Multi-Task Learning with Prior Information

Mengyuan Zhang[*] Kai Liu[†]

Abstract

Multi-task learning aims to boost the generalization performance of multiple related tasks simultaneously by leveraging information contained in those tasks. In this paper, we propose a multi-task learning framework, where we utilize prior knowledge about the relations between features. We also impose a penalty on the coefficients changing for each specific feature to ensure related tasks have similar coefficients on common features shared among them. In addition, we capture a common set of features via group sparsity. The objective is formulated as a non-smooth convex optimization problem, which can be solved with various methods, including gradient descent method with fixed stepsize, iterative shrinkage-thresholding algorithm (ISTA) with back-tracking, and its variation – fast iterative shrinkage-thresholding algorithm (FISTA). In light of the sub-linear convergence rate of the methods aforementioned, we propose an asymptotically linear convergent algorithm with a theoretical guarantee. Empirical experiments on both regression and classification tasks with real-world datasets demonstrate that our proposed algorithms are capable of improving the generalization performance of multiple related tasks.

1 Introduction

Over past decades, Multi-Task Learning (MTL) [4] has attracted great interest and has been applied successfully in various research areas, including computer vision [22], health informatics [17], and natural language processing [8], etc. MTL aims to boost the generalization performance of multiple related tasks simultaneously by leveraging potentially useful information contained in those related tasks. Generally speaking, the objective of a MTL problem can be formulated as:

$$\min_{\mathbf{W}} \sum_{i=1}^{m} \|\mathbf{X}_i \mathbf{w}_i - \mathbf{y}_i\|^2, \tag{1.1}$$

where $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_m]$ is the matrix formed by the weight vectors of m tasks, $\mathbf{X}_i$ denotes the input feature matrix for the i-th task, y_i is the corresponding response. Based on the above objective with the goal of minimizing the overall error across all the tasks, previous studies have been done to improve the performance by identifying the intrinsic relationships among tasks [25], such as a common set of features they share, and some outlier tasks which are in fact not related with other tasks. Existing methods focusing on finding common feature sets can be classified into two major categories: explicit parameter sharing, where all the tasks share some common coefficients explicitly [1, 2], and implicit structure sharing, which captures the shared structure among tasks implicitly by constraining a common low-rank subspace [19, 21]. Some studies also perform outlier task detection [11, 13] at the same time.

One key observation which is ignored by previous studies is: the prior knowledge regarding different features is not taken into account, which can play an important role in feature selection. For instance, expertise knowledge may indicate two features have a similar influence on the response, therefore the correspondent coefficients should be close to each other, while two features having opposite influences should have significantly different coefficients. Besides that, for common features in related tasks, coefficients for the same feature should not change dramatically across tasks. Such prior information is reasonable and not difficult to obtain when we deal with real-world data where each feature denotes a certain property of the data and relationships between different features can be inferred without much cost.

Another shortcoming of previous MTL studies targeting optimizing Eq.(1.1) is the slow convergence rate. In short, gradient descent requires $\mathcal{O}(\frac{1}{\epsilon})$ iterations to achieve ϵ accuracy, while momentum based methods will not exceed $\mathcal{O}(\frac{1}{\sqrt{\epsilon}})$, which is still sub-linear convergence rate. To accelerate the convergence asymptotically and in light of the objective by adding the regularization terms described above, we propose a novel updating algorithm that enjoys a linear convergence rate with $\mathcal{O}(log(\frac{1}{\epsilon}))$ iterations. One can see its advantage over its counterparts when ϵ becomes sufficiently small in terms of fewer iterations to obtain ϵ precision.

We explicitly list the main contributions of this paper as:

- We add a regularization term based on prior information to obtain more accurate coefficients

[*]mengyuz@clemson.edu, School of Computing, Clemson University.

[†]kail@clemson.edu, School of Computing, Clemson University.

for related tasks. We impose an extra constraint on the coefficients' changing for the same features among related tasks, which will lead the coefficients for the same features to change smoothly across similar tasks.

