exp\Big(-\frac{\|\boldsymbol{\alpha}^j\|_p^p}{\delta_j^p}\Big), \tag{4.24}$$

where $\|\cdot\|_p$ denotes the vector p-norm. With an appropriately chosen likelihood function $p(\mathbf{y}_t|\mathbf{X}_t, \boldsymbol{\alpha}_t) \propto \exp(-L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t))$, finding the MAP solution is equivalent to solving the

following problem:

$$\min_{\mathbf{A},\boldsymbol{\Delta}} \quad J = \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \sum_{j=1}^{d} \Big(\frac{\|\boldsymbol{\alpha}^j\|_p^p}{\delta_j^p} + T \ln \delta_j \Big). \tag{4.25}$$

Setting the derivative of J with respect to δ_j to zero, we obtain:

$$\delta_j = \Big(\frac{p}{T}\Big)^{1/p} \|\boldsymbol{\alpha}^j\|_p. \tag{4.26}$$

Bringing this back to (4.25), we have the following equivalent problem:

$$\min_{\mathbf{A}} \quad J = \sum\nolimits_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + T \sum\nolimits_{j=1}^{d} \ln \|\boldsymbol{\alpha}^j\|_p. \tag{4.27}$$

Now let us look at the multiplicative nature of α_j^t with different $p \in [1, \infty)$.

When $p = 1$, we have:

$$\begin{aligned}
\sum_{j=1}^{d} \ln \|\boldsymbol{\alpha}^j\|_1 &= \sum_{j=1}^{d} \ln \left(\sum_{t=1}^{T} |\alpha_j^t| \right) \\
&= \sum_{j=1}^{d} \ln \left(\sum_{t=1}^{T} |c_j \beta_j^t| \right) \\
&= \sum_{j=1}^{d} \left(\ln |c_j| + \ln \sum_{t=1}^{T} |\beta_j^t| \right) \\
&\leq \sum_{j=1}^{d} |c_j| + \sum_{j=1}^{d} \sum_{t=1}^{T} |\beta_j^t| - 2d,
\end{aligned}$$

where the inequality is because of the fact that $\ln z \leq z - 1$ for any $z > 0$. Therefore we can

optimize an upper bound of J in (4.27),

$$\min_{\mathbf{A}} \; J_1 = \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + T \sum_{j=1}^{d} |c_j| + T \sum_{j=1}^{d} \sum_{t=1}^{T} |\beta_j^t| - 2dT, \tag{4.28}$$

which is equivalent to the multiplicative formulation (4.2) where $\{p, k\} = \{1, 1\}$. This proves the following theorem:

Theorem 4.6. *When $\{p, k\} = \{1, 1\}$, optimizing the multiplicative formulation* (4.2) *is equivalent to maximizing a lower bound of the MAP solution under probabilistic model* (4.22) *with $p = 1$ in the prior definition.*

In the general case, we have:

$$\begin{aligned}
\sum_{j=1}^{d} \ln \|\boldsymbol{\alpha}^j\|_p &= \frac{1}{p} \sum_{j=1}^{d} \ln \left(\sum_{t=1}^{T} |c_j \beta_j^t|^p \right) \\
&= \frac{1}{p} \sum_{j=1}^{d} \ln \left((c_j^k)^{\frac{p}{k}} \cdot \sum_{t=1}^{T} |\beta_j^t|^p \right) \\
&= \frac{1}{k} \sum_{j=1}^{d} \ln |c_j|^k + \frac{1}{p} \sum_{j=1}^{d} \ln \sum_{t=1}^{T} |\beta_j^t|^p \\
&\leq \frac{1}{k} \sum_{j=1}^{d} |c_j|^k + \frac{1}{p} \sum_{t=1}^{T} \|\boldsymbol{\beta}_t\|_p^p - d(\frac{1}{k} + \frac{1}{p}).
\end{aligned}$$

These inequalities lead to an upper bound of J in (4.27). By minimizing the upper bound, the problem is formulated as:

$$\min_{\mathbf{A}} \quad J_{p,k} = \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \frac{T}{k} \|\mathbf{c}\|_k^k + \frac{T}{p} \sum_{t=1}^{T} \|\boldsymbol{\beta}_t\|_p^p, \tag{4.29}$$

which is equivalent to the general multiplicative formulation in (4.2). Therefore we have proved the following theorem:

Theorem 4.7. *Optimizing the multiplicative formulation* (4.2)*, in the form of* (4.29)*, is equivalent to maximizing a lower bound of the MAP solution under probabilistic model* (4.22) *with* $p \in (1, \infty)$ *in the prior definition.*

4.4 Optimization Algorithm

Alternating optimization algorithms have been used in both of the early methods [13, 60] to solve Problem (4.2) which alternate between solving two subproblems: solve for $\boldsymbol{\beta}_t$ with fixed $\mathbf{c}$; solve for $\mathbf{c}$ with fixed $\boldsymbol{\beta}_t$. The convergence property of such an alternating algorithm has been analyzed in [13, 60] that it converges to a local minimizer. In these early methods, both of the two subproblems have to be solved using iterative algorithms such as gradient descent, linear or quadratic program solvers.

Besides the algorithm that solves for $\mathbf{c}$ and $\boldsymbol{\beta}_t$ alternatively and can be applied to our formulations, we design an alternating optimization algorithm that utilizes the closed-form solution for $\mathbf{c}$ we have derived in Lemma 4.2 and the property that both Problems (4.2) and (4.5) are equivalent to the intermediate Problem (4.6) (or Problem (4.9)). In the algorithm to solve Problem (4.2), we start from an initial choice of $\mathbf{c}$. At iteration s, we start from $\mathbf{c}^s$, and solve for $\boldsymbol{\beta}_t^s$ with the fixed $\mathbf{c}^s$. We then compute the value of $\boldsymbol{\alpha}_t^s = \text{diag}(\mathbf{c}^s)\boldsymbol{\beta}_t^s$, which is used to update $\mathbf{c}^s$ to $\mathbf{c}^{s+1}$ according to Eq.(4.10) in Lemma 4.2. The overall procedure is summarized in **Algorithm 3**. As a central idea in designing this algorithm, at each iteration we update $\boldsymbol{\beta}$ to reduce the loss function, and update $\mathbf{c}$ to reduce the regularizer while maintaining the same loss function value. Note that if the value of $\boldsymbol{\alpha}$ is fixed, the loss will remain the same.

To analyze the convergence property of Algorithm 1, we utilize the fact that Problem

Algorithm 3 The blockwise coordinate descent algorithm for multiplicative MTFL

Input: $\mathbf{X}_t$, $\mathbf{y}_t$, $t = 1, \cdots, T$, as well as γ_1, γ_2, p and k
Initialize: $c_j = 1$, $\forall j = 1, \cdots, d$, and $s = 1$
repeat
  Compute $\tilde{\mathbf{X}}_t = \mathbf{X}_t \text{diag}(\mathbf{c}^s)$, $\forall\ t = 1, \cdots, T$
  for $t = 1, \cdots, T$ **do**
    Solve the following problem for $\boldsymbol{\beta}_t^s$

$$\min_{\boldsymbol{\beta}_t} L(\boldsymbol{\beta}_t, \tilde{\mathbf{X}}_t, \mathbf{y}_t) + \gamma_1 ||\boldsymbol{\beta}_t||_p^p \tag{4.30}$$

  end for
  Compute $\boldsymbol{\alpha}_t^s = \text{diag}(\mathbf{c}^s)\boldsymbol{\beta}_t^s$
  Set $s = s + 1$
  Compute $\mathbf{c}^{s+1}$ using $\boldsymbol{\alpha}_t^s$ according to Eq.(4.10)
until $\max_{t,j}(|(\alpha_j^t)^s - (\alpha_j^t)^{s-1}|) < \epsilon$ (or other proper termination rules)
Output: $\boldsymbol{\alpha}_t$, $\mathbf{c}$ and $\boldsymbol{\beta}_t$, $t = 1, \cdots, T$

(4.2) and Problem (4.9) are equivalent. For notational convenience, we denote the objective function of Problem (4.2) by $g(\mathbf{B}, \mathbf{c})$ that takes inputs $\boldsymbol{\beta}_t$ and $\mathbf{c}$. We denote the objective function of Problem (4.9) by $f(\mathbf{A}, \mathbf{c})$. Both objective functions comprise the sum of three parts. For instance, f can be written as follows:

$$f(\mathbf{A}, \mathbf{c}) = f_0(\mathbf{A}, \mathbf{c}) + f_{\mathbf{A}}(\mathbf{A}) + f_{\mathbf{c}}(\mathbf{c}), \tag{4.31}$$

where $f_0(\mathbf{A}, \mathbf{c}) = \gamma_1 \sum_{j=1}^d (c_j^{-p} \sum_{t=1}^T (\alpha_j^t)^p)$ is the part that involves both $\boldsymbol{\alpha}$ and $\mathbf{c}$, $f_{\mathbf{A}}(\mathbf{A}) = \sum_t L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t)$ is the part relying only on $\boldsymbol{\alpha}$, and $f_{\mathbf{c}}(\mathbf{c}) = \mu_2 \sum_{j=1}^d (c_j)^k$ is the part for $\mathbf{c}$ only. Let $\mathbf{z}$ be the vector consisting of all variables in Problem (4.9). Similar to what has been defined in [96], we define that the point $\mathbf{z}$ is a stationary point of f if $\mathbf{z} \in dom\ f$ where $dom\ f$ is the feasible region of f, and

$$\lim_{\lambda \to 0} [f(\mathbf{z} + \lambda \mathbf{b}) - f(\mathbf{z})]/\lambda \geq 0, \quad \forall \mathbf{b} \quad \text{such that} \quad (\mathbf{z} + \lambda \mathbf{b}) \in dom\ f. \tag{4.32}$$

where $\mathbf{b}$ denotes any feasible direction, (which corresponds to $\bigtriangledown f(\mathbf{z}) = 0$ if f is differentiable, or $\mathbf{0} \in \partial f(\mathbf{z})$ if f is non-differentiable where $\partial f(\mathbf{z})$ is the subgradient of f at $\mathbf{z}$). In our case, f is not differentiable at $\boldsymbol{\alpha} = 0$ or $\mathbf{c} = 0$ when p is set to an odd number. We also define that a point $\mathbf{z}$ is a coordinate-wise minimum point of f if $\mathbf{z} \in dom\ f$, and $\forall \mathbf{b}_k \in \mathbb{R}^{d_k}$ that makes $(0, \cdots, \mathbf{b}_k, \cdots, 0)$ a feasible direction, there exists a small $\epsilon > 0$, such that for all positive $\lambda \leq \epsilon$,

$$f(\mathbf{z} + \lambda(0, \cdots, \mathbf{b}_k, \cdots, 0)) \geq f(\mathbf{z}), \tag{4.33}$$

where k indexes the blocks of variables in our algorithm, which include $\boldsymbol{\alpha}_1, \cdots, \boldsymbol{\alpha}_T$ and $\mathbf{c}$, so $k = \{1, \cdots, T+1\}$, and the $(T+1)$-th block is for $\mathbf{c}$, d_k is the number of variables in the kth coordinate block and in our case $d_k = d$. The vector $(0, \cdots, \mathbf{b}_k, \cdots, 0)$ is a vector in $\mathbb{R}^{d \times (T+1)}$ and used to only vary $\mathbf{z}$ in the k-th block.

We first prove that for the sequence of points generated by Algorithm 3, the objective function f is monotonically non-increasing. Then we prove the sequence of points is bounded, because of which, the sequence will have accumulation points. We prove that each accumulation point is a coordinate-wise minimum point. Then according to [96], if f is regular at an accumulation point $\mathbf{z}^*$, this $\mathbf{z}^*$ is a stationary point.

Lemma 4.3. *Let the sequence of iterates generated by Algorithm 3 be* $\mathbf{z}^s = \{\mathbf{A}^s, \mathbf{c}^s\}_{s=1,2,\cdots}$ *and the function* f *is defined by (4.31), then* $f(\mathbf{A}^{s+1}, \mathbf{c}^{s+1}) \leq f(\mathbf{A}^s, \mathbf{c}^s)$.

Proof. First, note that for each $\{\mathbf{A}^s, \mathbf{c}^s\}$, we accordingly have $\{\mathbf{B}^s, \mathbf{c}^s\}$ where $\boldsymbol{\alpha}_t^s = \text{diag}(\mathbf{c}^s)\boldsymbol{\beta}_t^s$, $\forall t$, and we have $f(\mathbf{A}^s, \mathbf{c}^s) = g(\mathbf{B}^s, \mathbf{c}^s)$. Because we start with $\mathbf{c}$ so at each iteration $\mathbf{c}$ gets updated first (the order dose not matter actually). At iteration $s+1$, we compute $\mathbf{c}^{s+1}$ based on $\mathbf{A}^s$ according to Eq.(4.10). According to Lemma 4.2, $\mathbf{c}^{s+1}$ will reach the lower bound of f when $\mathbf{A}$ is fixed to $\mathbf{A}^s$. Hence $f(\mathbf{A}^s, \mathbf{c}^{s+1}) \leq f(\mathbf{A}^s, \mathbf{c}^s)$. Moreover, when $\mathbf{c}$ is updated, there is an implicit new value of $\tilde{\mathbf{B}}^s$ that is just computed as $(\tilde{\beta}_j^t)^s = (\alpha_j^t)^s/(c_j)^{s+1}$,

$\forall j$ and t. Then, $g(\tilde{\mathbf{B}}^s, \mathbf{c}^{s+1}) = f(\mathbf{A}^s, \mathbf{c}^{s+1})$. Next in Algorithm 3, we obtain $\mathbf{B}^{s+1}$ by solving Problem (4.30), i.e., by optimizing g with respect $\mathbf{B}$ when $\mathbf{c}$ is fixed to $\mathbf{c}^{s+1}$. Hence, $g(\mathbf{B}^{s+1}, \mathbf{c}^{s+1}) \leq g(\tilde{\mathbf{B}}^s, \mathbf{c}^{s+1})$. Then $\mathbf{A}$ will be updated by $\mathbf{A}^{s+1} = \text{diag}(\mathbf{c}^{s+1})\mathbf{B}^{s+1}$, which leads to $f(\mathbf{A}^{s+1}, \mathbf{c}^{s+1}) = g(\mathbf{B}^{s+1}, \mathbf{c}^{s+1})$. Overall, we have

$$f(\mathbf{A}^{s+1}, \mathbf{c}^{s+1}) = g(\mathbf{B}^{s+1}, \mathbf{c}^{s+1}) \leq g(\tilde{\mathbf{B}}^s, \mathbf{c}^{s+1}) = f(\mathbf{A}^s, \mathbf{c}^{s+1}) \leq f(\mathbf{A}^s, \mathbf{c}^s).$$

This proves that $f(\mathbf{A}^{s+1}, \mathbf{c}^{s+1}) \leq f(\mathbf{A}^s, \mathbf{c}^s)$. □

Based on the proof of Lemma 4.3, we can also show that $g(\mathbf{B}^{s+1}, \mathbf{c}^{s+1}) \leq g(\mathbf{B}^s, \mathbf{c}^s)$ for the sequence of $\{\mathbf{B}^s, \mathbf{c}^s\}_{s=1,2,\cdots}$ that is also created by Algorithm 1.

Lemma 4.4. *The sequence of iterates generated by Algorithm 3, $\mathbf{z}^s = \{\mathbf{A}^s, \mathbf{c}^s\}_{s=1,2,\cdots}$, (or equivalently, $\{\mathbf{B}^s, \mathbf{c}^s\}_{s=1,2,\cdots}$) is bounded.*

Proof. According to Lemma 4.3, we know that $f(\mathbf{A}^{s+1}, \mathbf{c}^{s+1}) \leq f(\mathbf{A}^s, \mathbf{c}^s)$, and $g(\mathbf{B}^{s+1}, \mathbf{c}^{s+1}) \leq g(\mathbf{B}^s, \mathbf{c}^s)$, $\forall s = 1, 2, \cdots$. Hence, $g(\mathbf{B}^s, \mathbf{c}^s) \leq g(\mathbf{B}^1, \mathbf{c}^1)$, $\forall s = 1, 2, \cdots$. Let $g(\mathbf{B}^1, \mathbf{c}^1) = C$. Then, $g(\mathbf{B}^s, \mathbf{c}^s)$ is upper bounded by C.

In our algorithm, we assume that the loss function in g can be either the least squares loss or the logistic regression loss of all the tasks. Hence, the loss terms are non-negative (actually most other loss functions, such as the hinge loss, are also non-negative). The two regularizers, one on $\boldsymbol{\beta}_t$ and the other on $\mathbf{c}$, are both non-negative. Thus, we have that $||\boldsymbol{\beta}_t^s||_p^p \leq C/\gamma_1$ and $||\mathbf{c}^s||_k^k \leq C/\gamma_2$, $\forall s = 1, 2, \cdots$. This shows that the sequence of $\{\mathbf{B}^s, \mathbf{c}^s\}_{s=1,2,\cdots}$ is bounded. Because $\boldsymbol{\alpha}_t^s = \text{diag}(\mathbf{c}^s)\boldsymbol{\beta}_t^s$, $||\boldsymbol{\alpha}_t^s||_p^p = \sum_{j=1}^d |(\alpha_j^t)^s|^p = \sum_{j=1}^d |(c_j^s(\beta_j^t)^s|^p \leq ||\mathbf{c}^s||_{2p}^{2p} + ||\boldsymbol{\beta}_t^s||_{2p}^{2p} \leq \tilde{C}$ where $\tilde{C}$ is a constant computed from C, γ_1 and γ_2. Thus, the sequence of $\mathbf{z}^s = \{\mathbf{A}^s, \mathbf{c}^s\}_{s=1,2,\cdots}$ is also bounded. □

Theorem 4.8. *The sequence $\mathbf{z}^s = \{\mathbf{A}^s, \mathbf{c}^s\}_{s=1,2,\cdots}$ generated by Algorithm 3 has at least*

one accumulation point. For any accumulation point $\mathbf{z}^* = \{\mathbf{A}^*, \mathbf{c}^*\}$, $\mathbf{z}^*$ *is a coordinate-wise minimum point of* f.

Proof. According to Lemma 4.4, the sequence $\mathbf{z}^s$ is bounded, so there exists at least one accumulation point $\mathbf{z}^*$ and a subsequence of $\{\mathbf{z}^s\}_{s=1,2,\cdots}$ that converges to $\mathbf{z}^*$. Without loss of generality and for notational convenience, let us just assume that $\{\mathbf{z}^s\}_{s=1,2,\cdots}$ converges to $\mathbf{z}^*$. Because $\mathbf{z}^*$ is an accumulation point, if it is the iterate at the current iteration s, then in the next iteration $s+1$, the same iterate $\mathbf{z}^*$ will be obtained. Hence, $\boldsymbol{\beta}_t^*$ is the optimal solution of Problem (4.30) when $\mathbf{c}$ is set to $\mathbf{c}^*$ (and all other $\boldsymbol{\beta}_{k\neq t} = \boldsymbol{\beta}_k^*$). Correspondingly, $\mathbf{c}^*$ is the optimal solution of Problem (4.2) when $\mathbf{B}$ is set to $\mathbf{B}^*$. Hence, for any feasible direction $(0, \cdots, \mathbf{b}_t, \cdots, 0)$, $\forall t = 1, 2, \cdots, T+1$, we have

$$f(\mathbf{z}^* + \lambda(0, \cdots, \mathbf{b}_t, \cdots, 0)) \geq f(\mathbf{z}^*) \tag{4.34}$$

for small λ values. Hence, $\mathbf{z}^*$ is a coordinate-wise minimum point of f. □

Theorem 4.9. *If both* p *and* k *are positive, and* $k \geq 1$, *the accumulation point* $\mathbf{z}^*$ *is a stationary point of* f.

