Journal of Machine Learning Research 17 (2016) 1-33 Submitted 5/15; Revised 11/15; Published 4/16

Multiplicative Multitask Feature Learning

Xin Wang WANGXIN@ENGR.UCONN.EDU
Department of Computer Science and Engineering
University of Connecticut
Storrs, CT 06279, USA

Jinbo Bi JINBO@ENGR.UCONN.EDU
Department of Computer Science and Engineering
University of Connecticut
Storrs, CT 06279, USA

Shipeng Yu SHIPENG.YU@SIEMENS.COM
Health Services Innovation Center
Siemens Healthcare
Malvern, PA 19355, USA

Jiangwen Sun JAVON@ENGR.UCONN.EDU
Department of Computer Science and Engineering
University of Connecticut
Storrs, CT 06279, USA

Minghu Song MINGHU.SONG@PFIZER.COM
Worldwide Research and Development
Pfizer Inc.
Groton, CT 06340, USA

Editor: Urun Dogan, Marius Kloft, Francesco Orabona, and Tatiana Tommasi

Abstract

We investigate a general framework of multiplicative multitask feature learning which decomposes individual 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 can be proved to be special cases of our 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 effects 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. An efficient blockwise coordinate descent algorithm is developed suitable for solving the entire family of formulations with rigorous convergence analysis. Simulation studies have identified the statistical properties of data that would be in favor of the new formulations. Extensive empirical studies on various classification and regression benchmark data sets 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.

Keywords: Multitask learning, regularization, sparse modeling, blockwise coordinate descent

1. Introduction

Multitask 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, 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 scientific fields, such as robotics (Zhang and Yeung, 2010), natural language processing (Ando and Zhang, 2005), computer aided diagnosis (Bi et al., 2008), and computer vision (Kang et al., 2011).

Research efforts have been devoted to various MultiTask Feature Learning (MTFL) algorithms. One direction of these works directly learns the dependencies among tasks, either by modeling the correlated regression or classification noise (Greene, 2002), or assuming that the model parameters share a common prior (Yu et al., 2005; Lee et al., 2007; Xue et al., 2007; Yu et al., 2007; Jacob et al., 2008), or by examining the tasks' covariance matrix (Bonilla et al., 2007; Zhang and Yeung, 2010; Guo et al., 2011; Yang et al., 2013). 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, Rai and Daume (2010) project the task parameters to explore the latent common substructure. Kang et al. (2011) 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, Passos et al. (2012) assume that subsets of features may be shared only between tasks in the same cluster. Kumar and Daume III (2012) allow overlapping of tasks in different groups by having several bases in common. Maurer et al. (2013) exploit a dictionary allowing sparse representations of the tasks. Gong et al. (2012) 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 (Meier et al., 2008; Zhao et al., 2009) 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 (Lee et al., 2010; Liu et al., 2009a; Obozinski and Taskar, 2006; Zhang et al., 2010; Zhou et al., 2010) 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 (Obozinski and Taskar, 2006; Evgeniou and Pontil, 2004) and ∞ (Turlach et al., 2005). Effective algorithms have since then been developed for the $\ell_{1,2}$ (Liu et al., 2009a) and $\ell_{1,\infty}$ (Quattoni et al., 2009) 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 (Zhang et al., 2010). 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 (Gong et al., 2013). To address this problem, non-convex regularizers, such as the capped-ℓ_1, ℓ_1 regularizer (Gong et al., 2013), have been proposed for the multitask 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 (Jalali et al., 2010; Chen et al., 2011; Gong et al., 2012) or multiplication (Xiong et al., 2006; Bi et al., 2008; Lozano and Swirszcz, 2012) 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 (Jalali et al., 2010) employs $\ell_{1,1}$ and $\ell_{1,\infty}$ regularizers to the two components. A robust MTFL method in (Chen et al., 2011) 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 (Gong et al., 2012). 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 (Bi et al., 2008), or the sparse ℓ_1-norm (i.e., multi-level LASSO (Lozano and Swirszcz, 2012)). The multi-level LASSO method has been analytically compared to the dirty model (Lozano and Swirszcz, 2012), 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 paper 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 (Bi et al., 2008; Lozano and Swirszcz, 2012).

- 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 (Bi et al., 2008; Lozano and Swirszcz, 2012) 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 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 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 1 for multiplicative MTFL .

Table 1: The regularization terms used in various MTFL methods.

	Model	Methods	Norms
Joint regularization models	$\mathbf{A}$	Evgeniou and Pontil (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. (2012)	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. (2012)	$\ell_{1,2}$ 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

The rest of the paper is organized as follows. Section 2 defines the mathematical notation and introduces the proposed models in detail. Section 3 discusses several important theoretical properties of the proposed models including equivalence analysis. Section 4 provides the statistical justification of the multiplicative MTFL models. In Section 5, we develop an efficient algorithm to solve the optimization problems in the proposed models with a convergence analysis. Section 6 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 data sets are also described in Section 6. In Section 7, we conclude this work.

2. 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 data set 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

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

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 (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 $\boldsymbol{\beta}_{t:t=1,\cdots,T}$:

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

where $L(\mathbf{c}, \boldsymbol{\beta}_t, \mathbf{X}_t, \mathbf{y}_t)$ is the same loss function used in Eq.(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 (2) becomes the formulation used in Bi et al. (2008). 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 (2) becomes the formulation used in Lozano and Swirszcz (2012), 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 Bi et al. (2008), 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 Bi et al. (2008); Lozano and Swirszcz (2012) 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 (2), then we obtain the following optimization problem

$$\min_{\boldsymbol{\beta}_t, \mathbf{c} \geq 0} \quad \sum_{t=1}^{T} L(\mathbf{c}, \boldsymbol{\beta}_t, \mathbf{X}_t, \mathbf{y}_t) + \gamma_1 \sum_{t=1}^{T} ||\boldsymbol{\beta}_t||_2^2 + \gamma_2 ||\mathbf{c}||_1. \tag{3}$$

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 (2).