- We present three methods to optimize the MTL with prior information objective, including the vanilla gradient descent with a fixed stepsize, the iterative shrinkage thresholding algorithm (ISTA) with modified stepsize searching [3, 14], and a novel algorithm with linear convergence rate, which is proved to have comparable speed with the fast iterative shrinkage thresholding algorithm (FISTA) with backtracking [3, 5] in experiments. We also provide the linear convergence rate proof of our proposed algorithm, validating the superiority of our new algorithm in terms of speed accelerating in the setting of MTL with prior information.

- We conduct empirical experiments with real-world data, the experiment results demonstrate the effectiveness of our proposed algorithm.

The remaining of this paper is organized as follows: In Section 2 we introduce the problem formulation of our proposed MTL with prior information. In Section 3, we present the three methods to solve the optimization problem we formulated. Detailed proof of the convergence rate of our proposed algorithm in the MTL scenario is provided in Section 4. In Section 5 we demonstrate the experimental results on both regression and classification tasks, showing the good performance of our proposed algorithm in MTL.

Before we start to formulate the problem, we first introduce some notations throughout the paper in advance for clarity and simplicity. Scalars, vectors, and matrices are denoted by lower case letters, bold lower case letters, and bold capital letters, respectively. x_i denotes the i-th entry of a vector $\mathbf{x}$, x_{ij} denotes the $(i,\ j)$-th entry of a matrix $\mathbf{X}$. For the i-th row in a matrix $\mathbf{X}$, we use $\mathbf{x}^i$, and for the i-th column we use $\mathbf{x}_i$. The $l_{p,q}$-norm of a matrix $\mathbf{X}$ is defined as $\|\mathbf{X}\|_{p,q} = (\sum_i((\sum_j x_{ij}^p)^{1/p})^q)^{1/q}$. The inner product of matrices $\mathbf{X}$ and $\mathbf{Y}$ is denoted as $\langle \mathbf{X}, \mathbf{Y}\rangle$. $\|\cdot\|_F$ represents the Frobenius norm. $\mathbf{P}^k$ denotes the $\mathbf{P}$ matrix we obtain in the k-th iteration, η_{Pk} denotes the stepsize of $\mathbf{P}$ in the k-th iteration, $prox()$ denotes the proximal operator.

2 Problem Formulation

In MTL, we are given m learning tasks associated with the input data $\{\mathbf{X}_1, \dots, \mathbf{X}_m\}$ and the corresponding responses $\{\mathbf{y}_1, \dots, \mathbf{y}_m\}$, where $\mathbf{X}_i \in \mathbb{R}^{n_i \times d}$ with each row as a sample, and $\mathbf{y}_i \in \mathbb{R}^{n_i \times 1}$. d is the number of features in each task, and n_i is the number of samples in the i_{th} task. For each task, we aim to learn a vector of coefficients $\mathbf{p}_i$ such that $\mathbf{y}_i \approx \mathbf{X}_i\mathbf{p}_i$, the matrix $\mathbf{P}$ is formed by the m coefficient vectors as $\mathbf{P} = [\mathbf{p}_1, \dots, \mathbf{p}_m] \in \mathbb{R}^{d\times m}$.

Assume we have some prior knowledge about the relationships between features, we are able to construct a matrix $\mathbf{D}$ to contain such prior information, in this way the regularization term $\|\mathbf{DP}\|_F^2$ is able to force that the learned coefficients are in accordance with the given prior information. To give a straightforward illustration of how it works, we provide an example as follows: suppose we have some prior knowledge regarding features, say the i_{th} feature and the j_{th} feature have a similar influence on the response, then accordingly the corresponding coefficients should be close, namely $\|\mathbf{p}^i - \mathbf{p}^j\|$ should be small. In practice, there can be various such feature relationship constraints (say s), by following the above we can formulate the constraint as:

$$\sum_{t=1}^{s} \|\mathbf{p}^{i(t)} - \mathbf{p}^{j(t)}\|^2 = \left\| \underbrace{\begin{bmatrix} d_{11} & \dots & d_{1d} \\ \vdots & \ddots & \dots \\ d_{s1} & \dots & d_{sd} \end{bmatrix}}_{\mathbf{D}} \cdot \begin{bmatrix} p_{11} & \dots & p_{1m} \\ \vdots & \ddots & \dots \\ p_{d1} & \dots & p_{dm} \end{bmatrix} \right\|_F^2, \tag{2.2}$$

where $i(t)$ and $j(t)$ denote the indices in t-th constraint and each row in $\mathbf{D}$ all elements are 0's except a pair of $\{1, -1\}$ indexed by $i(t)$ and $j(t)$. Furthermore, related tasks sharing a common set of features should have similar coefficients for each feature, thus we can integrate the term $\|\mathbf{p}_i - \mathbf{p}_{i+1}\|^2$ in the objective to ensure the smoothness of coefficients' changing between two adjacent tasks (such as a temporal task).