Proof. Due to Lemma 4.2, we know that the optimal solution of Problem (4.9) can only occur when $\mathbf{c}$ and $\boldsymbol{\alpha}_t$ satisfy Eq.(4.10), so any other points can be excluded from discussion. Hence, we define a new level set $Z^0 = \{\mathbf{z}|f(\mathbf{z}) \leq C\} \cap \{\mathbf{z}|c_j = (\gamma_1\gamma_2^{-1}\sum_{t=1}^{T}(\alpha_j^t)^p)^{\frac{1}{p+k}}, \forall \alpha_j^t\}$. We now prove that f is continuous on this set Z^0 and the gradient of f exists when p is even (differentiable case), and examine $\mathbf{0} \in \partial f$ at $\mathbf{z}^*$ for non-differentiable cases.

Given the definition of f, f is continuous with respect to α_j^t, $\forall j$ and t. Because f_0 is a division term that has a divisor based on c_j, we have the *division-by*-0 issue, so f is in general not continuous with respect to c_j at 0 (but continuous and differentiable at other values).

However, using L'Hospital's rule (i.e., $\lim_{\phi\to 0}\frac{f_1(\phi)}{f_2(\phi}) = \lim_{\phi\to 0}\frac{f_1'(\phi)}{f_2'(\phi}$), we show that f is continuous with respect to c_j at $c_j = 0$ in the set Z^0. When $c_j = 0$, $||\boldsymbol{\alpha}^j||_p^p = \sum_{t=1}^{T}(\alpha_j^t)^p = 0$ due to Eq.(4.10). Let $\phi = ||\boldsymbol{\alpha}^j||_p^p$. Since the function f_0 is a ratio of two items both approaching 0 as functions of ϕ, we can apply L'Hospital's rule as follows

$$\lim_{\phi\to 0}\frac{||\boldsymbol{\alpha}^j||_p^p}{(c_j)^p} = \lim_{\phi\to 0}\frac{\phi}{(\gamma_1\gamma_2^{-1}\phi)^{\frac{p}{p+k}}} = \lim_{\phi\to 0}\frac{1}{\frac{p}{p+k}(\gamma_1\gamma_2^{-1})^{\frac{p}{p+k}}\phi^{-\frac{k}{p+k}}} = \lim_{\phi\to 0}\frac{(p+k)\phi^{\frac{k}{p+k}}}{p(\gamma_1\gamma_2^{-1})^{\frac{p}{p+k}}} = 0$$

We then compute the partial derivative of f_0 with respect to c_j, which is $\frac{\partial f_0}{\partial c_j} = \frac{-p||\boldsymbol{\alpha}^j||_p^p}{(c_j)^{p+1}}$ for $c_j \neq 0$. Now when c_j approaches 0, we can prove continuity using L'Hospital's rule:

$$\frac{\partial f_0}{\partial c_j}|_{c_j\to 0} = \lim_{\phi\to 0}\frac{-p||\boldsymbol{\alpha}^j||_p^p}{(c_j)^{p+1}} = \lim_{\phi\to 0}\frac{-p\phi}{(\gamma_1\gamma_2^{-1}\phi)^{\frac{p+1}{p+k}}} = \lim_{\phi\to 0}\frac{-p(p+k)\phi^{\frac{k-1}{p+k}}}{(p+1)(\gamma_1\gamma_2^{-1})^{\frac{(p+1)}{p+k}}} = 0$$

when $k > 1$. When $k = 1$, the limit is a finite number $-p(p+k)/((p+1)(\gamma_1\gamma_2^{-1})^{\frac{(p+1)}{p+k}})$. Note that when an odd p is taken, $||\boldsymbol{\alpha}_t||_p^p = \sum|\alpha_j^t|^p$ is not differentiable at 0. However, the above limit exists no matter f is differentiable or not because we take ϕ as the varying parameter. With the above conditions, we use the results in [96] that when each subproblem has a unique minimum, which is the case in our algorithm because Subproblem (4.30) is strictly convex (for our chosen loss functions) and we have already proved the unique analytical solution of $\mathbf{c}$, $\mathbf{z}^*$ is a stationary point of f. □

We briefly discuss the computation cost of Algorithm 3. Subproblem (4.30) is essentially for single task learning, which can be solved by many existing efficient algorithms, such as gradient-based optimization methods. The second subproblem has a closed-form solution for $\mathbf{c}$, which requires only a minimal level of computation. The computation cost of Algorithm 3 only linearly increases with the number of tasks. Due to the nature of Algorithm 3, it can

be easily parallelizable and distributed to multiple processors when optimizing individual $\boldsymbol{\beta}_t$. More efficient algorithms may be designed for specific choices of p in the future.

4.5 Experiments

We empirically evaluated the performance of the multiplicative MTFL algorithms on both synthetic datasets and a variety of real-world datasets, where we solved either classification (using the logistic regression loss) or regression (using the least squares loss) problems. In the experiments, we implemented and compared Algorithm 3 for four parameter settings: $(p, k) = (2, 2)$, $(1, 1)$, $(2, 1)$, and $(1, 2)$, corresponding to four multiplicative MTFL (MMTFL) methods as listed in Table 4.2. Although when the values of (p, k) were (2,2) and (1,1), the two models corresponded respectively to the same methods in [13] and [60], they were solved differently from prior methods using our Algorithm 3 with higher computational efficiency. When $(p, k) = (2, 1)$ and $(1, 2)$, the resultant models corresponded to the two new formulations.

In our experiments, the multiplicative MTFL methods were also compared with the additive MTFL methods that decompose the model parameters into an addition of two components, such as the Dirty model (DMTL) [43] and the robust MTFL (rMTFL) [29]. Single task learning (STL) approaches were also implemented as baselines and compared with all of the MTFL algorithms in the experiments. We list the various methods used for comparison in our experiments as follows:

- STL_lasso : Learning each task independently with $||\boldsymbol{\alpha}_t||_1$ as the regularizer.
- STL_ridge : Learning each task independently with $||\boldsymbol{\alpha}_t||_2^2$ as the regularizer.
- DMTL [43] : The dirty model with regularizers $||\mathbf{P}||_{1,1}$ and $||\mathbf{Q}||_{1,\infty}$, where $\mathbf{A} = \mathbf{P} + \mathbf{Q}$.

- rMTFL [29] : Robust multi-task feature learning with the regularizers $||\mathbf{P}||_{1,2}$ and $||\mathbf{Q}^T||_{1,2}$, where $\mathbf{A} = \mathbf{P} + \mathbf{Q}$.

- MMTFL$\{2, 2\}$ [13] : Multiplicative multi-task feature learning with regularizers $||\mathbf{B}||_F^2$ and $||\mathbf{c}||_2^2$, where $\mathbf{A} = \text{diag}(\mathbf{c})\mathbf{B}$.

- MMTFL$\{1, 1\}$ [60] : Multiplicative multi-task feature learning with regularizers $||\mathbf{B}||_{1,1}$ and $||\mathbf{c}||_1$, where $\mathbf{A} = \text{diag}(\mathbf{c})\mathbf{B}$.

- MMTFL$\{2, 1\}$(New formulation 1) : Multiplicative multi-task feature learning with regularizers $||\mathbf{B}||_F^2$ and $||\mathbf{c}||_1$, where $\mathbf{A} = \text{diag}(\mathbf{c})\mathbf{B}$.

- MMTFL$\{1, 2\}$(New formulation 2) : Multiplicative multi-task feature learning with regularizers $||\mathbf{B}||_{1,1}$ and $||\mathbf{c}||_2^2$, where $\mathbf{A} = \text{diag}(\mathbf{c})\mathbf{B}$.

In all experiments, unless otherwise noted, the original dataset was partitioned to have 25%, 33% or 50% of the data in a training set and the rest used for testing. For each specified partition ratio (corresponding to a trial), we randomly partitioned the data 15 times and reported the average performance. The same tuning process was used to tune the hyperparameters (e.g.,γ_1 and γ_2) of every method in the comparison. In every trial, an internal three-fold cross validation (CV) was performed within the training data of the first partition to select a proper hyperparameter value for each of the methods from the choices of 2^k with $k = -10, -9, \cdots, 7$. In the subsequent partitions of each trial, the hyperparameters were fixed to the values that yielded the best performance in the CV.

The regression performance was measured by the coefficient of determination, denoted by R^2, which measures the explained variance of the data by the fitted model. In particular, we used the following formula to report performance $R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - f_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$ where $\bar{y}$ is the mean of the observed values of y and f_i is the prediction of the observed y_i. The reported values in

our experiments were the averaged R^2 over all tasks. The R^2 value ranges from 0 to 1, and a higher value indicates better regression performance. The classification performance was measured by the F1 score, which is the harmonic mean of precision and recall. The numbers we reported in each trial was the F1 score averaged over all tasks. Similarly, the F1 score also ranges from 0 to 1 and higher values represent better classification performance.

4.5.1 Simulation Studies

We created three categories of datasets, all of which were designed for regression experiments to evaluate the behaviors of the different methods. The datasets in each category were created with a pre-specified feature sharing structure. The first two categories were designed to validate the scenarios that we hypothesized for our two new formulations to work. We performed sensitivity analyses using these two categories of datasets, i.e., studying how performance was altered when the number of tasks or the number of features varied. Because we also empirically compared with a few additive decomposition based methods, it would be interesting to see how multiplicative MTFL behaved in a scenario that was actually in favor of additive MTFL. Hence, in the third category, the feature sharing structure was generated following the assumption that the robust MTFL method used [29].

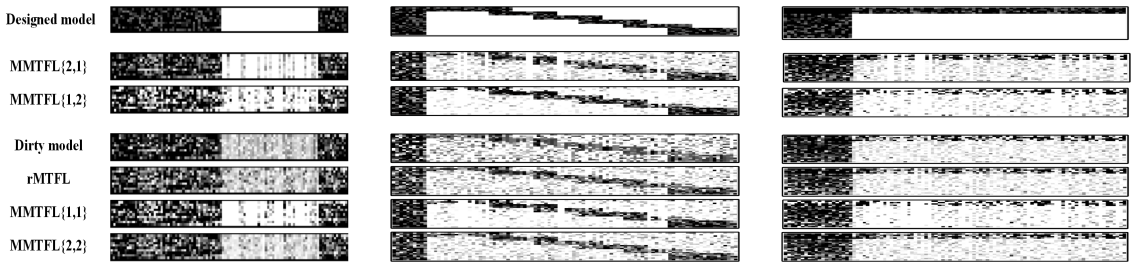
In all experiments, we created input examples $\mathbf{X}_t$ for each task t using a number of features randomly drawn from the standard multivariate Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{1})$, and pre-defined $\mathbf{A} = [\boldsymbol{\alpha}_1, \cdots, \boldsymbol{\alpha}_T]$ across all tasks. The responses for each task was computed by $\mathbf{y}_t = \mathbf{X}_t\boldsymbol{\alpha}_t + \boldsymbol{\epsilon}_t$ where $\boldsymbol{\epsilon}_t \sim \mathcal{N}(0, 0.5)$ was the noise introduced to the model. If an entry of $\mathbf{A}$ was set to non-zero, its value was randomly sampled from a uniform distribution in the interval $[0.5, 1.5]$. As a side note, we transposed $\mathbf{A}$ in Figure 4.1 for better illustration.

4.5.1.1 Synthetic Datasets

Category 1 (C1). In the C1 experiments, 40% of the rows in the matrix $\mathbf{A}$ were set to 0. Since each row corresponded to a feature across the tasks, the zero rows made the features irrelevant to all tasks. All remaining features received non-zero values in $\mathbf{A}$ so that they were used in every task's model although with different combination weights. Hence, the individual models were sparse with respect to all synthesized features, but not sparse with respect to the selected features. The pre-defined $\mathbf{A}$ is demonstrated in Figure 4.1a(top), where we transpose $\mathbf{A}$ to have columns representing features and a darker spot indicates that the particular element of $\mathbf{A}$ had a larger absolute value. Note that this synthetic data follows the assumption that has motivated the early block-wise joint regularized methods that used matrix-norm-based regularizers. Those methods were developed with an assumption that a subset of features was shared by all tasks. However, we observed that the level of shrinkage needed would be different for $\mathbf{c}$ and $\boldsymbol{\beta}$, which corresponded to non-matrix-norm-based regularizers. We hypothesized that the proposed new formulation (4.3) would produce better regression performance than existing models on this data category.

To examine how the number of tasks influenced the performance of different methods, we varied the number of tasks from 1, 5, 10, 50, and 100 to 1000. For each task, we created 200 examples, each represented by 100 features. We also tested the different methods when increasing the number of features. In this set of experiments, we used 20 tasks and each task contained 1000 examples. Each example was represented by a number of features ranging from 200 to 1000 with a step size of 200.

Category 2 (C2). In the C2 experiments, 10% of the rows in $\mathbf{A}$ were set to non-zero and these features were shared by all tasks. We then arranged the tasks to follow six different sparse structures (the staircases) as shown in Figure 4.1b(top), where we once



(a) Synthetic data C1 (b) Synthetic data C2 (c) Synthetic data C3

Figure 4.1: The true parameter matrix versus the parameter matrices constructed by the various methods on synthetic data. The figure shows the results when 200 examples and 100 features were created for 20 tasks each. Darker color indicates larger values in magnitude.

again transpose $\mathbf{A}$. Each of the remaining features except the 10% common features was used by a comparatively small proportion of the tasks. Consecutive tasks were grouped such that the neighboring groups of tasks shared 7% of the features besides the 10% common features, whereas the non-neighboring groups of tasks did not share any features. Therefore, no feature could be excluded from all tasks, but a majority of individual features (90%) was only useful for few tasks (i.e., the useful features for one task were sparse). In this case, the non-sparsity-inducing norm was suitable for regularizing $\mathbf{c}$ and sparsity-inducing norm was more suitable for regularizing $\boldsymbol{\beta}$. We hypothesized that the new formulation (4.4) would produce better regression performance than the other models on this dataset.

We created 20, 50, 100, 500 and 1000 tasks, respectively, to test the algorithms' sensitivity to the number of tasks. The numbers of tasks were chosen to make sure enough tasks in each of the six groups. The number of tasks in each group ranged from 3 to 170. We generated 200 examples and 100 features for each task. We also created another set of C2 datasets with the number of features changing from 200 to 1000 with a step size of 200 for 20 tasks and 1000 examples for each task.

Category 3 (D3). This category was synthesized following the model of the additive decomposition methods. It only contained one dataset where 200 examples, each represented

Table 4.3: Comparison of the test R^2 values obtained by the different MTFL methods on synthetic datasets using different partition ratios for training data (where standard deviation 0 means that it is less than 0.01).

Dataset		STL_lasso	STL_ridge	DMTL	rMTFL	MMTFL(2,2)	MMTFL(1,1)	MMTFL(2,1)	MMTFL(1,2)
C1	25%	0.32±0.02	0.45±0.01	0.60±0.02	0.58±0.02	0.64±0.02	0.54±0.03	**0.73±0.02**	0.42±0.04
	33%	0.54±0.03	0.62±0.02	0.73±0.01	0.61±0.02	0.79±0.02	0.76±0.01	**0.86±0.01**	0.65±0.03
	50%	0.70±0.02	0.86±0.01	0.75±0.01	0.66±0.01	0.86±0.01	0.88±0.01	**0.90±0.01**	0.84±0.01
C2	25%	0.42±0.03	0.33±0.01	0.36±0.01	0.46±0.01	0.45±0.01	0.35±0.05	0.46±0.02	**0.49±0.02**
	33%	0.77±0.02	0.63±0.01	0.42±0.02	0.63±0.03	0.69±0.02	0.75±0.01	0.67±0.03	**0.83±0.02**
	50%	0.95±0.01	0.89±0.01	0.81±0.01	0.83±0.01	0.91±0	0.95±0	0.92±0.01	**0.97±0**
C3	25%	0.28±0.03	0.43±0.03	0.50±0.04	**0.55±0.03**	0.54±0.03	0.31±0.02	0.43±0.02	0.34±0.02
	33%	0.47±0.01	0.56±0.02	0.60±0.02	**0.65±0.02**	0.64±0.02	0.45±0.04	0.60±0.04	0.47±0.03
	50%	0.77±0.01	0.78±0.01	0.76±0.02	**0.83±0.01**	0.81±0.02	0.75±0.01	0.79±0.02	0.76±0.01

by 100 features, were generated for each of 20 tasks. The parameter matrix $\mathbf{A} = \mathbf{P} + \mathbf{Q}$ where 80 rows in $\mathbf{P}$ and 16 columns in $\mathbf{Q}$ were set to 0. The component $\mathbf{P}$ was used to indicate the subset of relevant features across all tasks, and the component $\mathbf{Q}$ was used to tell that there were outlier tasks that did not share features with other tasks. Given this simulation process, this dataset would be in favor of the rMTFL model proposed in [29]. The designed model parameter matrix was shown in Figure 4.1c(top).

4.5.1.2 Performance on synthetic datasets

We first compared the regression performance of the different methods on the three categories of datasets. Table 4.3 shows the averaged R^2 values together with standard deviations for each method and each trial setting. The best results are shown in bold fonts. The results in Table 4.3 were obtained on synthetic datasets that had 20 tasks with 200 examples and 100 features for each task. We reported the test R^2 obtained on each dataset when 25%, 33% and 50% of the data were used in training. From Table 4.3, we observe that the proposed formulation (4.3)(MMTFL(2,1)) consistently outperformed other models on *C1* datasets, whereas the proposed model (4.4)(MMTFL(1,2)) consistently outperformed on *C2* datasets. The results confirmed our hypotheses that the two proposed models could be more suitable for learning the type of sharing structures in *C1* and *C2*. As anticipated, rMTFL model constantly outperformed other models on the *C3* dataset. Among the multiplicative

MTFL methods, MMTFL(2,2) achieved similar performance to that of rMTFL (off only by $0.01 \sim 0.02$ for average R^2).

In order to elucidate the different shrinkage effects of the different decomposition strategies and regularizers, we compared the true parameter matrix with the constructed parameter matrices by the six MTFL methods used in our experiments in Figure 4.1. From the results on the *C1* and *C2* datasets, we observe that only MMTFL(2,1), MMTFL(1,2) and MMTFL(1,1) produced reasonably sparse structures. The two additive decomposition methods could not yield a sufficient level of sparsity in the models. Although the unused features did receive smaller weights in general, they were not completely excluded. To evaluate the accuracy of feature selection, we quantitatively measured the discrepancy between the estimated models and the true model by computing the mean squared error (MSE) $trace((\mathbf{A}-\mathbf{A}_{est})^\top(\mathbf{A}-\mathbf{A}_{est}))$ where $\mathbf{A}_{est}$ was the matrix estimated by a method. We compared MSE values of individual mothods.

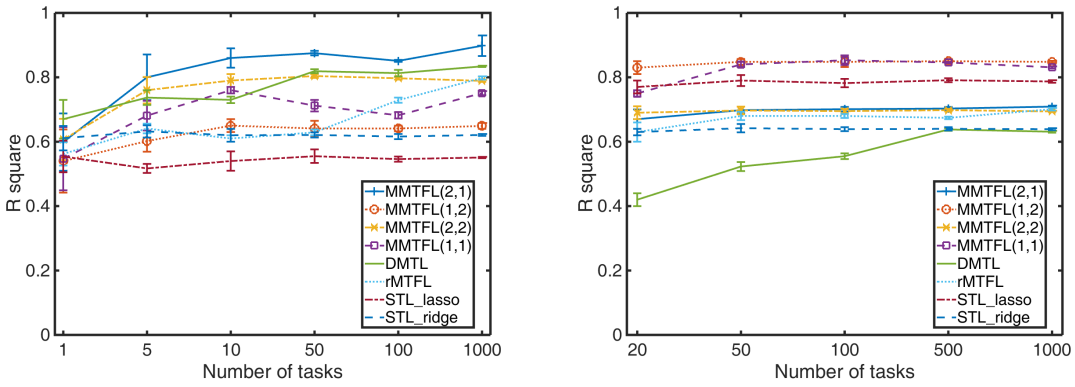
On the *C1* data, MMTFL(2,1) learned better combination weights (darker areas) for the relevant features than MMTFL(1,1). MMTFL(1,1) appeared to be unnecessarily too sparse because the useful features received much smaller weights than needed (lighter than the true model). The smallest MSE was achieved by MMTFL(2,1) with a value of 0.1, and the second best model, MMTFL(1,1), had MSE = 0.2 whereas the rMTFL model had the largest MSE = 0.25.