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

$$\min_{\boldsymbol{\beta}_t, \mathbf{c} \geq 0} \quad \sum_{t=1}^{T} L(\mathbf{c}, \boldsymbol{\beta}_t, \mathbf{X}_t, \mathbf{y}_t) + \gamma_1 \sum_{t=1}^{T} ||\boldsymbol{\beta}_t||_1 + \gamma_2 ||\mathbf{c}||_2^2. \tag{4}$$

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.

3. Theoretical Analysis

We first extend the formulation (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 $\boldsymbol{\alpha}_{t:t=1,\cdots,T}$:

$$\min_{\boldsymbol{\alpha}_t} \quad \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \lambda \sum_{j=1}^{d} \sqrt{||\boldsymbol{\alpha}^j||_p^{p/q}}. \tag{5}$$

Our main results of this paper include (i) a theorem that establishes the equivalence between the models derived from solving Problem (2) and Problem (5) for properly chosen values of λ, q, k, γ_1 and γ_2; (ii) a theorem that delineates the conditions for (2) to impose a convex (or concave) regularizer on the model parameter matrix $\mathbf{A}$; and (iii) an analytical solution of Problem (2) for $\mathbf{c}$ which shows how the sparsity of the across-task component is relative to the sparsity of task-specific components.

Theorem 1 (Main Result 1) *Let $\hat{\boldsymbol{\alpha}}_t$ be the optimal solution to Problem (5) and ($\hat{\boldsymbol{\beta}}_t$, $\hat{\mathbf{c}}$) be the optimal solution to Problem (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 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 (5) also minimizes the following optimization problem:

$$\min_{\boldsymbol{\alpha}_t, \boldsymbol{\sigma} \geq 0} \quad \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \mu_1 \sum_{j=1}^{d} \sigma_j^{-1} ||\boldsymbol{\alpha}^j||_p^{p/q} + \mu_2 \sum_{j=1}^{d} \sigma_j, \tag{6}$$

and the optimal solution of Problem (6) also minimizes Problem (5) when proper values of λ, μ_1 and μ_2 are chosen. The second theorem connects Problem (6) to the proposed formulation (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 $\mathrm{diag}(\hat{\mathbf{c}})\hat{\boldsymbol{\beta}}_t$ from the optimal $\hat{\boldsymbol{\beta}}_t$. ■

Note that when $p = 2$ and $q = 1$, the intermediate problem (6) uses a similar regularizer to that in Micchelli et al. (2013) where $\frac{|\alpha^j|^2}{\sigma_j} + \sigma_j$ is used to approximate $|\alpha^j|$ in the ℓ_1-norm regularizer. Problem (6) extends the discussion in Micchelli et al. (2013) to include more general regularizers according to p and q.

Lemma 2 *For any given $\boldsymbol{\alpha}_{t:t=1,\cdots,T}$, Problem (6) can be optimized with respect to $\boldsymbol{\sigma}$ by the following analytical solution*

$$\sigma_j = \mu_1^{\frac{1}{2}} \mu_2^{-\frac{1}{2}} \sqrt{||\boldsymbol{\alpha}^j||_p^{p/q}}. \tag{7}$$

Proof By the Cauchy-Schwarz inequality, we can derive a lower bound for the sum of the two regularizers in Problem (6) as follows:

$$\mu_1 \sum_{j=1}^{d} \sigma_j^{-1} ||\boldsymbol{\alpha}^j||_p^{p/q} + \mu_2 \sum_{j=1}^{d} \sigma_j \geq 2\sqrt{\mu_1 \mu_2} \sum_{j=1}^{d} \sqrt{||\boldsymbol{\alpha}^j||_p^{p/q}}, \tag{8}$$

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 (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.(8). Because the lower bound (the right hand side of Eq.(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 (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.(7). ■

Remark 3 *Based on the proof of Lemma 2, we also know that the objective function of (5) is the lower bound of the objective function of (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 (7). Hence, we can also conclude that if $(\hat{\boldsymbol{\alpha}}_t, \hat{\boldsymbol{\sigma}})$ is the optimal solution of Problem (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 *Let $\alpha_j^t = c_j \beta_j^t$ for all t and j. Replacing $\boldsymbol{\beta}_t$ by $\boldsymbol{\alpha}_t$ in Problem (2) yields the following optimization problem*

$$\min_{\boldsymbol{\alpha}_t, \mathbf{c} \geq 0} \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \gamma_1 \sum_{j=1}^{d} c_j^{-p} ||\boldsymbol{\alpha}^j||_p^p + \gamma_2 \sum_{j=1}^{d} (c_j)^k. \tag{9}$$

For any given $\boldsymbol{\alpha}_{t:t=1,\cdots,T}$, Problem (9) can be optimized with respect to $\mathbf{c}$ by the following analytical solution

$$c_j = (\gamma_1 \gamma_2^{-1} \sum_{t=1}^{T} (\alpha_j^t)^p)^{\frac{1}{p+k}}. \tag{10}$$

Proof This lemma can be proved following a similar argument in the proof of Lemma 2. By the Cauchy-Schwarz inequality, the sum of the two regularizers in (9) satisfies the following inequality

$$\gamma_1 \sum_{j=1}^{d} c_j^{-p} ||\boldsymbol{\alpha}^j||_p^p + \gamma_2 \sum_{j=1}^{d} c_j^k \geq 2\gamma_1^{\frac{k}{p+k}} \gamma_2^{\frac{p}{p+k}} \sum_{j=1}^{d} (||\boldsymbol{\alpha}^j||_p^p)^{\frac{k}{p+k}}, \tag{11}$$

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 2, we will prove that Problem (5) is equivalent to Problem (6) in the sense that an optimal solution of Problem (5) is also an optimal solution of Problem (6) and vice versa when λ, μ_1, and μ_2 satisfy certain conditions.

Theorem 5 *The solution sets of Problem (5) and Problem (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 (5), we show that the pair $(\hat{\mathbf{A}}, \hat{\boldsymbol{\sigma}})$ minimizes Problem (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 3, the objective function of (5) is the lower bound of the objective function of (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 (6) reaches the lower bound. In this case, Problem (5) at $\hat{\mathbf{A}}$ and Problem (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 (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 2, 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 (6), at $\tilde{\mathbf{A}}$, reaches the lower bound which is formulated as the objective of Problem (5)

when $\lambda = 2\sqrt{\mu_1\mu_2}$. In other words, the objective values of (6) and (5) are identical at $\tilde{\mathbf{A}}$. Hence, $\tilde{\mathbf{A}}$ achieves a lower objective value than that of $\hat{\mathbf{A}}$ for Problem (5), contradicting to the optimality of $\hat{\mathbf{A}}$.