Combined together, our multi-task learning with prior information is formulated as follows:

$$\begin{aligned} \min_{\mathbf{P}} & \frac{1}{2}\sum_{i=1}^{m} \|\mathbf{X}_i\mathbf{p}_i - \mathbf{y}_i\|^2 + \lambda\|\mathbf{P}\|_{2,1} \\ & + \frac{1}{2}\theta\|\mathbf{DP}\|_F^2 + \frac{1}{2}\epsilon\sum_{i=1}^{m-1} \|\mathbf{p}_i - \mathbf{p}_{i+1}\|^2, \end{aligned} \tag{2.3}$$

where all the parameters $\lambda, \theta, \epsilon$ are nonnegative. Specifically, the first term is the empirical loss, while the following $l_{2,1}$-norm regularization term is based on the group Lasso penalty, which is applied to the rows of $\mathbf{P}$ to identify a common set of features [1]. The last two regularization terms are aiming to obtain a $\mathbf{P}$ that is

[1]One can also change the group Lasso to Nuclear norm

consistent with given prior information. From the problem formulation, it's easy to see we have the beneficial fact that the smooth part in the objective function is strongly-convex.

3 Optimization Algorithm

In this section, we will show how to solve the problem formulated in Eq.(2.3) with multiple methods. Obviously, it is a nonsmooth convex problem due to the existence of the group Lasso regularization term. To handle this, we can decompose Eq.(2.3) into two parts:

$$\begin{aligned} f(\mathbf{P}) &= \frac{1}{2}\sum_{i=1}^{m}\|\mathbf{X}_i\mathbf{p}_i - \mathbf{y}_i\|^2 + \frac{1}{2}\theta\|\mathbf{D}\mathbf{P}\|_F^2 \\ &+ \frac{1}{2}\epsilon\sum_{i=1}^{m-1}\|\mathbf{p}_i - \mathbf{p}_{i+1}\|^2, \end{aligned} \tag{3.4}$$

$$g(\mathbf{P}) = \lambda\|\mathbf{P}\|_{2,1}, \tag{3.5}$$

thus

$$F(\mathbf{P}) = f(\mathbf{P}) + g(\mathbf{P}), \tag{3.6}$$

where $f(\mathbf{P})$ is a smooth differential strongly-convex function, $g(\mathbf{P})$ is a nondifferential convex function. Without loss of generality, let $s^i_{min}, s^i_{max}, d_{min}, d_{max}$ denote the minimum and maximum singular value of $\mathbf{X}_i^T\mathbf{X}_i$ and $\mathbf{D}^T\mathbf{D}$, respectively, the Lipschitz constant L_P of $f(\mathbf{P})$ can be calculated as $max_i(s^i_{max}) + \theta \cdot d_{max} + 2\epsilon$, and the strongly-convex constant σ_P can be calculated as $min_i(s^i_{min}) + \theta \cdot d_{min} + \epsilon$.

Following the basic approximation model in [3], given the Taylor expansion of $f(\mathbf{P})$ at $(\mathbf{A})$ is

$$T_{\mathbf{A},\eta_P}(\mathbf{P}) = f(\mathbf{A}) + \langle \nabla f(\mathbf{A}), \mathbf{P} - \mathbf{A}\rangle + \frac{\eta_P}{2}\|\mathbf{P} - \mathbf{A}\|_F^2, \tag{3.7}$$

we can minimize $F(\mathbf{P})$ via minimizing its quadratic approximation $M_{\mathbf{A},\eta_P}(\mathbf{P})$:

$$\arg\min_{\mathbf{P}} M_{\mathbf{A},\eta_P}(\mathbf{P}) = \arg\min_{\mathbf{P}} T_{\mathbf{A},\eta_P}(\mathbf{P}) + g(\mathbf{P}), \tag{3.8}$$

which admits a unique minimizer for any $\eta_P > 0$. Moreover, as long as $\eta_P \geq L_p$, then $M_{\mathbf{A},\eta_P}(\mathbf{P})$ is a majorization function w.r.t $F(\mathbf{P})$, therefore we can leverage majorize-minimization to optimize $\mathbf{P}$.