On the *C2* data, MMTFL(1,2) learned a model that was most comparable to the true model. Both MMTFL(1,2) and MMTFL(1,1) eliminated well the irrelevant features. However, if we compared the two rows corresponding to these two models in Figure 4.1, we could see that MMTFL(1,1) broke the staircases in several places (e.g., towards the lower right and the up left corners) by excluding more features than necessary. Note that the feature sharing patterns (particularly in synthetic data *C2*) may not be revealed by the recent methods on

clustered multi-task learning that cluster tasks into groups [51, 42, 119] because no cluster structure is present in Figure 4.1b. Rather, the sharing pattern in Figure 4.1b is actually in between the consecutive groups of tasks. MMTFL(1,2) had the smallest MSE = 0.05, which was smaller than that of the second best model MMTFL(1,1) by 0.025. DMTL received the largest MSE = 0.19.

Figure 4.1c shows the results on the *C3* dataset. MMTFL(2,1), MMTFL(1,2), and MMTFL(1,1) imposed excessive sparsity on the parameter matrix, which removed some useful features. The other three models, DMTL, rMTFL and MMTFL(2,2), produced similar parameter matrices, but rMTFL was originally designed to detect outlier tasks and thus was more favorable for this dataset. The rMTFL model obtained the smallest MSE (0.03), and MMTFL(2,2) had a similar performance (MSE=0.04), which was the same as that of DMTL. On this dataset, MMTFL(1,1) got the largest MSE (0.09). These results bring out an interesting observation that for the MTL scenarios that have outlier tasks but relevant tasks share the same set of features, MMTFL(2,2) (which corresponds to the very early joint regularized method using the $\ell_{1,2}$ matrix norm) is most suitable among the multiplicative MTFL methods.

Figure 4.2 compares the performance of different methods when we vary the number of tasks in the *C1* and *C2* categories. On each dataset, we used 33% of the data in training with 15 trials, and reported here the average R^2 values and standard deviation bars. From Figure 4.2, MMTFL(2,1) constantly performed the best among all methods on the *C1* datasets (but not in the single task learning) whereas MMTFL(1,2) outperformed the other models on the *C2* datasets. We also observed that on *C2* data, MMTFL(1,1) obtained very similar performance to that of MMTFL(1,2) after the number of tasks reached 50. On this data category, almost all methods reached a stable level of accuracy after the number of tasks reached 50 except DMTL. DMTL continued to gain knowledge from more relevant tasks

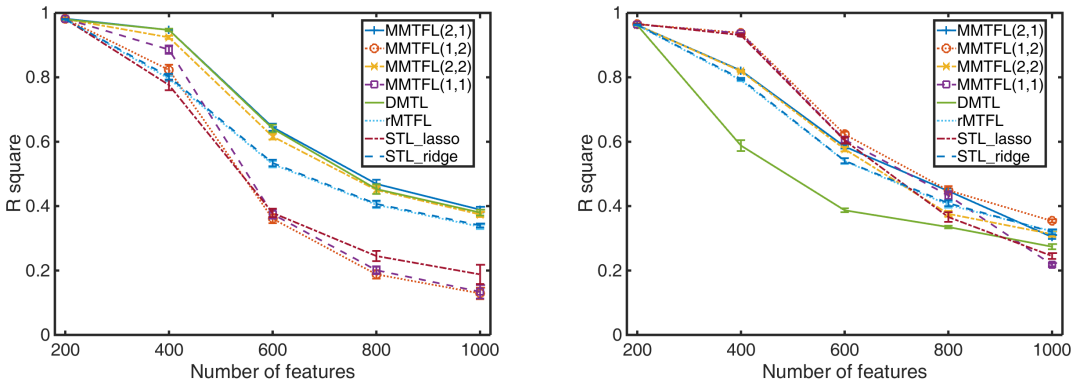


(a) On *C1* data (b) On *C2* data

Figure 4.2: The regression performance of different models on synthetic data *C1* and *C2* when the number of tasks is varied.

until it reached 500 tasks but it produced the lowest R^2 values among all methods. Overall, the results indicate that with the fixed dimension and sample size, when the number of tasks reaches a certain level, the transferable knowledge learned from the tasks can be saturated for a specific feature sharing structure. On *C1* data, we observe that the performance was not always monotonically improved or non-degraded (for all methods) when more tasks were included, which may indicate that when an unnecessarily large number of tasks was used, it could add more uncertainty to the learning process.

Figure 4.3 compares the performance of different methods when we vary the number of features in the *C1* and *C2* categories. Obviously, when the problem dimension was higher, the learning problem became more difficult (especially when the number of tasks and sample size remained the same). All methods dropped their performance substantially with increasing numbers of features although MMTFL(2,1) and MMTFL(1,2) still outperformed other methods, respectively, on the *C1* and *C2* datasets. This figure also shows that MMTFL(1,1) performed well on the *C2* datasets but much worse on the *C1* datasets. DMTL produced good performance, close to that of MMTFL(2,1), on the *C1* datasets.



(a) On *C1* data (b) On *C2* data

Figure 4.3: The regression performance of different models on synthetic data *C1* and *C2* when the number of features is varied.

4.5.2 Experiments with Benchmark Data

Extensive empirical studies were conducted on benchmark datasets where we tested the proposed multiplicative MTFL algorithms on ten real-world datasets. Among these datasets, three were for regression experiments and all others were for classification experiments. Characteristics of these datasets are summarized in the next section.

4.5.2.1 Benchmark Datasets

Sarcos [5]: Sarcos data were collected for a robotics problem of learning the inverse dynamics of a 7 degrees-of-freedom SARCOS anthropomorphic robot arm. Each observation has 21 features corresponding to 7 joint positions and their velocities and accelerations. We needed to map from the 21-dimensional input space to 7 joint torques, which corresponded to 7 tasks. For each task, we randomly selected 2000 cases for training and the remaining 5291 cases for test. Readers can consult with http://www.gaussianprocess.org/ gpml/data/ for more details.

CollegeDrinking [11]: The college drinking data were collected in order to identify alcohol

use patterns of college students and the risk factors associated with the binge drinking. The dataset contained daily responses from 100 college students to a survey questionnaire measuring various daily measures, such as drinking expectation, negative affects, and level of stress, every day in a 30 day period. The goal was to predict the amount of nighttime drinks based on 51 daily measures for each student, corresponding to 100 regression tasks. Because there were only 30 records for each person, we used 66%, 75% and 80% of the records to form the training set, and the rest for test.

QSAR [62]: The quantitative structure-activity relationship (QSAR) methods are commonly used to predict biological activities of chemical compounds in the field of drug discovery. The datasets we used were collected from three different types of drug activities, including binding to cannabinoid receptor 1 (CB1), inhibition of dipeptidyl peptidase 4 (DPP4) and time dependent 3A4 inhibitions (TDI). For each activity, there were 200 molecule examples represented by 2618 features. Three regression models were constructed to simultaneously predict the targets $-log(\text{IC50})$) of the CB1, DPP4 and TDI effectiveness based on the molecular features.

C.M.S.C. [61]: The Climate Model Simulation Crashes (C.M.S.C.) dataset contained records of simulated crashes encountered during climate model uncertainty quantification ensembles. The dataset comprised 3 tasks. There were 180 examples for each task. Each example was represented by an 18-dimensional feature vector. Each task is formed by a binary classification problem, which was to predict simulation outcomes (either fail or succeed) from the input parameter values for a climate model.

Landmine [105]: The original Landmine data contained 29 datasets where sets 1-15 corresponded to the geographical regions that were highly foliated and sets 16-29 corresponded to the regions with bare earth or desert. Each dataset could be used to build a binary classifier. We used the datasets 1-10 and 16-25 to form 20 tasks where each example was

represented by 9 features extracted from radar images. The number of examples varied between individual tasks ranging from 445 to 690.

Alphadigits [64]: This dataset was composed of binary 20 $\times$ 16 images of the 10 digits and capital letters. We used all the images of digits to form 10 binary classification tasks. For each digit, there were 39 images in this dataset. We labeled the images of a single digit as positive examples, and randomly selected other 39 images from other digits and labeled them as negative examples. All the pixels were concatenated to form a 320-dimensional feature vector for each image.

Underwatermine [59]: This dataset was originally used in the underwater mine classification problem that aimed to detect mines from non-mines based on the synthetic-aperture sonar images. The dataset consisted of 8 tasks with sample sizes ranging from 756 to 3562 for each task, and each task was a binary classification problem. Each example was represented by 13 features.

Animal recognition [51]: This dataset consisted of images from 20 animal classes. Each image was originally represented by 2000 features extracted using the bag of word descriptors from the Scale-invariant Feature Transform (SIFT), and then the dimensionality was reduced to 202 by a principal component analysis, retaining 95% of the data variance. For each animal class, there were 100 images. We formed 20 binary classification tasks where for each task, 100 positive examples were from a specific animal class and 100 negative examples were randomly sampled from other classes.

HWMA_base and HWMA_peak [76]: The heart wall motion abnormality (HWMA) detection dataset was used to analyze and predict if a heart had abnormal motion based on the image features extracted from stress test echocardiographs of 220 patients. The images were taken at the base dose and also peak dose of stress contrast. The wall of left ventricle was medically segmented into 16 segments, and 25 features were extracted from each seg-

ment. Every segment of every heart case was annotated by radiologists in terms of normal or abnormal motions. Thus, there were 16 binary classification tasks, each corresponding to one of the 16 heart segments, and each task comprised 220 examples.

4.5.2.2 Performance on real world datasets

Three real-world datasets, the Sarcos, CollegeDrinking and QSAR, were used in regression experiments. The performance of the different methods is summarized in Table 4.4 depicting the R^2 values averaged over the 15 re-partitions in each trial. MMTFL(2,1) achieved the best R^2 values on the Sarcos dataset (in all of the 3 trials) and the CollegeDrinking dataset (in 2 of the 3 trials). The Sarcos data appeared to be in favor of denser models given MMTFL(2,2) also performed reasonably well on this dataset. MMTFL(1,2) models achieved the best performance on the QSAR dataset consistently across all the 3 trial settings. On this dataset, it was obvious that MMTFL(1,2) was more suitable, which indicated that most of the 2618 features were useful for some tasks, but the tasks shared few features between each other. The difference between the proposed models and the additively decomposed models ranged from 1 % to 10%, and most importantly, the trend was consistent for the proposed models to outperform on these datasets.

The other seven real-world datasets were used in classification experiments. Table 4.5 summarizes the results where the F1 scores were averaged across the 15 random splits in each trial together with standard derivations. MMTFL(2,1) models achieved consistently the best performance on the C.S.M.C. and Landmine datasets in comparison with other models. In particular, we observed both MMTFL(1,1) and the MMTFL(2,1) models produced the best F1 scores in the trial with 33% training split whereas MMTFL(2,1) outperformed all other models in the trial with other partition ratios. These two datasets may prefer across-

task sparse models, indicating that many irrelevant features may exist in the data. For the remaining five datasets used in classification experiments, MMTFL(1,2) models showed generally better performance than all other models. The difference between the best model and other MMTFL models could reach 4% to 8%.

In particular, for the Alphadigits dataset, we used the raw pixels of the hand written digits as the features to build models. Each task aimed to learn a linear model in the original pixel dimensions to distinguish a digit from other nine digits. Thus, each model, or equivalently the weight vector of the linear model, could be re-shaped back into an image. Figure 4.4 compares the constructed models for each digit by each of the six MTFL methods. In the top of Figure 4.4, we illustrate some sample images. In the middle, we show the results by the four MMTFL methods whereas at the bottom we include the constructed additively decomposed models. Clearly, the additively decomposed models were much noisier and selected many undesirable features. Among the multiplicative MTFL methods, MMTFL(1,1) models were too sparse to trace out the digits. Overall, MMTFL(1,2) models were the closest to the shapes of the different digits. If we compare MMTFL(2,1) and (1,2), MMTFL(2,1) excluded too many features from all digits. It indicated that most of the pixels were useful for predicting a digit but not many pixels were shared by multiple digits.

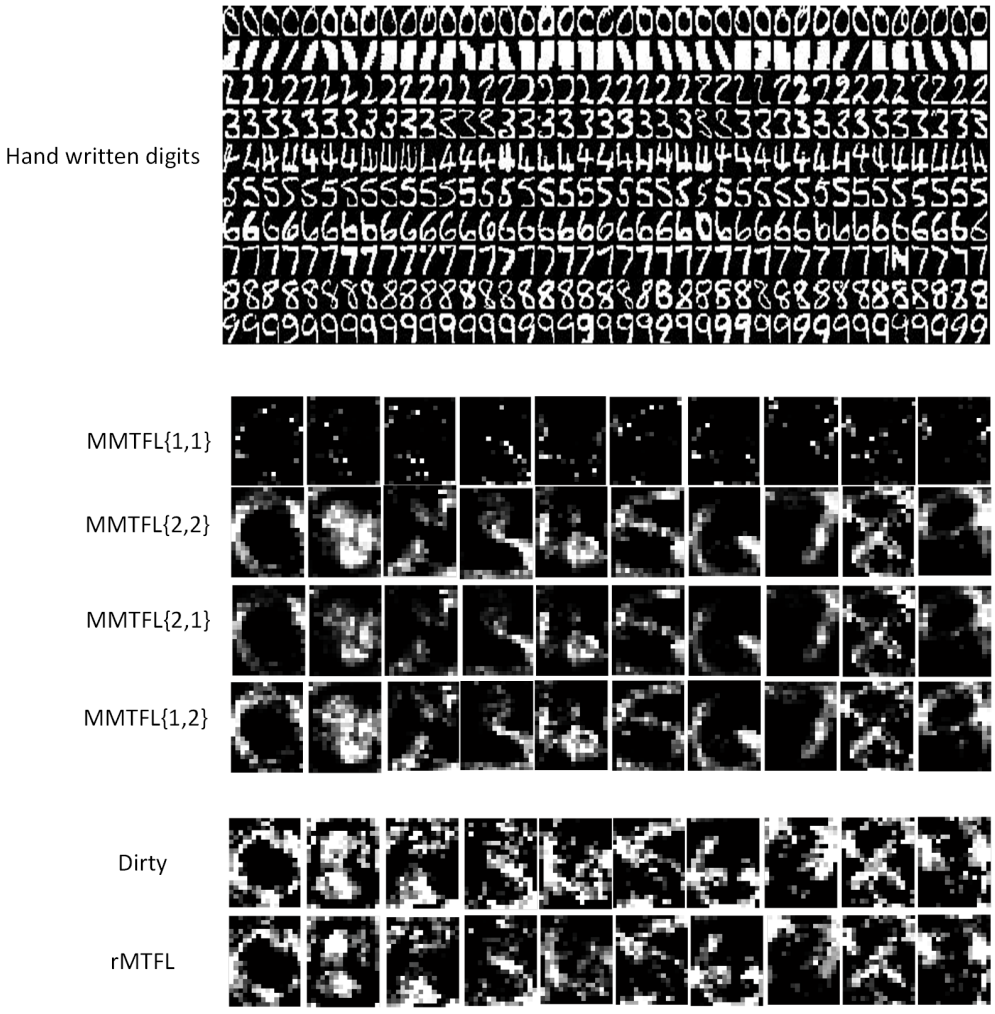


Figure 4.4: The models constructed by MTFL methods for individual digits using the pixels in hand written digit images. The models can be re-organized back into images. The pixel in the above images ranges from 0 to 1. The lighter, the closer to 1 for lucid illustration.

Table 4.4: Comparison of the R^2 values of the different MTFL methods on the benchmark regression datasets.

Dataset		STL_lasso	STL_ridge	DMTL	rMTFL	MMTFL(2,2)	MMTFL(1,1)	MMTFL(2,1)	MMTFL(1,2)
SARCOS	25%	0.75±0.02	0.78±0.02	0.86± 0	0.86±0	**0.89± 0**	0.88± 0	**0.89±0.01**	0.86±0.01
	33%	0.78±0.01	0.78±0.02	0.81±0.11	0.81±0.1	**0.90± 0**	0.89± 0	**0.90±0.01**	0.82±0.01
	50%	0.82±0.01	0.83±0.06	0.87± 0.1	0.87±0.1	0.89± 0	0.90± 0.01	**0.91±0.01**	0.86±0.01
CollegeDrinking	66%	0.15±0.05	0.09±0.02	0.11± 0.07	0.22±0.02	**0.23±0.04**	0.15± 0.02	0.21±0.01	0.19±0.04
	75%	0.18±0.05	0.15±0.02	0.18± 0.08	0.18±0.05	0.14±0.05	0.25± 0.08	**0.27±0.05**	0.21±0.08
	80%	0.11±0.06	0.09±0.04	0.19± 0.06	0.16±0.06	0.19±0.05	0.15± 0.04	**0.21±0.04**	0.15±0.04
QSAR	25%	0.39±0.03	0.36±0.01	0.39±0.01	0.39±0.01	0.34±0.02	0.38±0.03	0.38±0.03	**0.40±0.03**
	33%	0.39±0.04	0.39±0.04	0.40±0.04	0.39±0.04	0.37±0.04	0.41±0.04	0.40±0.05	**0.43±0.03**
	50%	0.44±0.06	0.41±0.03	0.44±0.03	0.44±0.04	0.40±0.06	0.44±0.04	0.45±0.05	**0.48±0.04**

Table 4.5: Comparison of the F1 scores of the different MTFL methods on the benchmark classification datasets.