Second, if $(\hat{\mathbf{A}}, \hat{\boldsymbol{\sigma}})$ minimizes Problem (6), we show that $\hat{\mathbf{A}}$ minimizes Problem (5).

Suppose that $\hat{\mathbf{A}}$ does not minimize Problem (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 (6) as proved in the first paragraph, and will bring the objective function of Problem (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 (5) and (6) have identical solution sets when $\lambda = 2\sqrt{\mu_1\mu_2}$. ■

In order to link Problem (6) to our formulation (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 (6) based on Lemmas 2 and 4.

Theorem 6 *The optimal solution $(\hat{\mathbf{A}}, \hat{\boldsymbol{\sigma}})$ of Problem (6) is equivalent to the optimal solution $(\hat{\mathbf{B}}, \hat{\mathbf{c}})$ of Problem (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 (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 (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 (6)), we know that $\hat{\boldsymbol{\beta}}_t$ and $\hat{\mathbf{c}}$ minimize the following objective function

$$J(\hat{\beta}_j^t, \hat{c}_j) = \sum_{t=1}^{T} L(\hat{\mathbf{c}}, \hat{\boldsymbol{\beta}}_t, \mathbf{X}_t, \mathbf{y}_t) + \mu_1 \sum_{j=1}^{d} ||\hat{\boldsymbol{\beta}}^j||_p^{p/q}\hat{c}_j^{(p-kq)/q} + \mu_2 \sum_{j=1}^{d} (\hat{c}_j)^k. \tag{12}$$

By Lemma 2 and Remark 3, 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.(12) yields the same objective function of Problem (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 (2) because otherwise, if any other solution $(\boldsymbol{\beta}, \mathbf{c})$ can further reduce the objective value of Problem (2), then the corresponding $\boldsymbol{\alpha}$ and $\boldsymbol{\sigma}$ will bring the objective function of Problem (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 (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 (6).

Substituting $\hat{\alpha}_j^t$, $\hat{\sigma}_j$ for $\hat{\beta}_j^t$, $\hat{c}_j$ in Problem (2) yields an objective function

$$J(\hat{\alpha}_j^t, \hat{\sigma}_j) = \sum_{t=1}^{T} L(\hat{\boldsymbol{\alpha}}_t, \mathbf{X}_t, \mathbf{y}_t) + \gamma_1 \sum_{j=1}^{d} ||\hat{\boldsymbol{\alpha}}^j||_p^p\hat{\sigma}_j^{-(p/k)} + \gamma_2 \sum_{j=1}^{d} \hat{\sigma}_j. \tag{13}$$

We hence know that $\hat{\alpha}_j^t$ and $\hat{\sigma}_j$ minimize Eq.(13). Similar to the proof of Lemma 4, 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.(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.(13) to the objective function of Problem (6). Thus, $\hat{\alpha}_j^t$ and $\hat{\sigma}_j$ optimize Problem (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 (2) and (5) are equivalent. Solving Problem (2) will yield an optimal solution $\hat{\boldsymbol{\alpha}}$ to Problem (5) and vice versa.

By the equivalence analysis, the proposed framework corresponds to a family of joint regularization methods as defined by Eq.(5). Assuming a convex loss function is used, this family includes some convex formulations when convex regularizers are applied to $\mathbf{A}$ in (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 (5) becomes the standard ℓ_p-norm. Correspondingly, when $k = p/(p-1)$ which is commonly not an integer except $p = 2$, our formulation (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 (5). Combinations of different p and k values will render the models different algorithmic behaviors.

Theorem 7 (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 $\mathbf{A}$:

$$\lambda\sum_{j=1}^{d}\sqrt{||\boldsymbol{\alpha}^j||_p^{p/q}} = \lambda\sum_{j=1}^{d}(||\boldsymbol{\alpha}^j||_p)^{\frac{p}{2q}} = \lambda\sum_{j=1}^{d}(||\boldsymbol{\alpha}^j||_p)^{\frac{kp}{k+q}}. \tag{14}$$

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

$$\min_{\boldsymbol{\beta}_t, \mathbf{c}\geq 0} \sum_{t=1}^{T} L(\mathbf{c}, \boldsymbol{\beta}_t, \mathbf{X}_t, \mathbf{y}_t) + \gamma_1\sum_{t=1}^{T}||\boldsymbol{\beta}_t||_2^2 + \gamma_2||\mathbf{c}||_2^2, \tag{15}$$

which is used in Bi et al. (2008). This problem is equivalent to the following joint regularization method as used in Obozinski and Taskar (2006); Argyriou et al. (2007).

$$\min_{\boldsymbol{\alpha}_t} \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \lambda \sum_{j=1}^{d} \sqrt{\sum_{t=1}^{T} (\alpha_j^t)^2}, \tag{16}$$

when $\lambda = 2\sqrt{\gamma_1\gamma_2}$. Problem (16) uses the so called $\ell_{1,2}$-norm to regularize the matrix $\mathbf{A}$.

Remark 9 *For a particular choice of $p = 1$ and $k = 1$, Problem (2) is formulated as*

$$\min_{\boldsymbol{\beta}_t, \mathbf{c} \geq 0} \sum_{t=1}^{T} L(\mathbf{c}, \boldsymbol{\beta}_t, \mathbf{X}_t, \mathbf{y}_t) + \gamma_1 \sum_{t=1}^{T} ||\boldsymbol{\beta}_t||_1 + \gamma_2 ||\mathbf{c}||_1, \tag{17}$$

which is used in Lozano and Swirszcz (2012). This problem is equivalent to the following joint regularization method

$$\min_{\boldsymbol{\alpha}_t} \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \lambda \sum_{j=1}^{d} \sqrt{\sum_{t=1}^{T} |\alpha_j^t|}, \tag{18}$$

when $\lambda = 2\sqrt{\gamma_1\gamma_2}$. Problem (17) imposes stronger sparsity on $\boldsymbol{\beta}$ and $\mathbf{c}$ (and correspondingly on $\boldsymbol{\alpha}$) than (15), which intends to shrink more model parameters to zero. Problem (18) uses a concave non-matrix-norm regularizer.