($\|\mathbf{P}\|_* = \sum_i \sigma_i(\mathbf{P})$) to obtain the low-rank property of $\mathbf{P}$, the whole general process remains the same except the proximal solution changing to *SVT* (Singular Value Thresholding) in Eq. (3.9).

Algorithm 1 Vanilla Gradient Descent Method with Constant Stepsize

```
**Input:** $\eta_P = L_P$.
**Initialization:** $\mathbf{P}^0$
**repeat**
  **1**. Set $\mathbf{A} = \mathbf{P}^{k-1}$.
  **2**. Calculate $\mathbf{P}^k$ according to Eq.(3.10).
**until** convergence
```

Therefore we can get the following optimization problem:

$$\begin{aligned} \mathbf{P} &= \arg\min_{\mathbf{P}} f(\mathbf{A}) + \langle \nabla f(\mathbf{A}), \mathbf{P} - \mathbf{A}\rangle + \frac{\eta_P}{2}\|\mathbf{P} - \mathbf{A}\|_F^2 + \lambda\|\mathbf{P}\|_{2,1} \\ &= \arg\min_{\mathbf{P}} \frac{\eta_P}{2}\|\mathbf{P} - (\mathbf{A} - \frac{1}{\eta_P}\nabla f(\mathbf{A}))\|_F^2 + \lambda\|\mathbf{P}\|_{2,1}, \end{aligned} \tag{3.9}$$

which leads to a closed proximal operator of rows in $\mathbf{P}$ with the following closed-form solution [20]:

$$\begin{aligned} \mathbf{U} &= \mathbf{A} - \frac{1}{\eta_P}\nabla f(\mathbf{A}), \\ \mathbf{p}^i &= prox_{\eta_P}(\mathbf{u^i}) = \max(0, 1 - \frac{\lambda}{\eta_P\|\mathbf{u^i}\|})\mathbf{u}^i. \end{aligned} \tag{3.10}$$

We propose three methods to solve the problem above, including the vanilla gradient descent method with constant stepsize, ISTA with modified stepsize searching, and an algorithm proposed by us with a linear convergence rate.

In vanilla gradient descent with constant stepsize, the optimal solution at the k-th iteration is obtained by solving the following problem

$$\mathbf{P}^k = \arg\min_{\mathbf{P}} M_{\mathbf{P}^{k-1},\eta_P}(\mathbf{P}) \tag{3.11}$$

with an appropriate stepsize $1/\eta_P$. We can verify that the objective has a sufficient decrease when we set η_P as the Lipschitz constant L_P of $f(\mathbf{P})$ with regards to $\mathbf{P}$. The algorithm is summarized in Algorithm 1.

While it is guaranteed that the objective in Eq.(2.3) is monotonically non-increasing with vanilla gradient descent, an obvious drawback of using $1/L_P$ as the constant stepsize is it is too small to achieve an optimal result rapidly. To improve it, we can apply ISTA with modified stepsize searching. In the previous work about ISTA with backtracking, $\eta_{P0} > 0$ and $\beta_P > 1$ are initialized randomly and we need to find the smallest non-negative integer i_k such that with $\eta_P = \beta_P^{i_k}\eta_{P^{k-1}}$ we have

$$F(prox_{\eta_P}(\mathbf{P}^{k-1})) \leq M_{\mathbf{P^{k-1}},\eta_P}(prox_{\eta_P}(\mathbf{P}^{k-1})). \tag{3.12}$$

One potential flaw in the ISTA with the conventional backtracking method described above lies in the initialization of η_0. It is possible that the value of η_0 at the