Dataset		STL_lasso	STL_ridge	DMTL	rMTFL	MMTFL(2,2)	MMTFL(1,1)	MMTFL(2,1)	MMTFL(1,2)
C.S.M.C.	25%	0.30±0.04	0.31±0.03	0.36±0	0.26±0.01	0.40±0	0.39±0.01	**0.43±0**	0.39±0
	33%	0.37±0.02	0.37±0.06	0.37±0.01	0.39±0.01	0.40±0.01	**0.45±0.01**	**0.45±0.01**	0.39±0.01
	50%	0.40±0.02	0.46±0.01	0.37±0.01	0.44±0.01	0.48±0.01	0.46±0.01	**0.49±0.01**	0.46±0
Landmine	25%	0.14±0.01	0.06±0.01	0.04±0.01	0.06±0	0.1±0.01	**0.17±0.01**	**0.17±0.01**	**0.17±0.01**
	33%	0.13±0.01	0.15±0.01	0.16±0.01	0.16±0.01	0.1±0.01	**0.18±0.01**	**0.18±0.01**	0.12±0.01
	50%	0.15±0.01	0.15±0.01	0.18±0.01	0.21±0.01	0.11±0.01	0.19±0.01	**0.23±0.01**	0.15±0.01
Alphadigits	25%	0.86±0.01	0.77±0.01	0.86±0.01	0.86±0	0.85±0.01	0.84±0.01	0.78±0.01	**0.88±0.01**
	33%	0.90±0.01	0.87±0.01	0.90±0.01	0.90±0.01	0.87±0.01	0.85±0.01	0.81±0.01	**0.91±0.01**
	50%	0.90±0.01	0.87±0.01	0.91±0.01	0.92±0.01	0.91±0.01	0.90±0.01	0.85±0.01	**0.93±0.01**
Underwatermine	25%	0.24±0.01	0.27±0.02	0.21±0.01	0.27±0	**0.29±0.01**	0.24±0.01	0.25±0.01	0.27±0.01
	33%	0.25±0.01	0.31±0.02	0.31±0.01	0.29±0.01	0.3±0.01	0.3±0.01	0.27±0.01	**0.32±0.01**
	50%	0.27±0.01	0.28±0.01	0.32±0.01	0.34±0.01	0.32±0.01	0.34±0.01	0.35±0.01	**0.37±0.01**
Animal	25%	0.63±0.01	0.63±0.01	0.65±0.01	0.65±0	**0.66±0.01**	0.64±0.01	0.63±0.01	**0.66±0.01**
	33%	0.66±0.01	0.67±0.01	0.66±0.01	0.65±0.01	0.67±0.01	0.66±0.01	0.66±0.01	**0.68±0.01**
	50%	0.67±0	0.68±0.01	0.68±0.01	0.66±0.01	**0.69±0.01**	0.67±0.01	0.68±0.01	**0.69±0.01**
HWMA_base	25%	0.35±0.01	0.37±0.01	0.42±0.01	0.4±0	**0.46±0.01**	0.36±0.01	0.44±0.01	0.45±0.01
	33%	0.38±0.01	0.34±0	0.45±0.01	0.47±0.01	046±0.01	0.44±0.01	0.47±0.01	**0.49±0.01**
	50%	0.44±0.01	0.45±0.01	0.48±0.01	0.48±0.01	0.51±0.01	0.47±0.01	0.47±0.01	**0.55±0.01**
HWMA_peak	25%	0.41±0	0.48±0.01	0.47±0.01	0.47±0	0.53±0.01	0.53±0.01	0.51±0.01	**0.54±0.01**
	33%	0.42±0.01	0.47±0.01	0.47±0.01	0.5±0.01	**0.56±0.01**	0.55±0.01	0.52±0.01	0.53±0.01
	50%	0.44±0.02	0.50±0.02	0.49±0.01	0.51±0.01	0.56±0.01	0.56±0.01	0.55±0.01	**0.58±0.01**

4.6 Conclusion

In this work of dissertation, we have studied a general framework of multiplicative multi-task feature learning. By decomposing the model parameter of each task into a product of two components: the across-task feature indicator and task-specific parameters, and applying different regularizers to the two components, we can select features for individual tasks and also search for the shared features among tasks. We have examined the theoretical properties of this framework when different regularizers are applied and found that this family of methods creates models equivalent to those of the joint regularized multi-task learning methods but with a more general form of regularization. Further, we show that this family consists of some convex and some non-convex formulations and specify the conditions to obtain convexity. An analytical formula is derived for the across-task component as related to the task-specific component, which sheds light on the different shrinkage effects in the various regularizers. An efficient algorithm is derived to solve the entire family of methods and also tested in our experiments for some chosen parameter values. Empirical results on synthetic data clearly show that there may not be a particular choice of regularizers that is universally better than other choices. We empirically show a few feature sharing patterns that are in favor of the two newly-proposed choices of regularizers. The extensive experimental results on real-world benchmark datasets also confirm the observation and demonstrate the advantages of the proposed two formulations over several existing methods.

Chapter 5

Learning Task Grouping with Multiplicative Feature Sharing Patterns in Each Group of Tasks

In multi-task learning studies, the relatedness of the tasks is the fundamental basis for MTL to be beneficial. It is thus essential to determine if the tasks invovled are indeed strongly related or if a grouping of the tasks is necessary to identify the sharing patterns in different groups. A task grouping may help a MTL method to enhance the performance of individual tasks rather than worsen it.

Several methods have been proposed for clustered MTL by enforcing tasks in one cluster to have similar parameters [105, 42, 119, 121, 36]. The similarity is measured in the entire feature space which ensures the grouped tasks share the exactly same features. However, it has been revealed that such clustering strategy may be too restrictive in practice [51, 116, 104]. In many cases, it is more beneficial to cluster tasks based on how they share and use different subsets of features. For example, in the image categorization problem, tasks can be

related because of the same subset of features generated from a region of interests. Besides, relevant tasks may also build up opposite relationships in predictive models because the common features may take opposite weights (rather than similar weights). A feature may be positively predictive of a task whereas negatively predictive of another in the same task group. Although the tasks may show a large dissimilarity in terms of model parameters, they are related in the way that they use the same subset of features and may be grouped together to benefit the learning process.

Multi-task feature learning (MTFL) captures the relatedness among the tasks typically by investigating the assumption that different tasks share a common representation in the feature space. Some of the existing MTFL algorithms exploit the low-dimensional subspace of features either by imposing a block-wise joint regularization on the model parameter matrix itself [70, 6, 114, 30], or by decomposing the parameter matrix into a summation [43, 29] or multiplication [13, 60, 101] of two components with different regularizations. Other methods such as [78, 1], project the original model parameters in order to search a latent common substructure. In MTFL, sparse feature structure may exist so that a group of tasks rely on a limited subset of features. When a feature is irrelevant across all tasks, sparsity-inducing regularizers, such as the $\ell_{1,p}$ matrix norm, can shrink the model coefficient matrix to eliminate the feature from all the predictive models.

In this work, we identify groups of tasks that have similar sparse structures in the feature space. For the tasks in a cluster, their models are constructed by decomposing the model parameter into an element-wise multiplication of two components with one component used to model feature sharing structure across tasks and the other component to estimate individual tasks' model parameters. We have experimented with different regularizers to regularize the two components. Particularly, sparsity inducing regularizer is applied to across-task component, to capture the sparse structure in features, while non-sparsity inducing regularizer is

applied to task-specific parameters, allowing the selected features to be used by all tasks in one cluster. Each cluster corresponds to a distinguished sparse structure of features, based on which the cluster indicator is jointly estimated with model parameters.

5.1 Related Work

In this section, we review the previous clustered MTL algorithms. These methods cluster tasks either by examining similarities between model parameters or by exploring the distinguished subset of features used by each cluster of tasks.

In the first direction, the early methods symmetric MTL (SMTL) and asymmetric MTL (AMTL) in [105] computed the similarity among the model parameters of tasks by placing a Dirichlet process prior on model parameters. The clustered multitask learning (CMTL) method in [42] calculated the centroids of all task parameters as well as of each cluster. Their method regularized the between-cluster and within-cluster variances of model parameters to capture cluster structure. CMTL derived a convex relaxation of their K-means based methods. The later work [120] derived a new method for the convex relaxation on clustered multitask learning (cCMTL). Different from CMTL in [42], between-cluster variance of model parameters was not considered. Multi-level task grouping (MeTaG) proposed in [36] identified the grouping structure at multiple levels rather than at one level as other methods do. Their method decomposed the model parameter into a summation of H components. With each component corresponding to one level, pairwise difference among all tasks in their model parameters was regularized with an ℓ_2 norm to learn the group structure at each level. Different from the methods introduced above, the approaches for grouping and overlap in MTL (GOMTL) [54] and MTL based on SVM (MTSVM) in [8] did not measure

the similarity between tasks. Both GOMTL and MTSVM assumed that the model parameters of each task can be generated by a linear combination of latent basis tasks. Tasks in a group lie in a low dimensional subspace. Flexible clustered multitask learning (FCMTL) [121] assumed that there exist several representative tasks among all tasks. To select the representative tasks for each individual task, a convex combination of the Euclidean distances between this task and all other tasks is evaluated. GOMTL, MTSVM and FCMTL are able to implement soft-clustering on tasks by allowing overlap on basis or representative tasks. The above methods require model parameters of all tasks within a cluster to be similar, however, they are ineffective to model the relatedness when tasks share the same subspace of features but have large differences in the weight values of the shared features.

Another way to learn task grouping is through the detection of shared features. The method of learning to group tasks based on MTFL (GMTL) [51] modeled the relatedness of tasks by learning shared features and formulated a mixed integer programming problem to jointly optimizing cluster indicator and model parameters. Both GMTL and our method exploit the feature sharing patterns by adopting MTFL techniques. However, GMTL used the block-wise joint regularization on the model parameters of the tasks directly, whereas we decompose the model parameter vector into a multiplication of two components and apply different regularizations to them. The two methods achieve different shrinkage effects on the estimated model parameters. Their difference will be discussed in more detail in the following Section 5.2.1. Unlike the GMTL and the proposed method, flexible task-clustered MTL (FlexTClus) proposed in [116] measured the pairwise differences among parameters of all tasks on each feature, to construct feature level task groups. A more recent work [104] decomposed a model parameter vector into a summation of two components, one of which was used to reflect global similarities among tasks and the other was used to perform task-feature co-clustering.

5.2 The Proposed Task Grouping Method

We introduce the proposed method in this section. Given T learning tasks in total, for each task t, $t \in \{1, \cdots, T\}$, we have a sample set $(\mathbf{X}_t \in \mathbb{R}^{\ell_t \times d}, \mathbf{y}_t \in \mathbb{R}^{\ell_t})$ that has ℓ_t examples, where the i-th row corresponds to the i-th example $\mathbf{x}_i^t$ of task t, $i \in \{1, \cdots, \ell_t\}$, and each example is represented by a vector of d features. The vector $\mathbf{y}_t$ contains y_i^t, the label of the i-th example for task t. We adopt functions of the linear form $y_i^t = \boldsymbol{\alpha}_t^\top (\mathbf{x}_i^t)^\top$ where $\boldsymbol{\alpha}_t \in \mathbb{R}^d$, which corresponds to computing $\mathbf{X}_t \boldsymbol{\alpha}_t$ on the training data. We define the parameter matrix or weight matrix $\mathbf{A} = [\boldsymbol{\alpha}_1, \cdots, \boldsymbol{\alpha}_T]$ and denote the rows of $\mathbf{A}$ by $\boldsymbol{\alpha}^j$, $j \in \{1, \cdots, d\}$. The following multi-task learning formulation is studied:

$$\min_{\boldsymbol{\alpha}_{t:t=1,\cdots,T}} \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \Omega(\boldsymbol{\alpha}_t) \tag{5.1}$$

where $L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t)$ is the empirical loss that can take any suitable form, such as the least square loss for regression tasks or the logistic regression loss for classification tasks.

The second term $\Omega(\boldsymbol{\alpha}_t)$ regularizes the model parameters for T tasks. A common strategy is to impose a blockwise joint regularization [65, 115] on the matrix $\mathbf{A}$ to shrink the effects of features across the tasks. These methods employ the $\ell_{1,p}$ matrix norm [55, 58, 70, 114, 122], which encourages sparsity on the rows of the matrix. When each row of $\mathbf{A}$ corresponds to a feature and a column represents an individual task, the $\ell_{1,p}$ regularizer intends to rule out the irrelevant features across tasks by shrinking the corresponding rows in $\mathbf{A}$ to zeros.

5.2.1 Multiplicative Multi-task Feature Learning

For the existing MTFL methods based on joint regularization, a major limitation is that it either selects a feature as relevant to all tasks or excludes it from all models, which is

very restrictive in practice where tasks may share some features but may also have their own specific features that are not relevant to other tasks. To overcome this limitation, an effective strategy is to decompose the model parameter into a multiplication of two components with regularization imposed on each component, which is referred as Multiplicative Multi-task Feature Learning in our recent work [101]. Chapter 4 of this dissertation has introduced our studies on multiplicative MTFL.

In multiplicative MTFL, model parameters are decomposed as $\boldsymbol{\alpha}_t = \text{diag}(\mathbf{c})\boldsymbol{\beta}_t$, where $\text{diag}(\mathbf{c})$ is a diagonal matrix with its diagonal elements composing $\mathbf{c}$. The vector $\mathbf{c}$ is used across all tasks, indicating if a feature is useful for any tasks. The $\boldsymbol{\beta}_t$ vector is only used for task t, containing the task-specific model parameters. Let j index the entries in these vectors. We have $\alpha_j^t = c_j \beta_j^t$. Typically $\mathbf{c}$ comprises binary entries that are equal to 0 or 1, but the integer constraint is often relaxed to require just non-negativity (i.e., $\mathbf{c} \geq 0$). Multiplicative multi-task feature learning (MMTFL) methods minimize a regularized loss function as follows for the best $\mathbf{c}$ and $\boldsymbol{\beta}_{t:t=1,\cdots,T}$,

$$\min_{\boldsymbol{\beta}_t, \mathbf{c} \geq 0} \quad \sum_{t=1}^{T} L(\mathbf{c}, \boldsymbol{\beta}_t, \mathbf{X}_t, \mathbf{y}_t) + \gamma_1 \sum_{t=1}^{T} ||\boldsymbol{\beta}_t||_p^p + \gamma_2 ||\mathbf{c}||_k^k \tag{5.2}$$

Regularizers are imposed on the two decomposed components, where p and k are positive integers. The tuning parameters γ_1, γ_2 are used to balance the empirical loss and regularizers. At optimality, if $c_j = 0$, the j-th variable is removed for all tasks, and the corresponding row vector $\boldsymbol{\alpha}^j = \mathbf{0}$; otherwise the j-th variable is selected for use in at least one of the $\boldsymbol{\alpha}$'s. Then, a specific $\boldsymbol{\beta}_t$ rules out the j-th variable from task t if $\beta_j^t = 0$.

The method proposed in [13] has both $p = k = 2$, which does not impose strong sparsity on the model parameters, while the method used in [60] has $p = k = 1$, which is suitable for learning from tasks with persistently sparse models. A more recent work of ours [101]

examines the general framework of the multiplicative decomposition Eq.(5.2) and proposes two new formulations requiring different sparse effects on variable selection with ($p = 2$, $k = 1$) and ($p = 1$, $k = 2$). The first new formulation is favorable to learning problems where irrelevant features exist across tasks, however, a lot of features are shared by individual tasks. In other words, the features used by each task are not sparse with respect to the selected features indicated by $\mathbf{c}$. The second formulation may help those learning tasks when the union of the features relevant to any given tasks includes many or even all features, but different tasks may share a limited number of features, i.e., each individual task uses a small amount of features selected by $\mathbf{c}$.

5.2.2 Learning Task Grouping

We assume that the tasks can be clustered into G groups, $G \leq T$ and each group of tasks presents a multiplicative feature sharing pattern. Let $\mathbf{H} \in \mathbb{R}^{G \times T}$ denote the indicator matrix, so we have $\mathbf{H} = [\mathbf{h}_1, \cdots, \mathbf{h}_T]$ and $\mathbf{H} = [(\mathbf{h}^1)^\top, \cdots, (\mathbf{h}^G)^\top]^\top$. All the elements of $\mathbf{H}$ are binary (either 0 or 1), such that $h_{gt:g=\{1,\cdots,G\},t=\{1,\cdots,T\}} := 1$ indicates that the t-th task belongs to the g-th cluster. To ensure that each task belongs to only one cluster, we require $\mathbf{H}\mathbf{H}^\top$ to be a diagonal matrix, i.e., $\mathbf{H}\mathbf{H}^\top = diag([T_1, \cdots, T_G])$, where $T_{g:g=\{1,\cdots,G\}}$ contains the number of cluster members for the g-th cluster. Let $\mathbf{S}_g$, $g = 1, \cdots, G$, contains the indices of the tasks in the g-th cluster such that $\mathbf{S}_g = \{t | h_{gt} = 1\}$. Let $\mathbf{C} \in \mathbb{R}^{d \times G}$ denote the feature selection matrix where each column is an indicator vector for a task group indicating which of the d features are used in the group. The model parameter matrix for T tasks is written as $\mathbf{A} = (\mathbf{C}\mathbf{H}) \circ \mathbf{B}$, where the symbol $\circ$ means an elementwise product of two matrices and we define $\mathbf{B} = [\boldsymbol{\beta}_1, \cdots, \boldsymbol{\beta}_T] = [(\boldsymbol{\beta}^1)^T, \cdots, (\boldsymbol{\beta}^d)^T]^\top$, such that the final model parameters for

the t-th task is written as

$$\boldsymbol{\alpha}_t = \text{diag}(\mathbf{C}\mathbf{h}_t)\boldsymbol{\beta}_t \tag{5.3}$$

Similar to the equation (5.1) but we substitute $\boldsymbol{\alpha}_t$ using equation (5.3) and impose regularizers to $\boldsymbol{\beta}_t$ and $\mathbf{C}$ respectively, we propose the formulation for clustered MMTFL as follows:

$$\begin{aligned} \min_{\boldsymbol{\beta}_t, \mathbf{C}, \mathbf{h}_t} \quad & \sum_{t=1}^{T} L(\mathbf{C}, \boldsymbol{\beta}_t, \mathbf{h}_t, \mathbf{X}_t, \mathbf{y}_t) + \gamma_1 \sum_{t=1}^{T} ||\boldsymbol{\beta}_t||_2^2 + \gamma_2 ||\mathbf{C}||_1 \\ s.t. \quad & \mathbf{C} \succeq 0, \\ & \mathbf{h}_t \in \{0,1\}^G, \quad ||\mathbf{h}_t||_2 = 1, \quad t = 1, \cdots, T, \end{aligned} \tag{5.4}$$

The regularizers will impose the ℓ_1 norm regularization on each column of $\mathbf{C}$, denoted by $\mathbf{c}_g$. For the tasks within the g-th cluster, the indicator vector $\mathbf{c}_g$ with sparsity-inducing regularizer will rule out the irrelevant features by shrinking the corresponding model parameters of the tasks in this cluster to zeros. It is expected that the identified tasks in each group should share the features selected by the column of $\mathbf{C}$ corresponding to the group. Hence, a non-sparsity-inducing regularizer can be appropriate for $\boldsymbol{\beta}_{t:\{t=1,\cdots,T_g\}}$. Although the multiplicative MTFL, as discussed earlier, can take a broad spectrum of regularizers, we focus here on the specific regularizer with ℓ_2 norm on $\boldsymbol{\beta}_t$ and ℓ_1 norm on $\mathbf{c}_g$. Suppose that the tasks from different clusters have small intersection of selected features, distinguished $\mathbf{c}_g$ is needed to reflect the particular sparse structure of the tasks in the g-th cluster. The indicator matrix $\mathbf{H}$ is jointly learned by solving Problem (5.4) that assigns the most favorable $\mathbf{c}_g$ to task $\boldsymbol{\beta}_t$ such that the empirical loss can be minimized.

Previous clustered MTL methods [42, 119] required the model weights $\boldsymbol{\alpha}_{t:t=\{1,\cdots,T_g\}}$ within a cluster to be similar. In practice, however, there exist many scenarios that tasks share a large number of features but their weights on these features are dissimilar. Our approach

is able to group such tasks, besides those that show similarities in parameter values, into a cluster to jointly learn their model parameters by transferring knowledge that these tasks depend on the same set of features. Hence, our approach can learn effectively for a much larger family of MTL problems. We further extend our early analysis on the multiplicative MTFL method to prove a similar result on the grouped tasks. In other words, minimizing Eq.(5.4) is equivalent to jointly regularizing $\boldsymbol{\alpha}_t$'s with a non-convex regularizer when the tasks are grouped. The following theorem characterizes this result.

Theorem 5.1. *Let* $\boldsymbol{\alpha}_t = diag(\mathbf{C}\mathbf{h}_t)\boldsymbol{\beta}_t$. *We obtain that*

(I). Solving Problem (5.4) is equivalent to solving the following problem

$$\min_{\boldsymbol{\alpha}_t, \mathbf{h}^g} \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \lambda \sum_{j=1}^{d} \sum_{g=1}^{G} \sqrt[3]{||diag(\mathbf{h}^g)(\boldsymbol{\alpha}^j)^\top||_2^2}, \tag{5.5}$$

when $\lambda = 2\gamma_1^{\frac{1}{3}}\gamma_2^{\frac{2}{3}}$.