Remark 10 *For a particular choice of $p = 2$ and $k = 1$, which corresponds to the new formulation (3). This problem is equivalent to the following joint regularization method*

$$\min_{\boldsymbol{\alpha}_t} \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \lambda \sum_{j=1}^{d} \sqrt[3]{\sum_{t=1}^{T} (\alpha_j^t)^2}, \tag{19}$$

when $\lambda = 2\gamma_1^{\frac{1}{3}}\gamma_2^{\frac{2}{3}}$. Problem (19) uses a concave non-matrix-norm regularizer as well but the concavity is weaker than Problem (18) in the sense that the polynomial order is $\frac{2}{3}$ rather than $\frac{1}{2}$.

The proposed formulation (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 (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 Lozano and Swirszcz (2012) where $p = k = 1$, the new formulation (3) may allow more selected features to be shared across different tasks.

Remark 11 *For a particular choice of $p = 1$ and $k = 2$, which corresponds to the new formulation (4). This problem is equivalent to the following joint regularization method*

$$\min_{\boldsymbol{\alpha}_t} \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \lambda \sum_{j=1}^{d} \sqrt[3]{\left(\sum_{t=1}^{T} |\alpha_j^t|\right)^2}, \tag{20}$$

when $\lambda = 2\gamma_1^{\frac{2}{3}}\gamma_2^{\frac{1}{3}}$. Problem (20) uses another concave non-matrix-norm regularizer that has the similar polynomial order of $\frac{2}{3}$ to Problem (19). In comparison with (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) 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 Lozano and Swirszcz (2012), this model allows more of the features that are only relevant to a few tasks to be selected. In comparison with the model in Bi et al. (2008), 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 12 (Main Result 3) *Let $\hat{\boldsymbol{\beta}}_t$, $t = 1, \cdots, T$, be the optimal solutions of Problem (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,*

$$\hat{c}_j = (\gamma_1/\gamma_2)^{\frac{1}{k}} \sqrt[k]{\sum_{t=1}^{T} (\hat{\beta}_j^t)^p}, \tag{21}$$

for all $j = 1, \cdots, d$, is optimal to Problem (2).

Proof This analytical formula can be directly derived from Theorem 5 and Theorem 6. 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 6, we obtain $\hat{c}_j = \left(\mu_1 \mu_2^{-1} ||\hat{\boldsymbol{\beta}}^j||_p^{p/q}\right)^{\frac{q}{2kq-p}}$. 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 (2) and (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 (21). ■

Remark 13 *For particular choices of p and k, the relation between the optimal $\mathbf{c}$ and $\boldsymbol{\beta}$ can be computed according to Theorem 12. Table 2 summarizes the relation formula for the common choices when $p \in \{1, 2\}$ and $k \in \{1, 2\}$.*

Table 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. 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

$$p(\mathbf{A}|\mathbf{X}, \mathbf{y}, \boldsymbol{\Delta}) \propto p(\mathbf{A}|\boldsymbol{\Delta}) \prod_{t=1}^{T} p(\mathbf{y}_t|\mathbf{X}_t, \boldsymbol{\alpha}_t). \tag{22}$$

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

$$p(z) = \frac{1}{2\rho\Gamma(1+1/p)} \exp\left(-\frac{|z-\mu|^p}{\rho^p}\right), \tag{23}$$

in which $\rho > 0$, $p > 0$, and $\Gamma(\cdot)$ is the Gamma function (Goodman and Kotz, 1973). 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 Zhang et al. (2010) for a similar treatment)

$$p(\mathbf{A}|\boldsymbol{\Delta}) \propto \prod_{j=1}^{d}\prod_{t=1}^{T} \frac{1}{\delta_j} \exp\Big(-\frac{|\alpha_j^t|^p}{\delta_j^p}\Big) = \prod_{j=1}^{d} \frac{1}{\delta_j^T} \exp\Big(-\frac{\|\boldsymbol{\alpha}^j\|_p^p}{\delta_j^p}\Big), \tag{24}$$

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:

$$\min_{\mathbf{A},\boldsymbol{\Delta}} \quad J = \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \sum_{j=1}^{d} \Big(\frac{\|\boldsymbol{\alpha}^j\|_p^p}{\delta_j^p} + T\ln\delta_j\Big). \tag{25}$$

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

$$\delta_j = \Big(\frac{p}{T}\Big)^{1/p} \|\boldsymbol{\alpha}^j\|_p. \tag{26}$$

Bringing this back to (25), we have the following equivalent problem:

$$\min_{\mathbf{A}} \quad J = \sum\nolimits_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + T \sum\nolimits_{j=1}^{d} \ln \|\boldsymbol{\alpha}^j\|_p. \tag{27}$$

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 (27),

$$\min_{\mathbf{A}} \; J_1 = \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + T \sum_{j=1}^{d} |c_j| + T \sum_{j=1}^{d} \sum_{t=1}^{T} |\beta_j^t| - 2dT, \tag{28}$$

which is equivalent to the multiplicative formulation (2) where $\{p, k\} = \{1, 1\}$. This proves the following theorem:

Theorem 14 *When $\{p, k\} = \{1, 1\}$, optimizing the multiplicative formulation* (2) *is equivalent to maximizing a lower bound of the MAP solution under probabilistic model* (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 (27). By minimizing the upper bound, the problem is formulated as:

$$\min_{\mathbf{A}} \quad J_{p,k} = \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \frac{T}{k}\|\mathbf{c}\|_k^k + \frac{T}{p}\sum_{t=1}^{T}\|\boldsymbol{\beta}_t\|_p^p, \tag{29}$$

which is equivalent to the general multiplicative formulation in (2). Therefore we have proved the following theorem:

Theorem 15 *Optimizing the multiplicative formulation* (2)*, in the form of* (29)*, is equivalent to maximizing a lower bound of the MAP solution under probabilistic model* (22) *with* $p \in (1, \infty)$ *in the prior definition.*

5. Optimization Algorithm

Alternating optimization algorithms have been used in both of the early methods (Bi et al., 2008; Lozano and Swirszcz, 2012) to solve Problem (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 Bi et al. (2008); Lozano and Swirszcz (2012) 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 and the property that both Problems (2) and (5) are equivalent to the intermediate Problem (6) (or Problem (9)). In the algorithm to solve Problem (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.(10) in Lemma 4. The overall procedure is summarized in **Algorithm 1**. 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 (2) and Problem (9) are equivalent. For notational convenience, we denote the objective function of Problem (2) by $g(\mathbf{B}, \mathbf{c})$ that takes inputs $\boldsymbol{\beta}_t$ and $\mathbf{c}$. We denote the objective function of Problem (9) by $f(\mathbf{A}, \mathbf{c})$. Both objective functions comprise the sum of three parts. For instance, f can be written as follows:

$$f(\mathbf{A}, \mathbf{c}) = f_0(\mathbf{A}, \mathbf{c}) + f_{\mathbf{A}}(\mathbf{A}) + f_{\mathbf{c}}(\mathbf{c}), \tag{31}$$

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 (9). Similar to what has been defined in Tseng (2001), 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

$$\lim_{\lambda \to 0}[f(\mathbf{z} + \lambda\mathbf{b}) - f(\mathbf{z})]/\lambda \geq 0, \quad \forall \mathbf{b} \quad \text{such that} \quad (\mathbf{z} + \lambda\mathbf{b}) \in dom\ f. \tag{32}$$

Algorithm 1 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 $\boldsymbol{\beta}_t^s$

$$\min_{\boldsymbol{\beta}_t} L(\boldsymbol{\beta}_t, \tilde{\mathbf{X}}_t, \mathbf{y}_t) + \gamma_1 ||\boldsymbol{\beta}_t||_p^p \tag{30}$$

  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.(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 $\boldsymbol{\beta}_t$, $t = 1, \cdots, T$

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 $\lambda \leq \epsilon$,

$$f(\mathbf{z} + \lambda(0, \cdots, \mathbf{b}_k, \cdots, 0)) \geq f(\mathbf{z}), \tag{33}$$

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 1, 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 Tseng (2001), if f is regular at an accumulation point $\mathbf{z}^*$, this $\mathbf{z}^*$ is a stationary point.

Lemma 16 *Let the sequence of iterates generated by Algorithm 1 be* $\mathbf{z}^s = \{\mathbf{A}^s, \mathbf{c}^s\}_{s=1,2,\cdots}$ *and the function* f *is defined by (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.(10). According to Lemma 4, $\mathbf{c}^{s+1}$ 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 1, we obtain $\mathbf{B}^{s+1}$ by solving Problem (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 16, 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 17 *The sequence of iterates generated by Algorithm 1, $\mathbf{z}^s = \{\mathbf{A}^s, \mathbf{c}^s\}_{s=1,2,\cdots}$, (or equivalently, $\{\mathbf{B}^s, \mathbf{c}^s\}_{s=1,2,\cdots}$) is bounded.*

Proof According to Lemma 16, 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 18 *The sequence $\mathbf{z}^s = \{\mathbf{A}^s, \mathbf{c}^s\}_{s=1,2,\cdots}$ generated by Algorithm 1 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 17, 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 (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 (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

$$f(\mathbf{z}^* + \lambda(0, \cdots, \mathbf{b}_t, \cdots, 0)) \geq f(\mathbf{z}^*), \tag{34}$$

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

Theorem 19 *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, we know that the optimal solution of Problem (9) can only occur when $\mathbf{c}$ and $\boldsymbol{\alpha}_t$ satisfy Eq.(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.(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 Tseng (2001) that when each subproblem has a unique minimum, which is the case in our algorithm because Subproblem (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 1. Subproblem (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 1 only linearly increases with the number of tasks. Due to the nature of Algorithm 1, 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.

6. Experiments

We empirically evaluated the performance of the multiplicative MTFL algorithms on both synthetic data sets and a variety of real-world data sets, 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 1 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 2. Although when the values of (p, k) were (2,2) and (1,1), the two models corresponded respectively to the same methods in Bi et al. (2008) and Lozano and Swirszcz (2012), they were solved differently from prior methods using our Algorithm 1 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) (Jalali et al., 2010) and the robust MTFL (rMTFL) (Gong et al., 2012). 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 (Jalali et al., 2010) : The dirty model with regularizers $||\mathbf{P}||_{1,1}$ and $||\mathbf{Q}||_{1,\infty}$, where $\mathbf{A} = \mathbf{P} + \mathbf{Q}$.
- rMTFL (Gong et al., 2012) : Robust multitask feature learning with the regularizers $||\mathbf{P}||_{1,2}$ and $||\mathbf{Q}^T||_{1,2}$, where $\mathbf{A} = \mathbf{P} + \mathbf{Q}$.
- MMTFL$\{2, 2\}$ (Bi et al., 2008) : Multiplicative multitask feature learning with regularizers $||\mathbf{B}||_F^2$ and $||\mathbf{c}||_2^2$, where $\mathbf{A} = \text{diag}(\mathbf{c})\mathbf{B}$.
- MMTFL$\{1, 1\}$ (Lozano and Swirszcz, 2012) : Multiplicative multitask 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 multitask 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 multitask 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 data set 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.