Algorithm 2 ISTA with Modified stepsize Searching

```
**Input:** $\eta_{P0} = L_P, 0 < \beta_P < 1$.
**Initialization:** $\mathbf{P}^0$
**repeat**
  1. Find the smallest integer $i_k$ such that with $\eta_P = \beta_P^{i_k}\eta_{P0}$ Eq. (3.13) is satisfied. Set $\eta_{Pk} = \eta_P/\beta_P$.
  3. With $\mathbf{A} = \mathbf{P}^{k-1}, \eta_P = \eta_{Pk}$, calculate $\mathbf{P}^k$ according to Eq.(3.10).
**until** convergence
```

Algorithm 3 Fast Algorithm with Linear Convergence rate

```
**Input:** $\eta_P = L_P, c = \frac{L_P}{\sigma_P}$.
**Initialization:** $\mathbf{P}^0$, set $\mathbf{A}^0 = \mathbf{P}^0$
**repeat**
  1. calculate $\mathbf{P}^k$ according to Eq.(3.10).
  2. Update $\mathbf{A}^k = \mathbf{P}^k + \frac{\sqrt{c}-1}{\sqrt{c}+1}(\mathbf{P}^k - \mathbf{P}^{k-1})$
**until** convergence
```

very first step is already larger than the actual Lipschitz constant L, and the starting step size is already too small to have fast convergence. And the stepsize η_k searching in the k_{th} iteration always starts from η_{k-1}, thus there may be a larger stepsize available satisfying the condition Eq.(3.12) in the k_{th} iteration that cannot be discovered by this searching process. For this reason, we do the stepsize searching reversely starting from setting η_0 to its Lipschitz constant L, then we keep shrinking it until Eq.(3.12) is not satisfied in terms of $\mathbf{P}$ in the optimization process, which is

$$F(prox_{\eta_P}(\mathbf{P}^{k-1})) > M_{\mathbf{P^{k-1}},\eta_P}(prox_{\eta_P}(\mathbf{P}^{k-1})), \tag{3.13}$$

in this way, we are able to find the largest stepsize meeting the condition in each iteration. By updating as Algorithm 2, the objective is decreasing much faster than vanilla gradient descent with constant step size.

While ISTA converges faster than vanilla gradient descent with constant stepsize, the convergence rate is still sublinear (including FISTA), therefore we propose a new algorithm to solve the multi-task learning problem in Eq.(3.9) with a linear convergence rate, utilizing the strongly-convex property of $f(\mathbf{P})$. As we said before, all the parameters $\lambda, \theta, \epsilon$ are nonnegative, thus we are able to guarantee that $f(\mathbf{P})$ in Eq.(3.4) is strongly-convex with σ_P, and Lipschitz smooth with L_P. The algorithm is summarized as Algorithm 3. By updating as Algorithm 3, although the objective in Eq.(2.3) is not guaranteed to be monotonically non-increasing, in general it can achieve an optimal solution with a higher convergence rate compared with Algorithm 1 and Algorithm 2.

Figure 1 shows the objective versus the number of iterations in five algorithms with a synthetic dataset. We can see that ISTA with our modified stepsize searching converges faster than ISTA with backtracking in [3], and the new algorithm we proposed generally converges faster than FISTA with backtracking in [3].

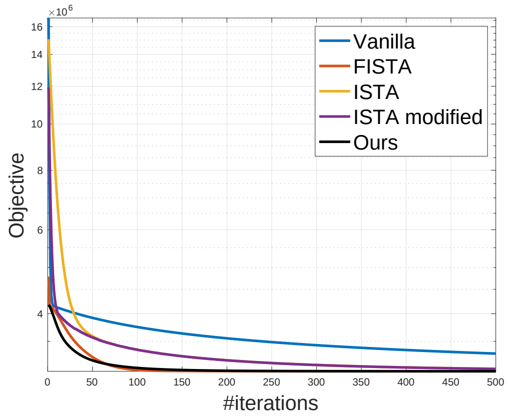


Figure 1: Objective plot in five algorithms.

4 Convergence Analysis

In the previous section, we mentioned Algorithm 2 has a sublinear convergence rate and Algorithm 3 has a linear convergence rate. There are other studies showing algorithms with a linear convergence rate for solving such problem [27], but different from these studies, there is no strong assumption required in our algorithm, and we utilize the momentum trick following the Nesterov accelerated gradient, which is proven to be unbeatable in general cases.