(II). Let $\hat{\boldsymbol{\beta}}_t$ *and* $\hat{\mathbf{h}}_t$ *optimize Problem (5.4) and* c_{jg} *be the (j,g)-th element of* $\mathbf{C}$. *The optimal solution* $\hat{\mathbf{C}}$ *of Problem (5.4) satisfy that*

$$\hat{c}_{jg} = \gamma_1\gamma_2^{-1}||diag(\hat{\mathbf{h}^g})(\hat{\boldsymbol{\beta}^j})^\top||_2^2, \quad j = \{1, \cdots, d\}, \quad g = \{1, \cdots, G\} \tag{5.6}$$

Proof. (I). The result can be directly derived from Theorem 1 from [101], where all the tasks can be seen as belonging to one cluster. Suppose that $(\hat{\mathbf{C}}, \hat{\boldsymbol{\beta}}_t, \hat{\mathbf{h}}_t)$ are optimal solutions to problem (5.4) and $\hat{\boldsymbol{S}}_g$ contains the indices of the tasks belonging to each cluster when problem (5.4) is optimized. We obtain that the objective of problem (5.4) becomes

$$\sum_{g=1}^{G} \sum_{t \in \hat{\boldsymbol{S}}_g} \Big(L(\hat{\mathbf{c}}_g, \hat{\boldsymbol{\beta}}_t, \mathbf{X}_t, \mathbf{y}_t) + \gamma_1||\boldsymbol{\beta}_t||_2^2 + \gamma_2||\mathbf{c}_g||_1 \Big) \tag{5.7}$$

where the objective of problem (5.4) is partitioned according to clusters of tasks. For each cluster g, we have $(\hat{\mathbf{c}}_g, \hat{\boldsymbol{\beta}}_t)$ optimizes the following problem:

$$\min_{\mathbf{c}_g, \boldsymbol{\beta}_t, t \in \hat{\mathbf{S}}_g} \sum_{t \in \hat{\mathbf{S}}_g} L(\mathbf{c}_g, \boldsymbol{\beta}_t, \mathbf{X}_t, \mathbf{y}_t) + \gamma_1 \sum_{t \in \hat{\mathbf{S}}_g} ||\boldsymbol{\beta}_t||_2^2 + \gamma_2 ||\mathbf{c}_g||_1 \tag{5.8}$$

According to Theorem 1 in [101], we have that $\hat{\boldsymbol{\alpha}}_t = \text{diag}(\hat{\mathbf{c}}_g)\hat{\boldsymbol{\beta}}_t$ optimizes the problem

$$\min_{\boldsymbol{\alpha}_t, t \in \hat{\mathbf{S}}_g} \sum_{t \in \hat{\mathbf{S}}_g} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \lambda \sum_{j=1}^{d} \sqrt[3]{||\text{diag}(\mathbf{h}^g)(\boldsymbol{\alpha}^j)^\top||_2^2} \tag{5.9}$$

where $\lambda = 2\gamma_1^{\frac{1}{3}}\gamma_2^{\frac{2}{3}}$. Within each cluster of tasks, the above conclusion holds. Hence, the result (I) in theorem 5.1 holds.

(II). Let $\hat{\boldsymbol{\beta}}_t$, $\hat{\mathbf{h}_t}$, and $\hat{\mathbf{C}}$ be the optimal solution of Problem (5.4). By substituting $\hat{\boldsymbol{\beta}}_t$ with $\hat{\boldsymbol{\beta}}_t = (\text{diag}(\hat{\mathbf{C}}\hat{\mathbf{h}_t}))^{-1}\hat{\boldsymbol{\alpha}}_t$ in Problem (5.4), the objective function can be transformed into

$$\boldsymbol{J}(\hat{\boldsymbol{\alpha}}_t, \hat{\mathbf{C}}, \hat{\mathbf{h}^g}) = \sum_{t=1}^{T} L(\hat{\boldsymbol{\alpha}}_t, \mathbf{X}_t, \mathbf{y}_t) + \sum_{g=1}^{G}\sum_{j=1}^{d} \left(\gamma_1 \frac{||\text{diag}(\hat{\mathbf{h}^g})(\hat{\boldsymbol{\alpha}^j})^\top||_2^2}{\hat{c_{jg}}^2} + \gamma_2 \hat{c_{jg}} \right). \tag{5.10}$$

Applying Cauchy-Schwarz inequality to the regularizers in equation (5.10), we obtain

$$\boldsymbol{J}(\hat{\boldsymbol{\alpha}}_t, \hat{\mathbf{C}}, \hat{\mathbf{h}^g}) \geq \sum_{t=1}^{T} L(\hat{\boldsymbol{\alpha}}_t, \mathbf{X}_t, \mathbf{y}_t) + \sum_{g=1}^{G}\sum_{j=1}^{d} \left(2\sqrt{\gamma_1\gamma_2 ||\text{diag}(\hat{\mathbf{h}^g})(\hat{\boldsymbol{\alpha}^j})^\top||_2^2 (\hat{c_{jg}})^{-1}} \right). \tag{5.11}$$

The equality holds only when

$$\hat{c_{jg}} = \sqrt[3]{\gamma_1\gamma_2^{-1} ||\text{diag}(\hat{\mathbf{h}^g})(\hat{\boldsymbol{\alpha}^j})^\top||_2^2}. \tag{5.12}$$

Substituting $\hat{\mathbf{C}}$ into Eq. (5.11) yields a function in terms of $\hat{\boldsymbol{\alpha}}_t$ and $\hat{\mathbf{h}}_t$ which is fixed if $\hat{\boldsymbol{\alpha}}_t$ and $\hat{\mathbf{h}}_t$ are fixed. Suppose that $\hat{c}_{jg}$ does not take the formula Eq. (5.12), then a strict

inequality holds in Eq. (5.11). Then, let another solution $\mathbf{C}^*$ be the one whose entries take the formula (5.12). Then $\boldsymbol{J}(\hat{\boldsymbol{\alpha}}_t, \hat{\mathbf{C}}, \hat{\mathbf{h}^g}) > \boldsymbol{J}(\hat{\boldsymbol{\alpha}}_t, \mathbf{C}^*, \hat{\mathbf{h}^g})$, which contradicts to the optimality of $(\hat{\boldsymbol{\beta}}_t, \hat{\mathbf{h}_t}, \hat{\mathbf{C}})$. Therefore, $\hat{\mathbf{C}}$ satisfies Eq. (5.12). Then, substituting $\hat{\boldsymbol{\alpha}^j}$ with $\hat{\boldsymbol{\alpha}^j} = \hat{c_{jg}}\hat{\boldsymbol{\beta}^j}$ into Eq. (5.12) yields the result.

□

5.3 Optimization Algorithms

Problem (5.4) is convex with respect to the parameters, $\boldsymbol{\beta}_{t:t=\{1,\cdots,T\}}$, $\mathbf{h}_{t:t=\{1,\cdots,T\}}$ and $\mathbf{C}$ individually, but is not a jointly convex problem. We adopt the blockwise coordinate descent (BCD) algorithm to solve problem (5.4), which has been used in previous works [119, 51, 54, 13, 60, 101] for efficiently solving non-convex optimization problems. The algorithm optimizes the following two sub-problems iteratively:

(1). We solve for $\mathbf{C}$ using an analytic solution according to the following theorem.

Theorem 5.2. *Let* $\boldsymbol{\alpha}_t = diag(\mathbf{C}\mathbf{h}_t)\boldsymbol{\beta}_t$, $t = 1, \cdots, T$. *For any given values of* $\boldsymbol{\alpha}_t$ *and* $\mathbf{h}_t$, *Problem (5.13), formulated as the following:*

$$\min_{\boldsymbol{\alpha}_t, \mathbf{h}^g, \mathbf{C} \succeq 0} \quad \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \sum_{g=1}^{G} \sum_{j=1}^{d} \left(\gamma_1 \frac{||diag(\mathbf{h}^g)(\boldsymbol{\alpha}^j)^\top||_2^2}{c_{jg}^2} + \gamma_2 c_{jg} \right), \tag{5.13}$$

where $\mathbf{h}^g$ *and* $\mathbf{h}^t$ *denote the g-th row and t-th column of* $\mathbf{H}$, *is optimized with respect to* $\mathbf{C}$ *if* $\mathbf{C}$ *takes the following form:*

$$c_{jg} = \sqrt[3]{\gamma_1 \gamma_2^{-1} ||diag(\mathbf{h}^g)(\boldsymbol{\alpha}^j)^\top||_2^2}. \tag{5.14}$$

Proof. The empirical loss terms in the objective function of Problem (5.13) are only asso-

ciated with $\boldsymbol{\alpha}_t$. Given that the values of $\boldsymbol{\alpha}_t$ and $\mathbf{h}^g$ are fixed, applying Cauchy-Schwarz inequality to the regularizers yields the result. □

Suppose that $\mathbf{C}^{(s)}$, $\mathbf{h}_t^{(s)}$ and $\boldsymbol{\beta}_t^{(s)}$ are the iterates in iteration s . Let $\boldsymbol{\alpha}_t^{(s)} = \text{diag}(\mathbf{C}^{(s)}\mathbf{h}_t^{(s)})\boldsymbol{\beta}_t^{(s)}$ and substitute $\boldsymbol{\beta}_t^{(s)}$ by using $\boldsymbol{\beta}_t^{(s)} = (\text{diag}(\mathbf{C}^{(s)})\mathbf{h}_t^{(s)})^{-1}\boldsymbol{\alpha}_t^{(s)}$ in Problem (5.4), then we derive from Problem (5.4) to Problem (5.13). According to Theorem 5.2, we calculate $\mathbf{C}^{(s+1)}$ from $\boldsymbol{\alpha}_t^{(s)}$ and $\mathbf{h}_t^{(s)}$ according to Eq. (5.14).

(2). We jointly solve for $\boldsymbol{\beta}_{t:t=\{1,\cdots,T\}}$ and $\mathbf{h}_{t:t=\{1,\cdots,T\}}$ with fixed $\mathbf{C}$ using the exhaustive search method. Given $\mathbf{C}$ is fixed, problem (5.4) becomes

$$\min_{\boldsymbol{\beta}_t,\mathbf{h}_t} \sum_{t=1}^{T} L(\boldsymbol{\beta}_t, \mathbf{h}_t, \mathbf{C}, \mathbf{X}_t, \mathbf{y}_t) + \gamma_1 \sum_{t=1}^{T} ||\boldsymbol{\beta}_t||_2^2 \tag{5.15}$$

which comprises of T learning problems. The optimizing procedure for solving $\boldsymbol{\beta}_t$ and $\mathbf{h}_t$ is summarized into Algorithm 4. Note that when $\mathbf{C}$ is fixed, learning the model parameter of a single task is independent from other tasks. The algorithm searches the most favorable $\mathbf{c}_g$ in an exhaustive search for each individual task, which is effective when G is a small number ($G << T$). The identified group sharing structure $\mathbf{c}_g$ is expected to be the one that decrease the objective of problem (5.15) the most.

The overall BCD algorithm for solving problem (5.4) is described in Algorithm 5. The algorithm iterates between two sub-problems introduced above until convergence.

The loss function we used in the proposed formulation (5.4) is either least squares loss or logistic regression loss. Any other loss function that is convex and differentiable in terms of $\boldsymbol{\alpha}$ will also be appropriate. Solving for $\mathbf{C}$ according to Eq.(5.14) guarantees that problem (5.4) is decreased to a lower bound for the current iterate $\boldsymbol{\beta}_t$. Optimizing Eq.(5.4) with respect to $\mathbf{h}_{t:t=1,\cdots,T}$ and $\boldsymbol{\beta}_{t:t=1,\cdots,T}$ also guarantees a non-increasing objective of problem (5.4). With these conditions, according to the theoretical properties of the BCD algorithm proved in

Algorithm 4 Solving for $\boldsymbol{\beta}_t$, $\mathbf{h}_t$, for $t = 1, \cdots, T$

Input: $\mathbf{X}_t$, $\mathbf{y}_t$, $\mathbf{C}$, γ_1, γ_2 and the pre-defined number of clusters G
for $t = 1, \cdots, T$ **do**
  for $g = 1, \cdots, G$ **do**
    Solve the following problem for $\boldsymbol{\beta}_{t\_new}$

$$\min_{\boldsymbol{\beta}_t} \quad \sum_{t=1}^{T} L(\boldsymbol{\beta}_t, \hat{\mathbf{X}}_t, \mathbf{y}_t) + \gamma_1 \sum_{t=1}^{T} ||\boldsymbol{\beta}_t||_2^2 \tag{5.16}$$

    where $\hat{\mathbf{X}}_t = \mathbf{X}_t \text{diag}(\mathbf{c}_g)$. Let the objective of problem (5.16) be $\boldsymbol{J}(\boldsymbol{\beta})$.
    if $\boldsymbol{J}(\boldsymbol{\beta}_{t\_new}) < \boldsymbol{J}(\boldsymbol{\beta}_t)$ **then**
      Update $\mathbf{h}_t$ to $\mathbf{h}_{t\_new}$ by letting $h_{gt} := 1$ and other elements be zeros;
      Update task-specific model parameter for the t-th task, $\boldsymbol{\beta}_t = \boldsymbol{\beta}_{t\_new}$.
    end if
  end for
end for
Output: $\boldsymbol{\beta}_{t\_new}$, $\mathbf{h}_{t\_new}$

Algorithm 5 The blockwise coordinate descent algorithm to solve problem (5.4)

Input: $\mathbf{X}_t$, $\mathbf{y}_t$, $t = \{1, \cdots, T\}$, as well as γ_1, γ_2, G
Initialize: For all $t = \{1, \cdots, T\}$ and $g = \{1, \cdots, G\}$, let $\mathbf{c}_g^{(0)} = \mathbf{1}$ and then solve Problem (5.16) for $\boldsymbol{\beta}_t^{(0)}$ by using the same $\mathbf{c}_g^{(0)}$, then we initialize $\mathbf{h}_t$ according to the results by running k-means algorithm on $\boldsymbol{\alpha}_t = \text{diag}(\mathbf{c}_g^{(0)})\boldsymbol{\beta}_t$ and let $\boldsymbol{\Theta} = (\mathbf{C}, \mathbf{h}_t, \boldsymbol{\beta}_t)$, $s = 0$
repeat
  Compute $\mathbf{C}^{(s+1)}$ according to the equation (5.14), where $\boldsymbol{\alpha}_t^{(s)} = \text{diag}(\mathbf{C}^{(s)}\mathbf{h}_t)\boldsymbol{\beta}_t^{(s)}$;
  Solve problem (5.15) for $\boldsymbol{\beta}_t^{(s+1)}$ and $\mathbf{h}_t^{(s+1)}$ with fixed $\mathbf{C}^{(s+1)}$ following Algorithm 4;
  Set $s = s + 1$
until $\max(|\boldsymbol{\Theta}^{(s+1)} - \boldsymbol{\Theta}^{(s)}|) < \epsilon$
Output: $\mathbf{C}$, $\mathbf{h}_{t:t=1,\cdots,T}$ and $\boldsymbol{\beta}_{t:t=1,\cdots,T}$

[96], if both sub-problems in Algorithm 5 have unique solutions, which is the case due to the strict convexity of the sub-problem (5.16) and the Cauchy-Schwarz inequality used to derived the analytical formula for $\mathbf{C}$, then Algorithm 5 has at least one accumulation point that is a coordinate-wise minimum point.

Problem (5.16) in Algorithm (4) is convex with respect to $\boldsymbol{\beta}_t$ if the loss function is

convex. When the least squares loss is used, problem (5.16) in Algorithm (4) is a Tikhonov regularization or ridge regression problem in statistical literatures. Each $\boldsymbol{\beta}_t$ has the analytical solution

$$\boldsymbol{\beta}_t = (\hat{\mathbf{X}}_t^\top \hat{\mathbf{X}}_t + \gamma_1 \boldsymbol{I})^{-1} \hat{\mathbf{X}}_t^\top \mathbf{y}_t.$$

When the logistic regression loss is used, problem (5.16) can also be solved by any well-developed gradient based algorithms. The run time of each iteration in Algorithm 5 is linearly proportional to TG.

5.4 The Bound of Expected Risk

This section describes an upper-bound for the expected error of our learning formulation. Note that Problem (5.4) can be equivalently re-written into the following optimization problem for appropriately chosen τ_c and τ_β:

$$\begin{aligned} \min_{\mathbf{C},\mathbf{H},\mathbf{B}} \quad & \sum_{t=1}^{T} L_t(\mathbf{C}, \boldsymbol{\beta}_t, \mathbf{h}_t, \mathbf{X}_t, \mathbf{y}_t) \\ \text{s.t.} \quad & \|\mathbf{C}\|_1 \leq \tau_c, \quad \|\boldsymbol{\beta}_t\|_2 \leq \tau_\beta, \\ & \|\mathbf{h}_t\|_0 \leq 1, \quad \|\mathbf{h}_t\|_F \leq 1, \\ & t = 1, ..., T. \end{aligned} \tag{5.17}$$

where $\boldsymbol{\beta}_t \in \mathbf{B}$, $\mathbf{h}_t \in \mathbf{H}$. Let us, respectively, write the expected risk as

$$R_L(\mathbf{C}, \mathbf{H}, \mathbf{B}) = \mathbb{E}_{(t)} \left[L_t(\mathbf{X}_t, \mathbf{C}, \mathbf{B}, \mathbf{H}, \mathbf{y}_t) \right] \tag{5.18}$$

and the empirical risk as:

$$\hat{R}_L(\mathbf{C},\mathbf{H},\mathbf{B}) = \frac{1}{T}\left[\sum_{t=1}^{T} L_t(\mathbf{X}_t,\mathbf{C},\mathbf{B},\mathbf{H},\mathbf{y}_t)\right]. \tag{5.19}$$

Next we implement Rademacher complexity to measure the complexity of a function class. By introducing it we connect the expected risk with the empirical risk to give the upper-bound.

Define $f(t) = \mathbf{X}_t \text{diag}(\mathbf{C}\mathbf{h}_t)\boldsymbol{\beta}_t$ and $f(t)$ belongs to the function class $F_\Theta = \mathbf{X}_t \text{diag}(\mathbf{C}\mathbf{h}_t)\boldsymbol{\beta}_t$ where $\Theta = (\mathbf{C},\mathbf{H},\mathbf{B} \mid \|\mathbf{C}\|_1 \leq \tau_c, \|\mathbf{h}_t\|_0 \leq 1, \|\mathbf{h}_t\|_F \leq 1, \|\boldsymbol{\beta}_t\|_2 \leq \tau_{\beta_t})$. The main result with respect to the bound of the expected risk is summarized in Theorem 5.3.

Theorem 5.3. *(Bound on Expected risk). Given* $\mathbf{X}_t \in \mathbb{R}^{\ell_t \times d}$, $\mathbf{y}_t \in \mathbb{R}^{\ell_t}$, $t = \{1,\cdots,T\}$, *let* $L_t(f(t),\mathbf{y}_t)$ *be a loss function that has Lipschitz constant* l_t *with respect to* $f(t)$ *and for all* t, l_t *are bounded by* L_l. *Let* $\tau_X = \max_t \|\mathbf{X}_t\|_F$. *Let* p *be a constant where* $0 < p < 1$. *In Problem (5.4), with probability of at least* $1-p$, *the expected risk of an optimal solution* $(\mathbf{C}^*,\mathbf{H}^*,\mathbf{B}^*)$ *will be bounded by:*

$$R_L(f^*) \leq \frac{4L_l}{T}\sum_{t=1}^{T} \tau_c\tau_{\boldsymbol{\beta}}\tau_X\sqrt{\frac{\log 2d}{\ell_t}} + L_l\sqrt{\frac{\log\frac{1}{p}}{2T}}, \tag{5.20}$$

This theorem can be proved based on the following several Lemmas. The following Lemma 5.1 is extended from the theoretical result in [7].