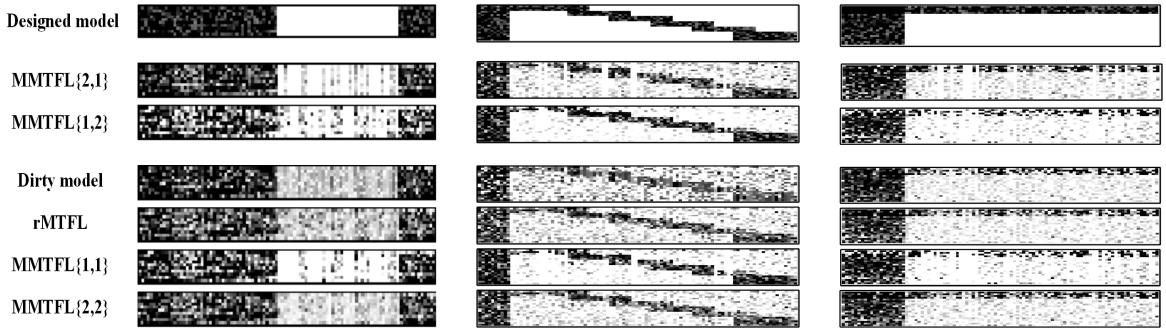
6.1 Simulation Studies

We created three categories of data sets, all of which were designed for regression experiments to evaluate the behaviors of the different methods. The data sets 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 data sets, 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 (Gong et al., 2012).

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 1 for better illustration.

6.1.1 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 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-



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

Figure 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.

based regularizers. We hypothesized that the proposed new formulation (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 1b(top), where we once 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) would produce better regression performance than the other models on this data set.

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 data sets 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 data set where 200 examples, each represented

Table 3: Comparison of the test R^2 values obtained by the different MTFL methods on synthetic data sets 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

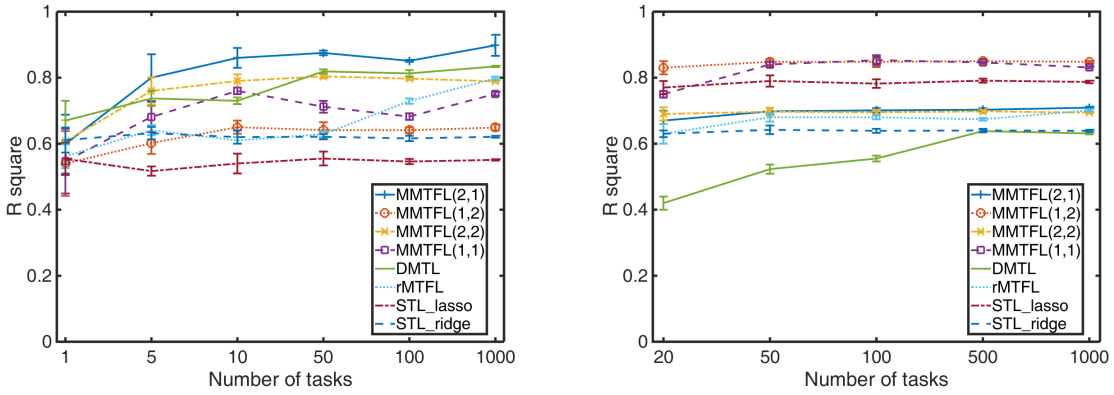
by 100 features, were generated for each of 20 tasks. The parameter matrix $\mathbf{A} = \mathbf{P} + \mathbf{Q}$ where 80 rows in $\mathbf{P}$ and 16 columns in $\mathbf{Q}$ were set to 0. The component $\mathbf{P}$ was used to indicate the subset of relevant features across all tasks, and the component $\mathbf{Q}$ was used to tell that there were outlier tasks that did not share features with other tasks. Given this simulation process, this data set would be in favor of the rMTFL model proposed in Gong et al. (2012). The designed model parameter matrix was shown in Figure 1c(top).

6.1.2 Performance on synthetic data sets

We first compared the regression performance of the different methods on the three categories of data sets. Table 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 3 were obtained on synthetic data sets that had 20 tasks with 200 examples and 100 features for each task. We reported the test R^2 obtained on each data set when 25%, 33% and 50% of the data were used in training. From Table 3, we observe that the proposed formulation (3)(MMTFL(2,1)) consistently outperformed other models on *C1* data sets, whereas the proposed model (4)(MMTFL(1,2)) consistently outperformed on *C2* data sets. 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* data set. 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 1. From the results on the *C1* and *C2* data sets, 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



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

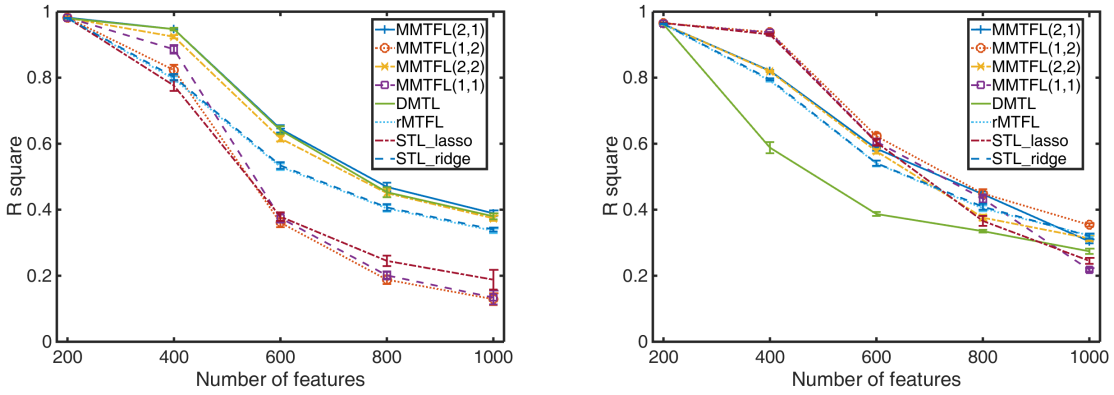
Figure 2: The regression performance of different models on synthetic data *C1* and *C2* when the number of tasks is varied.

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 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 multitask learning that cluster tasks into groups (Kang et al., 2011; Jacob et al., 2008; Zhou et al., 2011) because no cluster structure is present in Figure 1b. Rather, the sharing pattern in Figure 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 1c shows the results on the *C3* data set. 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 data set. 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 data set, 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 2 compares the performance of different methods when we vary the number of tasks in the *C1* and *C2* categories. On each data set, we used 33% of the data in training



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

Figure 3: The regression performance of different models on synthetic data *C1* and *C2* when the number of features is varied.

with 15 trials, and reported here the average R^2 values and standard deviation bars. From Figure 2, MMTFL(2,1) constantly performed the best among all methods on the *C1* data sets (but not in the single task learning) whereas MMTFL(1,2) outperformed the other models on the *C2* data sets. 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 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 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* data sets. This figure also shows that MMTFL(1,1) performed well on the *C2* data sets but much worse on the *C1* data sets. DMTL produced good performance, close to that of MMTFL(2,1), on the *C1* data sets.