The convergence proof of Algorithm 2 can be easily adapted from the proof in [3] to modified stepsize searching and be extended from vector variables to matrix variables due to the equivalence of matrix Frobenius norm and vector Euclidean norm. Here we only present the key lemma and theorem of the convergence rate:

Lemma 4.1. *If for $\mathbf{P} \in \mathbb{R}^{d\times m}$, we have $F(prox_{\eta_P}(\mathbf{P})) \leq M_{\mathbf{P},\eta_P}(prox_{\eta_P}(\mathbf{P}))$, then for any $\mathbf{A} \in \mathbb{R}^{d\times m}$, $F(\mathbf{A}) - F(prox_{\eta_P}(\mathbf{P})) \geq \frac{\eta_P}{2}\|prox_{\eta_P}(\mathbf{P}) - \mathbf{P}\|_F^2 + \eta_P\langle \mathbf{P} - \mathbf{A}, prox_{\eta_P}(\mathbf{P}) - \mathbf{P}\rangle$.*

We present the following theorem about the convergence rate of solving Eq.(2.3) via Algorithm 2:

Theorem 4.1. *Let $\mathbf{P}^k$ be the output generated by Algorithm 2 in the $k-th$ iteration, then for any $k \geq 1$ we have $F(\mathbf{P}^k) - F(\mathbf{P}^*) = O(\frac{1}{k})$, where $\mathbf{P}^*$ is the optimal solution in Eq.(2.3).*

Before diving into the detailed proof of convergence rate in Algorithm 3, we first provide two useful lemmas that are important for the following proof process:

LEMMA 4.2. *For $F(x) = f(x) + g(x)$, if $g(x)$ is convex, and $f(x)$ is $\sigma-$strongly convex and $L-$smooth, then for any x, y and $\alpha > 0$ satisfying*

$$f(prox_\alpha(y)) \leq f(y) + \langle \nabla f(y), prox_\alpha(y) - y\rangle + \frac{\alpha}{2}\|prox_\alpha(y) - y\|^2 \tag{4.14}$$

the following inequality holds:

$$\begin{aligned} &F(x) - F(prox_\alpha(y)) \\ \geq &\frac{\alpha}{2}\|x - prox_\alpha(y)\|^2 - \frac{\alpha}{2}\|x-y\|^2 \\ &+ f(x) - f(y) - \langle \nabla f(y), x - y\rangle. \end{aligned} \tag{4.15}$$

Proof. Consider function $\phi(u) = f(y) + \langle \nabla f(y), u - y\rangle + g(u) + \frac{\alpha}{2}\|u - y\|^2$, it is obvious that such $\phi(u)$ is $\alpha-$strongly convex and $prox_\alpha(y) = \arg\min_u(\phi(u))$. Then we have

$$\phi(x) - \phi(prox_\alpha(y)) \geq \frac{\alpha}{2}\|x - prox_\alpha(y)\|^2. \tag{4.16}$$

According to Eq.(4.14):

$$\begin{aligned} &\phi(prox_\alpha(y)) \\ =&f(y) + \langle \nabla f(y), prox_\alpha(y) - y\rangle \\ &+ \frac{\alpha}{2}\|y - prox_\alpha(y)\|^2 + g(prox_\alpha(y)) \\ \geq&f(prox_\alpha(y)) + g(prox_\alpha(y)) \\ =&F(prox_\alpha(y)), \end{aligned} \tag{4.17}$$

combine with Eq.(4.16), we obtain

$$\phi(x) - F(prox_\alpha(y)) \geq \frac{\alpha}{2}\|x - prox_\alpha(y)\|^2. \tag{4.18}$$

Therefore it's easy to get the following inequality after we plug in the formula of $\phi(x)$:

$$\begin{aligned} &\phi(x) - F(prox_\alpha(y)) \\ =&F(x) - f(x) + f(y) + \frac{\alpha}{2}\|x-y\|^2 \\ &+ \langle \nabla f(y), x - y\rangle - F(prox_\alpha(y)) \\ \geq&\frac{\alpha}{2}\|x - prox_\alpha(y)\|^2, \end{aligned} \tag{4.19}$$

which is the same as Eq.(4.15). ◻

LEMMA 4.3. *For any vector $\mathbf{a}, \mathbf{b}$ and constant $\beta < 1$, we have the following equation:*

$$\|\mathbf{a}+\mathbf{b}\|^2 - \beta\|\mathbf{a}\|^2 = (1-\beta)\|\mathbf{a} + \frac{1}{1-\beta}\mathbf{b}\|^2 - \frac{\beta}{1-\beta}\|\mathbf{b}\|^2. \tag{4.20}$$