Lemma 5.1. *Let* $\mathcal{R}(F_\Theta)$ *be the Rademacher complexity of the function class* F_Θ *written as:*

$$\mathcal{R}(F_\Theta) = \mathbb{E}[\sup_{f\in F_\Theta} \frac{1}{T}\sum_{t=1}^{T} \omega_t L_t(f(t),\mathbf{y}_t)], \tag{5.21}$$

where each ω_t *takes values* $\{\pm 1\}$ *with equal probability. Then with probability at least* $1-p$,

for all $f \in F_\Theta$ we have:

$$R_L(f) - \hat{R}_L(f) \leq 2\mathbb{E}[\mathcal{R}(F_\Theta)] + L_l\sqrt{\frac{\log \frac{1}{p}}{2T}}. \tag{5.22}$$

Based on Lemma 5.1, an upper bound on $\mathcal{R}_L(f)$ can be derived by bounding $\mathbb{E}[\mathcal{R}(F_\Theta)]$ from above.

Lemma 5.2. *Given $\tau_X = \max_t \|\mathbf{X}_t\|_F$, we have*

$$\mathbb{E}[\mathcal{R}(F_\Theta)] \leq \frac{2L_l}{T}\sum_{t=1}^{T}\tau_c\tau_\beta\tau_X\sqrt{\frac{\log 2d}{\ell_t}}. \tag{5.23}$$

To prove Lemma 5.2, we will use the result of Lemma 5.3 which is described as the following.

Lemma 5.3. *Let $S_w = \{\mathbf{W} \in \mathbb{R}^{d\times d} \mid \|\mathbf{W}\|_* \leq w\}$. For any sequence $\mathbf{A}_i \in \mathbb{R}^{d\times d}$, $i = \{1, \cdots, m\}$, let $a = \max_i \|\mathbf{A}_i\|_2$, and ω_i takes $\{-1, +1\}$ with equal probability. Then,*

$$\mathbb{E}_\omega[\sup_{W\in S_w}\frac{1}{m}\sum_{i=1}^{m}\omega_i\|\mathbf{W}\mathbf{A}_i\|_*] \leq 2aw\sqrt{\frac{\log 2d}{m}}. \tag{5.24}$$

This Lemma re-states Theorem 2 in [50]. By Rademacher contraction principle [66],

$\mathcal{R}(F_\Theta)$ can be upper bounded by

$$\begin{aligned}
\mathcal{R}(F_\Theta) &\leq L_l \mathbb{E}_\omega[\sup_{\|\mathbf{C}\|_1 \leq \tau_c} \frac{1}{T}\sum_{t=1}^{T}\frac{1}{\ell_t}\sum_{i=1}^{\ell_t}\omega_{t_i}\mathbf{x}_i^t \text{diag}(\mathbf{C}\mathbf{h}_t)\boldsymbol{\beta}_t] \\
&= L_l \mathbb{E}_\omega[\sup_{\|\mathbf{C}\|_1 \leq \tau_c} \frac{1}{T}\sum_{t=1}^{T}\frac{1}{\ell_t}\sum_{i=1}^{\ell_t}\omega_{t_i} tr(\text{diag}(\mathbf{C}\mathbf{h}_t)\boldsymbol{\beta}_t\mathbf{x}_i^t)] \\
&\leq L_l \mathbb{E}_\omega[\sup_{\|\mathbf{C}\|_1 \leq \tau_c} \frac{1}{T}\sum_{t=1}^{T}\frac{1}{\ell_t}\sum_{i=1}^{\ell_t}\omega_{t_i} tr(\text{diag}(\mathbf{C}\mathbf{h}_t)\boldsymbol{\beta}_t\mathbf{x}_i^t)] \\
&\leq \frac{2L_l}{T}\sum_{t=1}^{T}(\tau_c \max_i \|\boldsymbol{\beta}_t\mathbf{x}_i^t\|_F\sqrt{\frac{\log 2d}{\ell_t}}).
\end{aligned}$$

where we make use of the fact that $tr(\text{diag}(\mathbf{C}\mathbf{h}_t)) \leq \|\mathbf{C}\|_1 \leq \tau_C$, and the last step is derived by applying Lemma 5.3. For the real situations $\mathbf{X}$ is always upper-bounded or normalized, we assume $\|\mathbf{X}_t\|_F \leq \tau_X$, $t = 1, ..., T$, then one can derive that

$$\max_i \|\boldsymbol{\beta}_t\mathbf{x}_i^t\|_F \leq \|\mathbf{X}_t\|_F\|\boldsymbol{\beta}_t\|_2 \leq \tau_X\tau_\beta$$

So an upper bound of $\mathbb{E}[\mathcal{R}(F_\Theta)]$ is as follows:

$$\mathbb{E}[\mathcal{R}(F_\Theta)] \leq \frac{2L_l}{T}\sum_{t=1}^{T}\tau_c\tau_\beta\tau_X\sqrt{\frac{\log 2d}{\ell_t}}. \tag{5.25}$$

Hence, we prove Lemma 5.2 to be hold.

According to the above three Lemmas, we obtain the conclusion of Theorem 5.3, thus one could construct a feasible solution set by setting proper τ_c and τ_β, then $\hat{R}_L(f)$ and $\mathbb{E}[\mathcal{R}(F_\Theta)]$ can be reasonably small to lower the upper bound of expected risk, which provides basis for the learning process [49].

5.5 Experiments

We compared the proposed method with the following published multi-task learning approaches. The first four methods did not learn the task groups while the last four methods are clustered multi-task learning algorithms. The proposed method was referred to as CMMTFL(2,1).

STL_lasso: Single-task learning is applied to each task independently using $||\boldsymbol{\alpha}_t||_1$ as the regularizer.

DMTL [43]: The dirty model for MTFL with regularizers $||\mathbf{P}||_{1,1}$ and $||\mathbf{Q}||_{1,\infty}$, where $\mathbf{A} = \mathbf{P} + \mathbf{Q}$.

MMTFL(2,1) [101]: Multiplicative MTFL algorithm proposed in our previous work, using $||\mathbf{B}||_2^2$ and $||\mathbf{c}||_1^1$ as regularizers, where $\mathbf{A} = diag(\mathbf{c}) \cdot \mathbf{B}$.

CMTL [42]: Tasks in the same cluster are enforced to be close to a centroid, squared 2-norm are used to measure the compactness between different clusters and within each individual cluster.

cCMTL [119]: Clustered MTL via alternating structure optimization (ASO), where the problem was solved with a convex relaxation.

GMTFL [51]: This method formulate the clustered MTFL problem as a mixed integer programming problem, where the squared trace norm $||\mathbf{W}_g||_*^2$ was imposed to the parameter matrix formed by tasks within g-th group.

GOMTL [54]: Task parameters in one group are assumed to lie in a low dimensional subspace and tasks can share one or more bases. Regularizers are $||\mathbf{S}||_1$ and $||\mathbf{L}||_F^2$ where $\mathbf{A} = \mathbf{LS}$.

We tested different methods on both synthetic and real-world datasets. For all the datasets, we used different proportions of data, such as [25%, 33%, 50%], to be the sizes of

training sets and the rest of data was used for tests. The partition was repeated for 15 times and each varying partition was used for one trail. We calculated the averaged performance of the 15 trails for each method. In order to tune the predefined hyper-parameters for different compared methods, such as the number of clusters in the proposed method, CMTL, cCMTL, GMTFL and the dimension of subspace in GOMTL, we fixed the regularization parameters by default as one in these methods at first, and used one third of the training data as a validation set to select the hyper-parameters from a candidate set [1,2,$\cdots$,10]. This tuning method was also used in [37] and similar to their observation, the candidate set yielded empirically better performance in our experiments. Regularization parameters of each compared method were tuned after we fixed the hyper-parameters mentioned above. We selected the regularization parameters yielding the best performance on the validation set from the choices of 2^k, $k = [-10, -9, \cdots, 7]$. After the first trail, we fixed the selected regularization parameters and other hyper-parameters for the rest trails.

We computed the coefficient of determination, denoted by R^2 which measures the explained variance of the data by the fitted model. In particular, we used the following formula to report performance $R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - f_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$ where $\bar{y}$ is the mean of the observed values of y and f_i is the prediction of the observed y_i. The reported values in our experiments were the averaged R^2 over all tasks. The R^2 value ranges from 0 to 1, and a higher value indicates better regression performance. The classification performance was measured by the AUC (area under curve) computed as the area under the Receiver Operating Characteristic (ROC) curve of the learnt classifier. The numbers we reported in each trial was the AUC averaged over all tasks. The AUC also ranges from 0 to 1 and higher values represent better classification performance.

5.5.1 Synthetic data

In this section, we describe the results of testing compared methods on two synthetic datasets for regression tasks. We created each example $\mathbf{x}$ as the input variables from the multivariate normal distribution $\mathcal{N}(\mathbf{0}, \mathbf{1})$. For each task, we created 200 examples with a certain number of features. We respectively created 100 features for synthetic data D1 and synthetic data D2. The response variables $\mathbf{y}_t$ for each task t was computed by $\mathbf{y}_t = \mathbf{X}_t\boldsymbol{\alpha}_t + \boldsymbol{\epsilon}_t$ with a pre-specified $\boldsymbol{\alpha}_t$, where $\epsilon_t \sim \mathcal{N}(0, \frac{1}{\sqrt{2}})$ is the introduced noise. The values of $\boldsymbol{\alpha}$'s were specified in a particular way for us to explore how the structure of feature sharing influenced the clustering of multiple tasks and the performance of multiple models.

- **Synthetic Data D1.** We created the model parameter matrix $\mathbf{A}$ for 20 tasks in such a way that 10% of the rows in $\mathbf{A}$ were set to non-zero and let all tasks share these features. The non-zero values in $\mathbf{A}$ follows a uniform distribution $\mathbf{U}(0.5, 1.5)$. We then arranged the tasks to follow six different sparse structures (the staircases) as shown in Figure 5.1, where $\mathbf{A}$ was transposed. Each of the remaining features except the 10% common features was used by a comparatively small proportion of the tasks. Consecutive tasks were grouped such that the neighboring groups of tasks shared 7% of the features besides the 10% common features. It was naturally to observe that the synthetic tasks form 6 clusters. Within each cluster, the useful features supposed to be selected by $\mathbf{C}$ were not sparse. We hypothesized that the proposed formulation (5.4) would produce better regression performance.

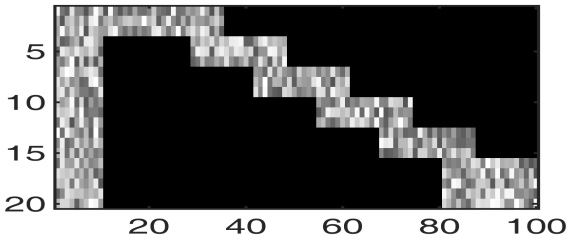


Figure 5.1: The model parameter $\mathbf{A}$ generated on synthetic data D1. We rotate $\mathbf{A}$ by 90 degrees such that each row corresponds to parameters of one task and each column corresponds to one feature dimension. The lighter the color is, the larger an entry of $\mathbf{A}$ was assigned with a magnitude.

- **Synthetic Data D2.** We created 8 tasks with the designed model parameter matrix follows a certain structure. Let us consider a matrix as the following:

$$\begin{bmatrix} 1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 \\ 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 \\ 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 \end{bmatrix}$$

  We created the model parameter matrix $\mathbf{A}$ by expanding the above matrix. Each non-zero entry was replaced by a 20×1 dimensional vector with the elements' magnitudes generated from uniform distribution $\mathbf{U}(0.5, 1.5)$. The sign of each element within the generated vector was the same as the sign used in the above matrix. Each zero entry in the above matrix was expanded to a 20×1 dimensional vector with all zero elements. Hence each task had 100 features. According to the feature sharing structure and the different sparsity pattern in feature space, the first four tasks were designed to form one cluster and the last four tasks formed the other cluster. However, k-means clustering based methods may not achieve the desired clustering structures because tasks sharing

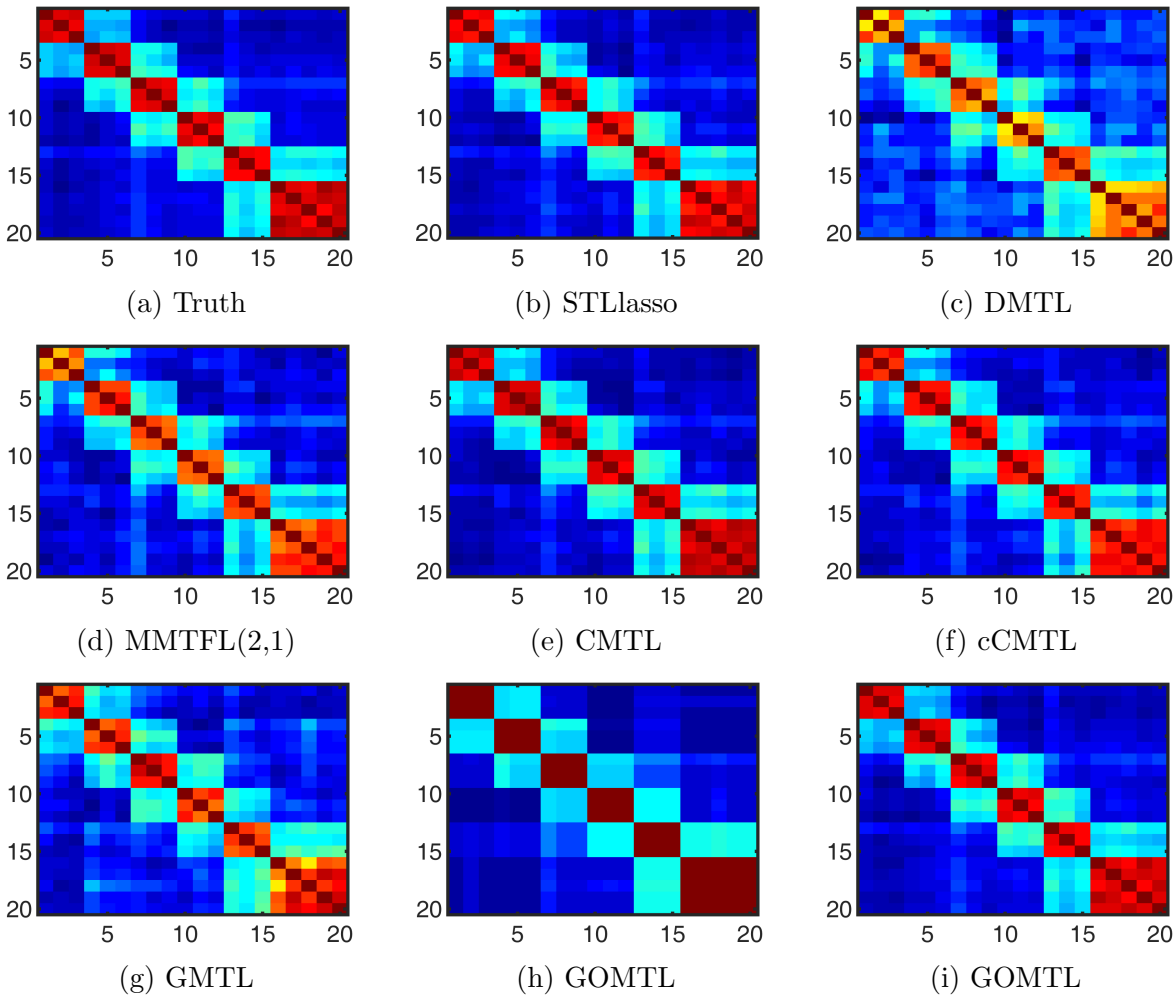
Table 5.1: Comparison of the R^2 values obtained by the different multi-task learning methods on synthetic datasets (where standard deviation 0 means that it is less than 0.01).

Data set		STL_lasso	DMTL	MMTFL(2,1)	CMTL	cCMTL	GMTFL	GOMTL	CMMTFL(2,1)
D1	25%	0.47±0.03	0.45±0.02	0.44±0.02	0.79±0.01	0.62±0.02	0.62±0.02	0.86±0.01	**0.88±0.03**
	33%	0.77±0.01	0.40±0.03	0.67±0.01	0.89±0.01	0.76±0	0.74±0.02	0.89±0.01	**0.96±0.01**
	50%	0.91±0	0.80±0.01	0.91±0.01	0.96±0	0.93±0.01	0.91±0.01	0.91±0	**0.98±0**
D2	25%	0.23±0.04	0.39±0.05	0.17±0.06	0.43±0.04	0.54±0.01	0.42±0.03	0.35±0.03	**0.56±0.03**
	33%	0.45±0.05	0.52±0.03	0.44±0.02	0.63±0.04	0.67±0.03	0.57±0.02	0.66±0.03	**0.83±0.05**
	50%	0.93±0.01	0.92±0.01	0.81±0.01	0.89±0.01	0.91±0.01	0.73±0.02	0.75±0.04	**0.97±0**

the same subset of features still had large dissimilarities. In this case, our method is more suitable to discover the true clustering structure on synthetic data D2.

For the two created datasets, we set the predefined number of clusters to be six and two respectively. Table 5.1 shows the regression performance of compared methods on two synthetic datasets. The proposed method CMMTFL(2,1) consistently achieved the best performance on all trails using different sizes of training sets. The results validated our hypothesis on the performance of CMMTFl(2,1). Our method was more suitable to find the distinguished sparsity patterns among clusters and group tasks sharing the same subset of features together. The performances of CMTL and cCMTL were degraded to some degree on synthetic data D2 when they are compared to those on synthetic data D1, because the similarity of parameters based on Euclidean distances between most tasks in synthetic data D2 were trivial.

Figure 5.2 shows the correlation matrices of the estimated parameters of tasks by different methods on synthetic data D1 using 33% of samples to form training set. The figure shows that our method CMMTFL(2,1) recovered the designed clustering structure to the best, while the GOMTL method was very restrictive to have parameters of tasks to be similar compared to other methods. Figure 5.3 (the upper three figures) illustrates the estimated clustering structure by the proposed method CMMTFl(2,1) on synthetic data D1. The figure shows that CMMTFL(2,1) recovered the true clustering structure after running 10



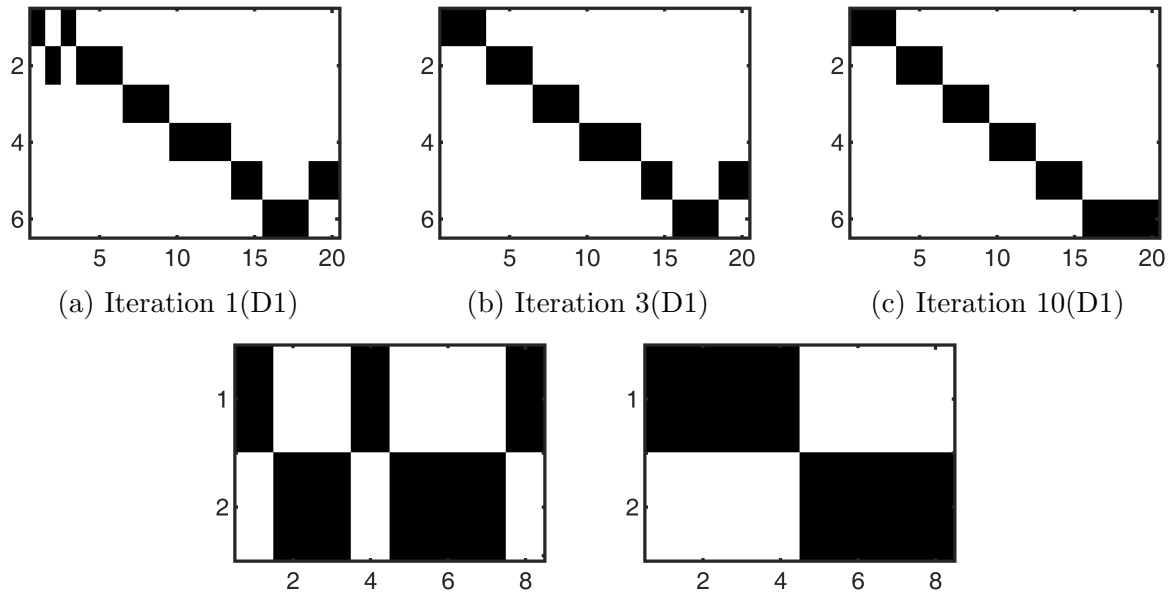
(a) Truth (b) STLlasso (c) DMTL

(d) MMTFL(2,1) (e) CMTL (f) cCMTL

(g) GMTL (h) GOMTL (i) GOMTL

Figure 5.2: The correlation matrices of the synthesized parameter matrix and the other parameter matrices constructed by various methods on synthetic data D1.

iterations of Algorithm 5. Figure 5.4 shows the correlation matrices computed for synthetic data D2 using the same size of training set as what was used for Figure 5.2. Compared to the synthetic data D1, the created tasks in synthetic data D2 had less correlation even two tasks were using the same subset of features. We observed that the proposed method also estimated the model parameters the best and revealed the tasks' relationships that were the closest to the designed truth between tasks. Although the other compared methods such as CMTL and cCMTL can discover the relatedness of tasks on data D2 similar to our method.