6.2 Experiments with Benchmark Data

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

6.2.1 Benchmark Datasets

Sarcos (Argyriou et al., 2007): 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 (Bi et al., 2013): 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 data set 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 (Ma et al., 2015): The quantitative structure-activity relationship (QSAR) methods are commonly used to predict biological activities of chemical compounds in the field of drug discovery. The data sets 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. (Lucas et al., 2013): The Climate Model Simulation Crashes (C.M.S.C.) data set contained records of simulated crashes encountered during climate model uncertainty quantification ensembles. The data set 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 (Xue et al., 2007): The original Landmine data contained 29 data sets 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 data set could be used to build a binary classifier. We used the data sets 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 (Maurer et al., 2013): This data set was composed of binary 20×16 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 data set. 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 (Liu et al., 2009b): This data set 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 data set 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 (Kang et al., 2011): This data set 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 (Qazi et al., 2007; Bi and Wang, 2015): The heart wall motion abnormality (HWMA) detection data set 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 segment. 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.

6.2.2 Performance on real world data sets

Three real-world data sets, the Sarcos, CollegeDrinking and QSAR, were used in regression experiments. The performance of the different methods is summarized in Table 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 data set (in all of the 3 trials) and the CollegeDrinking data set (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 data set. MMTFL(1,2) models achieved the best performance on the QSAR data set consistently across all the 3 trial settings. On this data set, 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 data sets.

The other seven real-world data sets were used in classification experiments. Table 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 data sets 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 data sets may prefer across-task sparse models, indicating that many irrelevant features may exist in the data. For the remaining five data sets 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%.

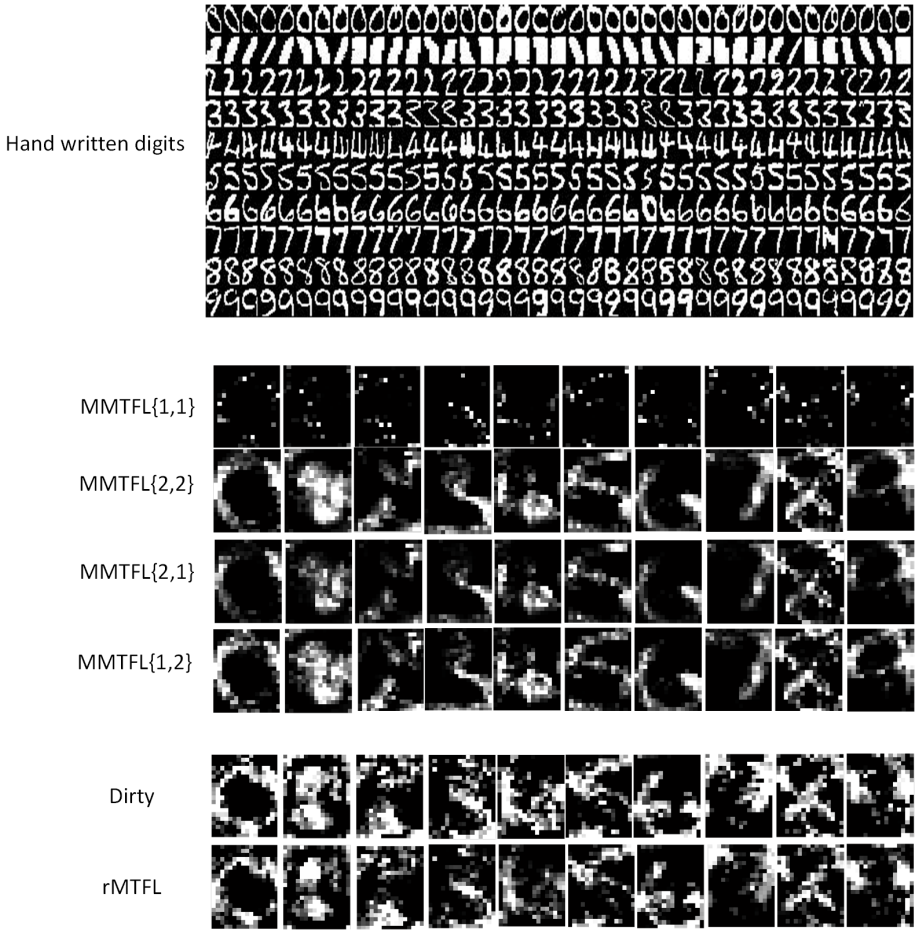


Figure 4: The models constructed by MTFL methods for individual digits using the pixels in hand written digit images. The models can be re-organized back into images. The pixel in the above images ranges from 0 to 1. The lighter, the closer to 1 for lucid illustration.

In particular, for the Alphadigits data set, 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 compares the constructed models for each digit by each of the six MTFL methods. In the top of Figure 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.

Table 4: Comparison of the R^2 values of the different MTFL methods on the benchmark regression data sets.

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 5: Comparison of the F1 scores of the different MTFL methods on the benchmark classification data sets.

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**

7. Conclusion

In this paper, we have studied a general framework of multiplicative multitask 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 multitask 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 data sets also confirm the observation and demonstrate the advantages of the proposed two formulations over several existing methods.

Acknowledgments

This work was supported by NSF grants IIS-1320586, DBI-1356655 and the NIH grant R01DA037349. Jinbo Bi was also supported by NSF grants CCF-1514357, IIS-1407205 and IIS-1447711. Correspondence should be addressed to Jinbo Bi (jinbo@engr.uconn.edu).

References

Rie Kubota Ando and Tong Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853, December 2005.

Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Multi-task feature learning. In *Advances in Neural Information Processing Systems*, pages 41–48. 2007.

Jinbo Bi and Xin Wang. Learning classifiers from dual annotation ambiguity via a minmax framework. *Neurocomputing*, 151, Part 2(0):891 – 904, 2015.