Here we introduce the theorem about the convergence rate of Algorithm 3:

THEOREM 4.2. *For $F(x) = f(x) + g(x)$ in Eq.(3.6), $g(x)$ is convex, and $f(x)$ is $\sigma-$strongly convex and $L-$smooth, let $c = \frac{L}{\sigma}$ and $t = \sqrt{c}$. Let $\mathbf{P}^k$ be the k_{th} iteration's output in Algorithm 3, $\mathbf{P}^*$ be the optimal solution,$V_k = F(\mathbf{P}^k) - F(\mathbf{P}^*)$. Then for any $k \geq 1$ we have $V_k \leq (1-\frac{1}{t})^k(V_0 + \frac{\sigma}{2}\|\mathbf{P}^0 - \mathbf{P}^*\|^2)$.*

Proof. According to Lemma 4.2 and the fact that $f(x)$ is $\sigma-$strongly convex and $L-$smooth, we obtain

$$\begin{aligned} &F(x) - F(prox_L(y)) \\ \geq&\frac{L}{2}\|x - prox_L(y)\|^2 - \frac{L}{2}\|x-y\|^2 + \frac{\sigma}{2}\|x-y\|^2. \end{aligned} \tag{4.21}$$

Invoking the above inequality with $x = \frac{1}{t}\mathbf{P}^* + (1-\frac{1}{t})\mathbf{P}^k$ and $y = \mathbf{A}^k$ in Algorithm 3, we get

$$\begin{aligned} &F(\frac{1}{t}\mathbf{P}^* + (1-\frac{1}{t})\mathbf{P}^k) - F(\mathbf{P}^{k+1}) \\ \geq&\frac{L}{2}\|\mathbf{P}^{k+1} - (\frac{1}{t}\mathbf{P}^* + (1-\frac{1}{t})\mathbf{P}^k)\|^2 \\ &- \frac{L-\sigma}{2}\|\mathbf{A}^k - (\frac{1}{t}\mathbf{P}^* + (1-\frac{1}{t})\mathbf{P}^k)\|^2 \\ =&\frac{L}{2t^2}\|t\mathbf{P}^{k+1} - (\mathbf{P}^* + (t-1)\mathbf{P}^k)\|^2 \\ &- \frac{L-\sigma}{2t^2}\|t\mathbf{A}^k - (\mathbf{P}^* + (t-1)\mathbf{P}^k)\|^2. \end{aligned} \tag{4.22}$$

Since f is a $\sigma-$strongly convex function, for $\alpha \in [0,1]$, we have $f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha)f(y) - \frac{\sigma\alpha(1-\alpha)}{2}\|x-y\|^2$, and obviously $\frac{1}{t} \in [0,1]$, so we have

$$\begin{aligned} &F(\frac{1}{t}\mathbf{P}^* + (1-\frac{1}{t})\mathbf{P}^k) \\ \leq&\frac{1}{t}F(\mathbf{P}^*) + (1-\frac{1}{t})F(\mathbf{P}^k) - \frac{\sigma t^{-1}(1-t^{-1})}{2}\|\mathbf{P}^k - \mathbf{P}^*\|^2. \end{aligned} \tag{4.23}$$

With $V_k = F(\mathbf{P}^k) - F(\mathbf{P}^*)$, we can get

$$\begin{aligned} &F(\frac{1}{t}\mathbf{P}^* + (1-\frac{1}{t})\mathbf{P}^k) - F(\mathbf{P}^{k+1}) \\ \leq&(1-t^{-1})V_k - V_{k+1} - \frac{\sigma t^{-1}(1-t^{-1})}{2}\|\mathbf{P}^k - \mathbf{P}^*\|^2. \end{aligned} \tag{4.24}$$