(a) Iteration 1(D1) (b) Iteration 3(D1) (c) Iteration 10(D1)

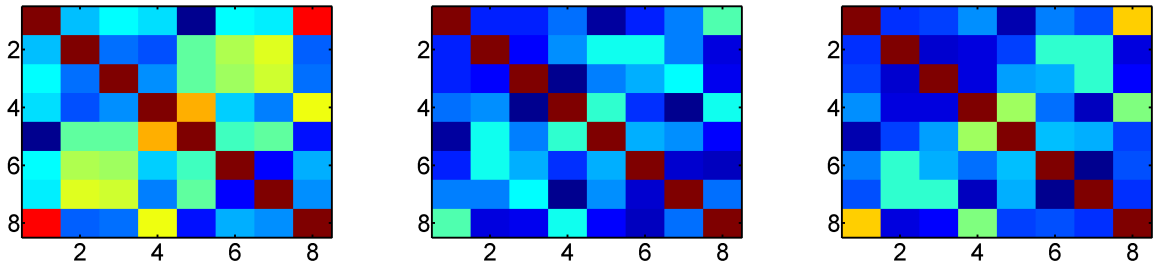
(d) Iteration 1(D2) (e) Iteration 3(D2)

Figure 5.3: The estimated cluster indicator **H** for running Algorithm 5 on synthetic data D1 (5.3a, 5.3b and 5.3c) and D2 (5.3d and 5.3e) on the 33% trail. The dark color indicates that the corresponding element is 1. Rows correspond to clusters while columns are tasks.

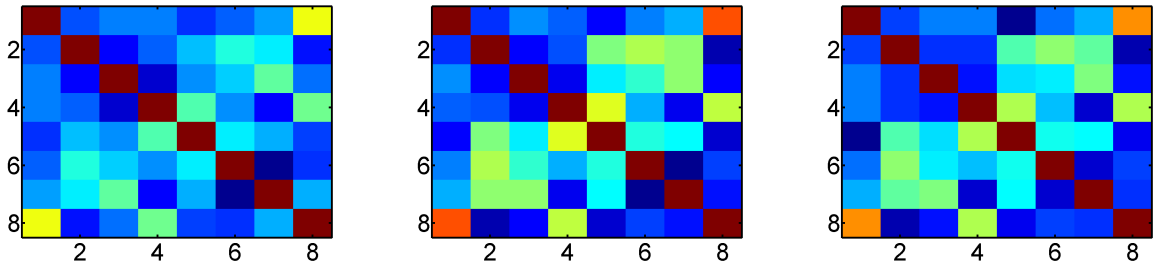
These methods do not impose sparsity on the feature space such that redundant features may be involved in the learning models. The estimated clustering structure learnt from synthetic data D2 is also shown in Figure 5.3 (the lower two figures). We observed that the desired clustering structure was recovered after three iterations by our method.

5.5.2 Microarray data analysis

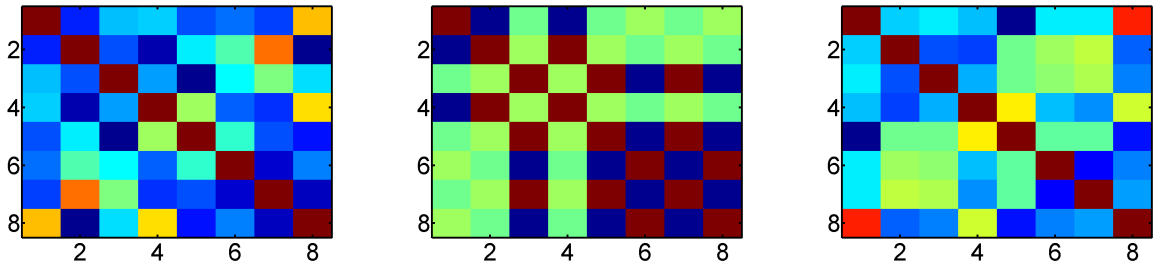
The Microarray Data Analysis dataset used in [102] has 118 microarrays (samples) in total. It utilizes the expression levels of 21 genes in the mevalonate pathway as the features and the expression levels of 18 genes in the plastidial pathway as the responses. This implies that we will use the expression levels of 21 genes to learn 18 tasks, each of which is a regression problem. All the features and responses are made log transformated first and then centered



(a) Truth (b) STLlasso (c) DMTL



(d) MMTFL(2,1) (e) CMTL (f) cCMTL



(g) GMTL (h) GOMTL (i) CMMTFL21

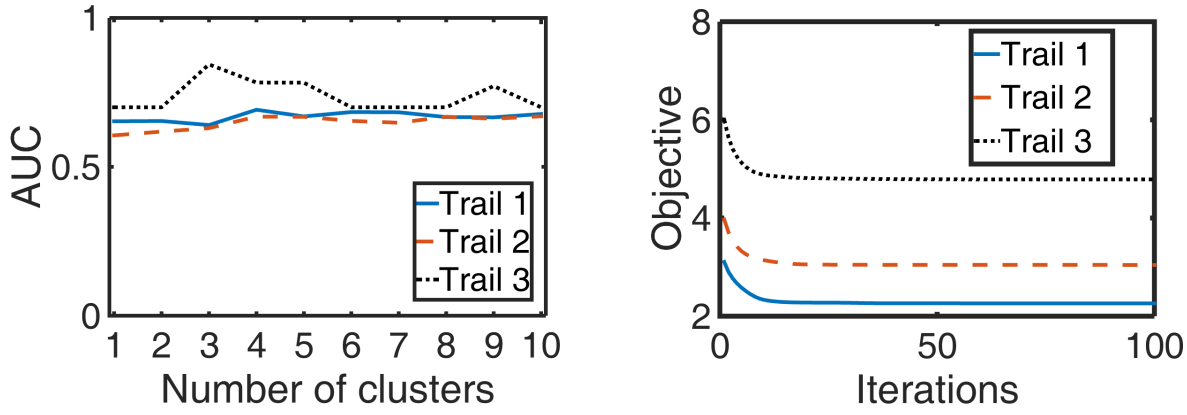
Figure 5.4: The correlation matrices of the synthesized parameter matrix and the other parameter matrices constructed by various methods on synthetic data D2.

and standardized to unit variance. Table 5.2 shows the averaged R^2 based on 15 random partitions. We observe that the proposed method achieved a superior performance to all other methods. We also notice that the clustered MTL methods generally performed better than the non-clustered MTL methods on this data.

5.5.3 MHC-I binding prediction

The IEDB MHC-I peptide binding dataset was proposed and used in [74, 42]. This dataset contains binding affinities of various peptides with different MHC-I molecules. In this experiment, we have 10 binary classification tasks. The number of examples for each task vary from 59 to 197. The goal of each task is to predict whether a peptide binds a molecule, such that 10 tasks correspond to 10 different molecules and each example in the task thus corresponds to one peptide.

Table 5.3 shows the classification performance of compared methods on the MHC-I dataset. Our method CMMTFL(2,1) achieved the best performance on all three trails. The averaged AUC achieved by CMMTFL(2,1) were higher than other methods by 3 to 13 percentages. We also examined how the predefined number of clusters influenced the performance of our method. We let the number of clusters increase from 1 to 10 for training the proposed model. We computed the AUC on the validation set used in each trail. Figure 5.5 (the left Figure 5.5a) shows the AUC achieved by our method on three trials as the number of clusters increased. Our method reached the highest AUC when the number of cluster was set to be 3. Figure 5.5 (the right Figure 5.5b) illustrated the objective values as the Algorithm 5 run for one time on different trails. For all the experiments on this dataset, our algorithm terminated within 200 iterations.



(a) (b)

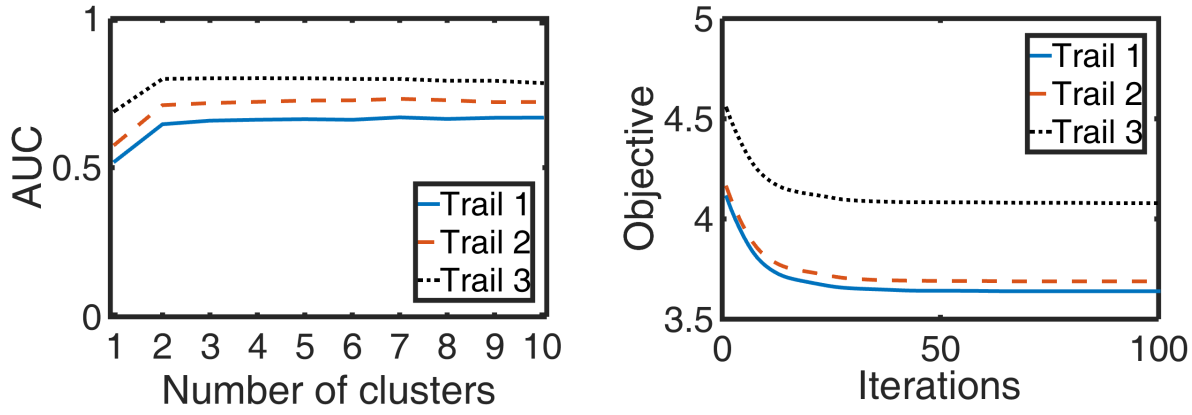
Figure 5.5: Left: The AUC values with an increasing number of clusters for CMMTFL(2,1) on MHC-I data. Right: the objective values computed for CMMTFL(2,1) as we run Algorithm 5 on MHC-I data.

Table 5.2: Comparison of the performances in terms of R^2 obtained by the different multi-task learning methods on the Microarray dataset (where standard deviation 0 means that it is less than 0.01).

Data set		STL_lasso	DMTL	MMTFL(2,1)	CMTL	cCMTL	GMTFL	GOMTL	CMMTFL(2,1)
Microarray	25%	0.38±0.01	0.36±0.02	0.34±0.01	0.59±0.02	0.47±0.02	0.45±0.02	0.56±0.03	**0.72±0.02**
	33%	0.54±0.02	0.39±0.02	0.47±0.02	0.63±0.02	0.55±0.03	0.52±0.02	0.58±0.03	**0.76±0.02**
	50%	0.69±0.01	0.63±0.01	0.64±0.01	0.69±0.01	0.67±0.03	0.65±0.03	0.60±0.03	**0.78±0.01**

Table 5.3: Comparison of the performances in terms of AUC obtained by the different multi-task learning methods on MHC-I binding and Animal recognition datasets (where standard deviation 0 means that it is less than 0.01).

Data set		STL_lasso	DMTL	MMTFL(2,1)	CMTL	cCMTL	GMTFL	GOMTL	CMMTFL(2,1)
MHC-I	25%	0.65±0.01	0.67±0.02	0.67±0.01	0.71±0.01	0.73±0.01	0.65±0.01	0.65±0	**0.75±0.01**
	33%	0.69±0.01	0.67±0.02	0.64±0.01	0.74±0.01	0.73±0	0.70±0.01	0.67±0.02	**0.77±0.01**
	50%	0.70±0.01	0.71±0.01	0.70±0.02	0.74±0	0.74±0.01	0.73±0.02	0.67±0.01	**0.80±0.02**
Animal	25%	0.62±0.01	0.57±0.01	0.56±0.01	0.62±0.01	0.63±0.01	0.65±0.01	0.60±0.02	**0.67±0.01**
	33%	0.66±0.01	0.60±0.02	0.63±0.01	0.65±0.01	0.63±0.01	0.66±0.01	0.63±0.03	**0.67±0.01**
	50%	0.68±0	0.61±0.01	0.62±0.0	0.66±0.01	0.67±0.01	**0.69±0.02**	0.65±0	**0.69±0.01**



(a) (b)

Figure 5.6: Left: The AUC values with an increasing number of clusters for CMMTFL(2,1) on Animal recognition data. Right: the objective function values computed for CMMTFL(2,1) as we run Algorithm 5 on Animal recognition data.

5.5.4 Animal recognition

We also tested the proposed methods on an image classification problem. The dataset we used was proposed in [51]. This data contains 20 tasks with each task aiming to predict if an image shows one class of animals. Each task contains 200 images with half of those showing one class of animals and the other half having different animals. Each image or example was represented by 202 features.

The performance of compared methods in this experiment was summarized in Table 5.3, which shows that our method outperformed all other methods on the first two trails and achieved the best AUC jointly with GMTL on the third trial. In Figure 5.6, we illustrate the computed AUC on the validation set in each trail when different numbers of clusters were predefined. Figure 5.6 also shows the yielded objective function values for one run of training on the animal recognition data.

5.6 Conclusions

In this work, we propose a new approach to learn task grouping in MTL. The method does not require tasks in one cluster to have similar model parameters, but clustering tasks by examining if they share a similar subset of feature representations. Sparse structure of feature space is exploited for each cluster to remove the irrelevant features across tasks within the cluster. Empirical results on both synthetic and real world datasets validate that the proposed method is capable to produce desired clustering structures among tasks and thus achieve superior performance to state-of-art methods.

For future work, we will be interested to inferring the number of clusters from the training dataset automatically, since the current approach still requires the number of clusters to be specified as a hyperparameter, which needs a extra tuning procedure and thus increases the computational costs.

Chapter 6

Concluding remarks

In this dissertation, we have presented our studies toward solving Machine Learning problems in Imprecisely Supervised Learning, where we construct learning models using imprecise or limited supervision. In the first direction of the study, we propose new models to learn classifiers from inconsistent annotations collected from multiple labelers with varying expertise, based on bi-convex optimization algorithm. Adding another layer of difficulty, we develop a new approach to learn classifiers from dual annotation ambiguity, where multiple inconsistent versions of labels are associated with a bag of instances or examples with each individual instance having no label, moreover, a bag gets a positive class label as long as one of its instances shows evidence to be positive, otherwise the bag is labeled negative with all its instances to be negative. Along the second direction, we investigate the new approaches in multitask learning, where related tasks are trained jointly with the learning expertise of the tasks shared to the benefit of all. We propose two new formulations in multiplicative multitask feature learning, assuming that the related tasks share a common subspace of features. We also develop an approach to learning task grouping with multiplicative multitask feature

sharing patterns within each group of tasks. Tasks using different subset of features will be clustered into different clusters. Our contributions in this dissertation are summarized as the following:

- **Learning classifiers from multiple inconsistent annotations.** We develop bi-convex programs based on SVM algorithm to construct classifiers and estimate the reliability of each labeler simultaneously. Each labeler is associated with a reliability parameter, which can be a constant, or class-dependent, or varies for different examples. The hinge loss is modified by replacing the true labels by the weighted combination of labelers' labels with reliabilities as weights. Statistical justification is discussed to motivate the use of linear combination of labels. In parallel to the expectation-maximization algorithm for logistic based methods, efficient alternating algorithms are developed to solve the proposed bi-convex programs. Experimental results on benchmark datasets and three real-world biomedical problems demonstrate that the proposed methods either outperform or are competitive to the state of the art.

- **Learning classifiers from dual annotation ambiguity.** We propose a novel optimization framework to learn classifiers from dual annotation ambiguity, where a class label is associated with a bag of instances rather than each instance itself and multiple inconsistent class labels are collected from annotators with varying expertise. We modify the hinge loss to employ the weighted consensus of different labelers' labels and further generalize the notion of loss functions to bags of multiple instances. An alternating optimization algorithm has been derived to efficiently solve the two models. The proposed algorithms outperform existing methods on benchmark data sets collected for document classification, real-life crowd-sourced data sets, and a medical problem of heart wall motion analysis with diagnoses from multiple radiologists.

- **Multiplicative multitask feature learning.** We propose and examine a general framework of the multiplicative decomposition that enables a variety of regularizers to be applied. The general form corresponds to a family of MTFL methods, including all early methods that decompose model parameters as a product of two components. Our theoretical analysis has revealed that this family of methods is actually equivalent to the joint regularization based approach but with a more general form of regularizers, including matrix-norm based and non-matrix-norm based regularizers. Two new MTFL formulations are derived from the proposed general framework. We empirically illustrate the scenarios where the two new formulations are more suitable for solving the MTFL problems.

- **Learning task grouping with multiplicative multitask feature sharing patterns within each group of tasks.** By decomposing each task's model parameter vector into a component-wise product of two vectors with one vector selecting features and the other one capturing the weights on the selected features for each task, we cluster tasks into one cluster if they select the same subset of features. We formulate an optimization problem to jointly estimate the parameters and grouping structure of the tasks. The decomposed components can be regularized differently according to hypothesized feature sharing structure. An alternating optimization algorithm has been developed to solve the proposed problem. Our method outperforms several other clustered multitask learning methods in the experiments with both synthetic and real-world datasets.

Bibliography

[1] A. Agarwal, H. D. Iii, and S. Gerber. Learning multiple tasks using manifold regularization. In *Proceedings of Advances in Neural Information Processing Systems*, pages 46–54, 2010.

[2] P. S. Albert and L. E. Dodd. A cautionary note on the robustness of latent class models for estimating diagnostic error without a gold standard. *Biometrics*, 60(2):427–435, 2004.

[3] R. K. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853, Dec. 2005.

[4] S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. *Advances in neural information processing systems*, 15:561–568, 2003.

[5] A. Argyriou, T. Evgeniou, and M. Pontil. Multi-task feature learning. In *Proceedings of Advances in Neural Information Processing Systems*, pages 41–48. 2007.

[6] A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

[7] P. L. Bartlett and S. Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *The Journal of Machine Learning Research*, 3:463–482, 2003.

[8] A. Barzilai and K. Crammer. Convex multi-task learning by clustering. In *Proceedings of Artificial Intelligence and Statistics Conference*, pages 65–73, 2015.

[9] J. C. Bezdek and R. J. Hathaway. Some notes on alternating optimization. In *Advances in Soft Computing, Lecture Notes in Computer Sciences*, volume 2275, pages 288–300, 2002.

[10] J. Bi and J. Liang. Multiple instance learning of pulmonary embolism detection with geodesic distance along vascular structure. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2007.

[11] J. Bi, J. Sun, Y. Wu, H. Tennen, and S. Armeli. A machine learning approach to college drinking prediction and risk factor identification. *ACM Transactions on Intelligent Systems Technology*, 4:72:1–72:24, 2013.

[12] J. Bi and X. Wang. Learning classifiers from dual annotation ambiguity via a minmax framework. *Neurocomputing*, 151, Part 2:891 – 904, 2015.

[13] J. Bi, T. Xiong, S. Yu, M. Dundar, and R. B. Rao. An improved multi-task learning approach with applications in medical diagnosis. In *Proceedings of ECML'08*, pages 117–132, 2008.

[14] C. Blaschke, E. Leon, M. Krallinger, and A. Valencia. Evaluation of biocreative assessment of task 2. *BMC Bioinformatics*, 6(Suppl 1):S16, 2005.

[15] J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. Wortman. Learning bounds for domain adaptation. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances*

in Neural Information Processing Systems 20, pages 129–136. MIT Press, Cambridge, MA, 2008.

[16] E. Bonilla, K. M. Chai, and C. Williams. Multi-task gaussian process prediction. In *Advances in neural information processing systems*, pages 153–160, 2007.