Jinbo Bi, Tao Xiong, Shipeng Yu, Murat Dundar, and R. Bharat Rao. An improved multi-task learning approach with applications in medical diagnosis. In *Proceedings of the 2008 European Conference on Machine Learning and Knowledge Discovery in Databases - Part I*, pages 117–132, 2008.

Jinbo Bi, Jiangwen Sun, Yu Wu, Howard Tennen, and Stephen 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.

Edwin Bonilla, Kian Ming Chai, and Christopher Williams. Multi-task gaussian process prediction. In *Advances in neural information processing systems*, pages 153–160, 2007.

Jianhui Chen, Jiayu Zhou, and Jieping Ye. Integrating low-rank and group-sparse structures for robust multi-task learning. In *Proceedings of ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 42–50, 2011.

Theodoros Evgeniou and Massimiliano Pontil. Regularized multi–task learning. In *Proceedings of ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 109–117, 2004.

Pinghua Gong, Jieping Ye, and Changshui Zhang. Robust multi-task feature learning. In *Proceedings of ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 895–903, 2012.

Pinghua Gong, Jieping Ye, and Changshui Zhang. Multi-stage multi-task feature learning. *The Journal of Machine Learning Research*, 14(1):2979–3010, 2013.

Irwin R Goodman and Samuel Kotz. Multivariate θ-generalized normal distributions. *Journal of Multivariate Analysis*, 3(2):204–219, 1973.

William H. Greene. *Econometric Analysis*. Prentice Hall, fifth edition, 2002.

Shengbo Guo, Onno Zoeter, and Cédric Archambeau. Sparse bayesian multi-task learning. In *Advances in Neural Information Processing Systems*, pages 1755–1763, 2011.

Laurent Jacob, Bach Francis, and Jean-Philippe Vert. Clustered multi-task learning: a convex formulation. In *Advances in Neural Information Processing Systems*, pages 745–752, 2008.

Ali Jalali, Sujay Sanghavi, Chao Ruan, and Pradeep K Ravikumar. A dirty model for multi-task learning. In *Advances in Neural Information Processing Systems*, pages 964–972, 2010.

Zhuoliang Kang, Kristen Grauman, and Fei Sha. Learning with whom to share in multi-task feature learning. In *Proceedings of the International Conference on Machine Learning*, pages 521–528, 2011.

Abhishek Kumar and Hal Daume III. Learning task grouping and overlap in multi-task learning. In *Proceedings of the International Conference on Machine Learning*, pages 1383–1390, 2012.

Seunghak Lee, Jun Zhu, and Eric Xing. Adaptive multi-task lasso: with application to eQTL detection. In *Advances in Neural Information Processing Systems*, pages 1306–1314. 2010.

Su-In Lee, Vassil Chatalbashev, David Vickrey, and Daphne Koller. Learning a meta-level prior for feature relevance from multiple related tasks. In *Proceedings of the 24th International Conference on Machine Learning*, pages 489–496, New York, NY, USA, 2007.

Jun Liu, Shuiwang Ji, and Jieping 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, 2009a.

Qiuhua Liu, Xuejun Liao, Hui Li, Jason R Stack, and Lawrence Carin. Semisupervised multitask learning. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(6):1074–1086, 2009b.

Aurelie Lozano and Grzegorz Swirszcz. Multi-level lasso for sparse multi-task regression. In *Proceedings of the International Conference on Machine Learning*, pages 361–368, 2012.

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.

Junshui Ma, Robert P. Sheridan, Andy Liaw, George E. Dahl, and Vladimir Svetnik. Deep neural nets as a method for quantitative structureactivity relationships. *Journal of Chemical Information and Modeling*, 55(2):263–274, 2015.

Andreas Maurer, Massimiliano Pontil, and Bernardino Romera-Paredes. Sparse coding for multitask and transfer learning. In *Proceedings of the 30th International Conference on Machine Learning*, pages 343–351, 2013.

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.

Charles A Micchelli, Jean M Morales, and Massimiliano Pontil. Regularizers for structured sparsity. *Advances in Computational Mathematics*, 38(3):455–489, 2013.

Guillaume Obozinski and Ben Taskar. Multi-task feature selection. In *Technical report, Statistics Department, UC Berkeley*, 2006.

Alexandre Passos, Piyush Rai, Jacques Wainer, and Hal Daume III. Flexible modeling of latent task structures in multitask learning. In *Proceedings of the International Conference on Machine Learning*, pages 1103–1110, 2012.

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.

Ariadna Quattoni, Xavier Carreras, Michael Collins, and Trevor Darrell. An efficient projection for l1,infinity regularization. In *Proceedings of the International Conference on Machine Learning*, pages 108–115, 2009.

Piyush Rai and Hal Daume. Infinite predictor subspace models for multitask learning. In *International Conference on Artificial Intelligence and Statistics*, pages 613–620, 2010.

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

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

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*, pages 69–76, 2006.

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.

Ming Yang, Yingming Li, and Zhongfei Mark 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.

Kai Yu, Volker Tresp, and Anton Schwaighofer. Learning gaussian processes from multiple tasks. In *Proceedings of the 22Nd International Conference on Machine Learning*, pages 1012–1019, New York, NY, USA, 2005.

Shipeng Yu, Volker Tresp, and Kai Yu. Robust multi-task learning with t-processes. In *Proceedings of the 24th International Conference on Machine Learning*, pages 1103–1110, 2007.

Yu Zhang and Dit-Yan Yeung. A convex formulation for learning task relationship in multi-task learning. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 733–742, 2010.

Yu Zhang, Dit-Yan Yeung, and Qian Xu. Probabilistic multi-task feature selection. In *Advances in Neural Information Processing Systems*, pages 2559–2567, 2010.

Peng Zhao, Guilherme Rocha, and Bin Yu. The composite absolute penalties family for grouped and hierarchical variable selection. *The Annals of Statistics*, pages 3468–3497, 2009.

Jiayu Zhou, Jianhui Chen, and Jieping Ye. Clustered multi-task learning via alternating structure optimization. In *Advances in Neural Information Processing Systems*, pages 702–710, 2011.

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