[17] K. H. Borgwardt. *The Simplex Method: a probabilistic analysis*, volume 1 in Algorithms and Combinatorics. Springer-Verlag, New York, 1980.

[18] J. D. Burger, E. Doughty, S. Bayer, D. Tresner-Kirsch, B. Wellner, J. Aberdeen, K. Lee, M. G. Kann, and L. Hirschman. Validating candidate gene-mutation relations in medline abstracts via crowdsourcing. In *Data Integration in the Life Sciences*, pages 83–91. Springer, 2012.

[19] J. Chen, J. Zhou, and J. Ye. Integrating low-rank and group-sparse structures for robust multi-task learning. In *Proceedings of International Conference on Knowledge Discovery and Data Mining*, pages 42–50, 2011.

[20] Y. Chen, J. Bi, and J. Wang. MILES: multiple instance learning via embedded instance selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(12):1–17, 2006.

[21] K. Crammer, M. Kearns, and J. Wortman. Learning from multiple sources. *Journal of Machine Learning Research*, 9(2):1757–1774, 2009.

[22] G. B. Dantzig and G. Infanger. *Stochastic Programming*, volume v.v. 150. Springer, Dordrecht, 2011.

[23] A. P. Dawid and A. M. Skeene. Maximum likelihood estimation of observed error-rates using the EM algorithm. *Applied Statistics*, 28(1):20–28, 1979.

[24] O. Dekel, O. Shamir, and I. th Annual International Conference on Machine Learning. Good learners for evil teachers. In *Proceedings of the International Conference on Machine Learning*, pages 233–240, 2009.

[25] T. Evgeniou and M. Pontil. Regularized multi–task learning. In *Proceedings of International Conference on Knowledge Discovery and Data Mining*, pages 109–117, 2004.

[26] M. Fang, J. Yin, and D. Tao. Active learning for crowdsourcing using knowledge transfer. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, pages 1809–1815, 2014.

[27] T. Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.

[28] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003*, volume 2, pages 264–271, 2003.

[29] P. Gong, J. Ye, and C. Zhang. Robust multi-task feature learning. In *Proceedings of International Conference on Knowledge Discovery and Data Mining*, pages 895–903, 2012.

[30] P. Gong, J. Ye, and C. Zhang. Multi-stage multi-task feature learning. *The Journal of Machine Learning Research*, 14(1):2979–3010, 2013.

[31] B. M. Good and A. I. Su. Crowdsourcing for bioinformatics. *Bioinformatics*, page btt333, 2013.

[32] I. R. Goodman and S. Kotz. Multivariate θ-generalized normal distributions. *Journal of Multivariate Analysis*, 3(2):204–219, 1973.

[33] J. Gorski, F. Pfeuffer, and K. Klamroth. Biconvex sets and optimization with biconvex functions: a survey and extensions. *Mathematical Methods of Operations Research*, 66(3):373–407, 2007.

[34] W. Greene. *Econometric Analysis*. Prentice Hall, fifth edition, 2002.

[35] S. Guo, O. Zoeter, and C. Archambeau. Sparse bayesian multi-task learning. In *Advances in Neural Information Processing Systems*, pages 1755–1763, 2011.

[36] L. Han and Y. Zhang. Learning multi-level task groups in multi-task learning. In *AAAI Conference on Artificial Intelligence*, pages 2638–2644, 2015.

[37] L. Han and Y. Zhang. Learning tree structure in multi-task learning. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 397–406, New York, NY, USA, 2015. ACM.

[38] C.-J. Ho, S. Jabbari, and J. W. Vaughan. Adaptive task assignment for crowdsourced classification. In *Proceedings of the International Conference on Machine Learning*, pages 534–542, 2013.

[39] S. L. Hui and X. H. Zhou. Evaluation of diagnostic tests without gold standards. *Statistical methods in medical research*, 7(4):354–70, 1998.

[40] IBM ILOG CPLEX Division, New York, NY. *IBM ILOG CPLEX Callable library 12.1 Reference Manual*, 2009.

[41] T. W. J. B. J. Whitehill, P. Ruvolo and J. Movellan. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Proceedings of the 2009 Neural Information Processing Systems (NIPS) Conference.*, pages 2035–2043, 2009.

[42] L. Jacob, B. Francis, and J.-P. Vert. Clustered multi-task learning: a convex formulation. In *Proceedings of Advances in Neural Information Processing Systems*, pages 745–752, 2008.

[43] A. Jalali, S. Sanghavi, C. Ruan, and P. K. Ravikumar. A dirty model for multi-task learning. In *Proceedings of Advances in Neural Information Processing Systems*, pages 964–972, 2010.

[44] R. Jin and Z. Ghahramani. Learning with multiple labels. In S. T. S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems*, pages 897–904. MIT Press, Cambridge, MA, 2003.

[45] R. Jin, S. Wang, and Z.-H. Zhou. Learning a distance metric from multi-instance multi-label data. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 896–902, 2009.

[46] T. Kadir and M. Brady. Saliency, scale and image description. *International Journal of Computer Vision*, 45(2):83–105, 2001.

[47] H. Kajino and H. Kashima. A convex formulation for learning from crowds. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, pages 73–79, 2012.

[48] H. Kajino, Y. Tsuboi, and H. Kashima. Clustering crowds. In *Proceedings of the AAAI conference on Artificial Intelligence*, pages 1120–1127, 2013.

[49] S. M. Kakade, S. Shalev-Shwartz, and A. Tewari. Regularization techniques for learning with matrices. *J. Mach. Learn. Res.*, 13:1865–1890, June 2012.

[50] S. M. Kakade, K. Sridharan, and A. Tewari. On the complexity of linear prediction:

Risk bounds, margin bounds, and regularization. In *Advances in neural information processing systems*, pages 793–800, 2009.

[51] Z. Kang, K. Grauman, and F. Sha. Learning with whom to share in multi-task feature learning. In *Proceedings of International Conference on Machine Learning*, pages 521–528, 2011.

[52] D. R. Karger, S. Oh, and D. Shah. Iterative learning for reliable crowdsourcing systems. In *Advances in Neural Information Processing Systems*, pages 1953–1961, 2011.

[53] V. Klee and G. J. Minty. How good is the simplex algorithm? in inequalities-III. pages 159–175, 1972.

[54] A. Kumar and H. Daume III. Learning task grouping and overlap in multi-task learning. In *Proceedings of International Conference on Machine Learning*, pages 1383–1390, 2012.

[55] S. Lee, J. Zhu, and E. Xing. Adaptive multi-task lasso: with application to eQTL detection. In *Proceedings of Advances in Neural Information Processing Systems*, pages 1306–1314. 2010.

[56] S.-I. Lee, V. Chatalbashev, D. Vickrey, and D. Koller. Learning a meta-level prior for feature relevance from multiple related tasks. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pages 489–496, New York, NY, USA, 2007.

[57] C. Liu and Y.-M. Wang. Truelabel+confusion: A spectrum of probabilistic models in analyzing multiple ratings. In *Proceedings of the International Conference on Machine Learning*, pages 225–232, 2012.

[58] J. Liu, S. Ji, and J. Ye. Multi-task feature learning via efficient $\ell_{1,2}$-norm minimization. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 339–348, 2009.

[59] Q. Liu, X. Liao, H. Li, J. R. Stack, and L. Carin. Semisupervised multitask learning. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(6):1074–1086, 2009.

[60] A. Lozano and G. Swirszcz. Multi-level lasso for sparse multi-task regression. In *Proceedings of International Conference on Machine Learning*, pages 361–368, 2012.

[61] D. D. Lucas, R. Klein, J. Tannahill, D. Ivanova, S. Brandon, D. Domyancic, and Y. Zhang. Failure analysis of parameter-induced simulation crashes in climate models. *Geoscientific Model Development Discussions*, 6(1):585–623, 2013.

[62] J. Ma, R. P. Sheridan, A. Liaw, G. E. Dahl, and V. Svetnik. Deep neural nets as a method for quantitative structureactivity relationships. *Journal of Chemical Information and Modeling*, 55(2):263–274, 2015.

[63] O. Maron and A. L. Ratan. Multiple instance learning for natural scene classification. In *Proceedings of the International Conference on Machine Learning*, pages 341–349, 1998.

[64] A. Maurer, M. Pontil, and B. Romera-Paredes. Sparse coding for multitask and transfer learning. In *Proceedings of the 30th International Conference on Machine Learning*, pages 343–351, 2013.

[65] L. Meier, S. v. d. Geer, and P. Bühlmann. The group lasso for logistic regression. *Journal of the Royal Statistical Society: Series B*, 70 Part 1:53–71, 2008.

[66] R. Meir and T. Zhang. Generalization error bounds for bayesian mixture algorithms. *The Journal of Machine Learning Research*, 4:839–860, 2003.

[67] C. A. Micchelli, J. M. Morales, and M. Pontil. Regularizers for structured sparsity. *Advances in Computational Mathematics*, 38(3):455–489, 2013.

[68] B. Mozafari, P. Sarkar, M. J. Franklin, M. I. Jordan, and S. Madden. Active learning for crowd-sourced databases. *Computing Research Repository (CoRR)*, abs/1209.3686, 2012.

[69] T. B. Nguyen, S. Wang, V. Anugu, N. Rose, M. McKenna, N. Petrick, J. E. Burns, and R. M. Summers. Distributed human intelligence for colonic polyp classification in computer-aided detection for ct colonography. *Radiology*, 2012.

[70] G. Obozinski and B. Taskar. Multi-task feature selection. In *Technical report, Statistics Department, UC Berkeley*, 2006.

[71] T. Ojala, M. Pietikainen, and D. Harwood. Performance evaluation of texture measures with classification based on kullback discrimination of distributions. In *Proceedings of the 12th International Conference on Pattern Recognition (ICPR), 1994*, volume 1, pages 582–585, 1994.

[72] S. B. P. Welinder, S. Branson and P. Perona. The multidimensional wisdom of crowds. In *Proceedings of the 2010 Neural Information Processing Systems (NIPS) Conference*, pages 2424–2432, 2010.

[73] A. Passos, P. Rai, J. Wainer, and H. Daume III. Flexible modeling of latent task structures in multitask learning. In *Proceedings of International Conference on Machine Learning*, pages 1103–1110, 2012.

[74] B. Peters, H.-H. Bui, S. Frankild, M. Nielsen, C. Lundegaard, E. Kostem, D. Basch, K. Lamberth, M. Harndahl, W. Fleri, et al. A community resource benchmarking predictions of peptide binding to mhc-i molecules. *PLoS Comput Biol*, 2(6):e65, 2006.

[75] N. Pochet and J. Suykens. Support vector machines versus logistic regression: improving prospective performance in clinical decision-making. *Ultrasound in Obstetrics & Gynecology*, 27(6):607–608, 2006.

[76] M. Qazi, G. Fung, S. Krishnan, J. Bi, B. Rao, and A. Katz. Automated heart abnormality detection using sparse linear classifiers. *IEEE Engineering in Medicine and Biology*, 26(2):56–63, 2007.

[77] A. Quattoni, X. Carreras, M. Collins, and T. Darrell. An efficient projection for l1,infinity regularization. In *Proceedings of International Conference on Machine Learning*, pages 108–115, 2009.

[78] P. Rai and H. Daume. Infinite predictor subspace models for multitask learning. In *International Conference on Artificial Intelligence and Statistics*, pages 613–620, 2010.

[79] S. Ray and M. Craven. Supervised versus multiple instance learning: an empirical comparison. In *Proceedings of the International Conference on Machine Learning*, 2005.

[80] V. C. Raykar, B. Krishnapuram, J. Bi, M. Dundar, and B. Rao. Bayesian multiple instance learning: automatic feature selection and inductive transfer. *Proceedings of the 25th International Conference on Machine Learning*, pages 808–815, 2008.

[81] V. C. Raykar and S. Yu. Ranking annotators for crowdsourced labeling tasks. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, editors, *Ad-*

vances in Neural Information Processing Systems 20, pages 1809–1817, Cambridge, MA, 2011. MIT Press.

[82] V. C. Raykar, S. Yu, G. H. Valadez, C. Florin, L. Bogoni, L. H. Zhao, and L. Moy. Learning from crowds. *Journal of Machine Learning Research*, 11:1297–1322, 2010.

[83] V. C. Raykar, S. Yu, L. H. Zhao, A. Jerebko, C. Florin, G. H. Valadez, L. Bogoni, and L. Moy. Supervised learning from multiple experts: who to trust when everyone lies a bit. *Proceedings of the 26th International Conference on Machine Learning*, pages 96–103, 2009.

[84] R.Duin, P.Juszczak, P.Paclik, E.Pekalska, D. Ridder, D.Tax, and S.Verzakov. Prtools, a matlab toolbox for pattern recognition, 2010.

[85] D. A. Salazar, J. I. Vélez, and J. C. Salazar. Comparison between svm and logistic regression: Which one is better to discriminate? *Revista Colombiana de Estadística*, 35(2):223–237, 2012.

[86] B. Settles, M. Craven, and S. Ray. Multiple-instance active learning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1289–1296. MIT Press, 2008.

[87] R. Shamir. Probabilistic analysis in linear programming. *Statistical Science*, 8(1):57–64, 1993.

[88] A. Shapiro, D. Dentcheva, and A. Ruszczynski. *Lectures on Stochastic Programming: Modeling and Theory*. SIAM, Philadelphia, PA, 2009.

[89] V. S. Sheng, F. Provost, and P. G. Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining*, pages 614–622, 2008.

[90] A. Singla, I. Bogunovic, G. Bartók, A. Karbasi, and A. Krause. Near-optimally teaching the crowd to classify. In *Proceedings of the 31st International Conference on Machine Learning*, pages 154–162, 2014.

[91] P. Smyth, U. Fayyad, M. Burl, P. Perona, and P. Baldi. Inferring ground truth from subjective labeling of Venus images. pages 1085–1092, 1995.

[92] R. Snow, B. O'Connor, D. Jurafsky, and A. Ng. Cheap and fast - but is it good? evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods on Natural Language Processing*, pages 254–263, 2008.

[93] D. A. Spielman and S.-H. Teng. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *Journal of the ACM (JACM)*, 51(3):385–463, 2004.

[94] D. Tax. MIL, a Matlab toolbox for multiple instance learning, Mar 2013.

[95] Y. Tian and J. Zhu. Learning from crowds in the presence of schools of thought. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 226–234, 2012.

[96] P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109(3):475–494, 2001.

[97] B. A. Turlach, W. N. Wenables, and S. J. Wright. Simultaneous variable selection. *Technometrics*, 47(3):349–363, 2005.

[98] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 1995.

[99] T. Verplancke, S. Van Looy, D. Benoit, S. Vansteelandt, P. Depuydt, F. De Turck, and J. Decruyenaere. Support vector machine versus logistic regression modeling for prediction of hospital mortality in critically ill patients with haematological malignancies. *BMC Medical Informatics and Decision Making*, 8(1):56, 2008.

[100] P. Viola, J. C. Platt, and C. Zhang. Multiple instance boosting for object detection. In *In Neural Information Processing Systems 18*, pages 1419–1426. MIT Press, 2006.

[101] X. Wang, J. Bi, S. Yu, and J. Sun. On multiplicative multitask feature learning. In *Advances in Neural Information Processing Systems*, pages 2411–2419, 2014.

[102] A. Wille, P. Zimmermann, E. Vranová, A. Fürholz, O. Laule, S. Bleuler, L. Hennig, A. Prelić, P. von Rohr, L. Thiele, E. Zitzler, W. Gruissem, and P. Bühlmann. Sparse graphical gaussian modeling of the isoprenoid gene network in arabidopsis thaliana. *Genome Biology*, 4:1–13, 2004.

[103] T. Xiong, J. Bi, B. Rao, and V. Cherkassky. Probabilistic joint feature selection for multi-task learning. In *Proceedings of SIAM International Conference on Data Mining (SDM)*, pages 69–76, 2006.

[104] L. Xu, A. Huang, J. Chen, and E. Chen. Exploiting task-feature co-clusters in multi-task learning. In *AAAI Conference on Artificial Intelligence*, pages 1931–1937, 2015.

[105] Y. Xue, X. Liao, L. Carin, and B. Krishnapuram. Multi-task learning for classification with dirichlet process priors. *Journal of Machine Learning Research*, 8:35–63, 2007.

[106] Y. Yan, R. Rosales, G. Fung, and J. Dy. Modeling multiple annotator expertise in the semi-supervised learning scenario. *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence*, pages 241–248, 2010.

[107] Y. Yan, R. Rosales, G. Fung, M. Schmidt, G. Hermosillo, L. Bogoni, L. Moy, and J. Dy. Modeling annotator expertise: learning when everybody knows a bit of something. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 932–939, 2010.

[108] M. Yang, Y. Li, and Z. M. Zhang. Multi-task learning with gaussian matrix generalized inverse gaussian model. In *Proceedings of the 30th International Conference on Machine Learning*, pages 423–431, 2013.

[109] M. Yetisgen-Yildiz, I. Solti, F. Xia, and S. R. Halgrim. Preliminary experience with amazon's mechanical turk for annotating medical named entities. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 180–183. Association for Computational Linguistics, 2010.

[110] K. Yu, V. Tresp, and A. Schwaighofer. Learning gaussian processes from multiple tasks. In *Proceedings of the 22Nd International Conference on Machine Learning*, ICML '05, pages 1012–1019, New York, NY, USA, 2005.

[111] S. Yu, V. Tresp, and K. Yu. Robust multi-task learning with t-processes. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pages 1103–1110, 2007.

[112] M.-L. Zhang and Z.-H. Zhou. M3MIML: A maximum margin method for multi-instance multi-label learning. In *Proceedings of the 8th IEEE International Conference on Data Mining*, pages 688–697. IEEE, 2008.

[113] Y. Zhang and D.-Y. Yeung. A convex formulation for learning task relationship in multi-task learning. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 2010.

[114] Y. Zhang, D.-Y. Yeung, and Q. Xu. Probabilistic multi-task feature selection. In *Proceedings of Advances in Neural Information Processing Systems*, pages 2559–2567, 2010.

[115] P. Zhao, G. Rocha, and B. Yu. The composite absolute penalties family for grouped and hierarchical variable selection. *The Annals of Statistics*, pages 3468–3497, 2009.

[116] W. Zhong and J. Kwok. Convex multitask learning with flexible task clusters. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 49–56, July 2012.

[117] D. Zhou, Q. Liu, J. C. Platt, and C. Meek. Aggregating ordinal labels from crowds by minimax conditional entropy. In *Proceedings of the 31st International Conference on Machine Learning*, pages 262–270, 2014.

[118] D. Zhou, J. Platt, S. Basu, and Y. Mao. Learning from the wisdom of crowds by minimax entropy. In *Advances in Neural Information Processing Systems (NIPS)*, volume 25, pages 2204–2212, 2012.

[119] J. Zhou, J. Chen, and J. Ye. Clustered multi-task learning via alternating structure optimization. In *Proceedings of Advances in Neural Information Processing Systems*, pages 702–710, 2011.

[120] J. Zhou, J. Chen, and J. Ye. Clustered multi-task learning via alternating structure optimization. In *Advances in neural information processing systems*, pages 702–710, 2011.

[121] Q. Zhou and Q. Zhao. Flexible clustered multi-task learning by learning representative

tasks. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 38(2):266–278, 2016.

[122] Y. Zhou, R. Jin, and S. C. Hoi. Exclusive lasso for multi-task feature selection. In *Proceedings of International Conference on Artificial Intelligence and Statistics*, pages 988–995, 2010.

[123] Z.-H. Zhou and M.-L. Zhang. Multi-instance multi-label learning with application to scene classification. In *Advances in Neural Information Processing Systems*, volume 19, pages 1609–1616, 2007.

[124] Z.-H. Zhou, M.-L. Zhang, S.-J. Huang, and Y.-F. Li. Multi-instance multi-label learning. *Artificial Intelligence*, 176(1):2291–2320, 2012.