Combine Eq.(4.24) and Eq.(4.22),we have

$$\begin{aligned} &\frac{L-\sigma}{2}\|t\mathbf{A}^k - (\mathbf{P}^* + (t-1)\mathbf{P}^k)\|^2 - \frac{\sigma(t-1)}{2}\|\mathbf{P}^k - \mathbf{P}^*\|^2 \\ \geq& t^2V_{k+1} - t(t-1)V_k + \frac{L}{2}\|t\mathbf{P}^{k+1} - (\mathbf{P}^* + (t-1)\mathbf{P}^k)\|^2. \end{aligned} \tag{4.25}$$

With Lemma 4.3 and set $a := \mathbf{P}^k - \mathbf{P}^*, b := t(\mathbf{A}^k - \mathbf{P}^k), \beta := \frac{\sigma(t-1)}{L-\sigma}$, then for the left side in the above inequality, we have

(4.26)
$$\begin{aligned}
&\frac{L-\sigma}{2}\|t\mathbf{A}^k - (\mathbf{P}^* + (t-1)\mathbf{P}^k)\|^2 \\
&- \frac{\sigma(t-1)}{2}\|\mathbf{P}^k - \mathbf{P}^*\|^2 \\
=&\frac{L-\sigma}{2}\{\|t(\mathbf{A}^k - \mathbf{P}^k) + (\mathbf{P}^k - \mathbf{P}^*)\|^2 \\
&- \frac{\sigma(t-1)}{L-\sigma}\|\mathbf{P}^k - \mathbf{P}^*\|^2\} \\
=&\frac{L-\sigma}{2}\{\frac{L-\sigma t}{L-\sigma}\|(\mathbf{P}^k - \mathbf{P}^*) + \frac{L-\sigma}{L-\sigma t}t(\mathbf{A}^k - \mathbf{P}^k)\|^2 \\
&- \frac{\sigma(t-1)}{L-\sigma t}\|t(\mathbf{A}^k - \mathbf{P}^k)\|^2\} \\
\leq&\frac{L-\sigma t}{2}\|\mathbf{P}^k - \mathbf{P}^* + \frac{L-\sigma}{L-\sigma t}t(\mathbf{A}^k - \mathbf{P}^k)\|^2
\end{aligned}$$

Therefore we have the following inequality according to Eq.(4.25):

(4.27)
$$\begin{aligned}
&t(t-1)V_k + \frac{L-\sigma t}{2}\|\mathbf{P}^k - \mathbf{P}^* + \frac{L-\sigma}{L-\sigma t}t(\mathbf{A}^k - \mathbf{P}^k)\|^2 \\
&\geq t^2 V_{k+1} + \frac{L}{2}\|t\mathbf{P}^{k+1} - (\mathbf{P}^* + (t-1)\mathbf{P}^k)\|^2.
\end{aligned}$$

With the update rule $\mathbf{A}^k = \mathbf{P}^k + \frac{\sqrt{c}-1}{\sqrt{c}+1}(\mathbf{P}^k - \mathbf{P}^{k-1})$ in Algorithm 3 and $t = \sqrt{c}$,

$$\begin{aligned}
&\mathbf{P}^k - \mathbf{P}^* + \frac{L-\sigma}{L-\sigma t}t(\mathbf{A}^k - \mathbf{P}^k) \\
=&\mathbf{P}^k - \mathbf{P}^* + \frac{L-\sigma}{L-\sigma t}\frac{t(t-1)}{t+1}(\mathbf{P}^k - \mathbf{P}^{k-1}) \\
=&t\mathbf{P}^k - (\mathbf{P}^* + (t-1)\mathbf{P}^{k-1}).
\end{aligned} \tag{4.28}$$

Since $L = \sigma t^2$, based on Eq.(4.27) and Eq.(4.28), divide both sides of the inequality by t^2, we have

(4.29)
$$\begin{aligned}
&V_{k+1} + \frac{\sigma}{2}\|t\mathbf{P}^{k+1} - (\mathbf{P}^* + (t-1)\mathbf{P}^k)\|^2 \\
\leq&(1 - \frac{1}{t})(V_k + \frac{\sigma}{2}\|t\mathbf{P}^k - (\mathbf{P}^* + (t-1)\mathbf{P}^{k-1})\|^2).
\end{aligned}$$

For $k = 0$, with the initialization setting $\mathbf{A}^0 = \mathbf{P}^0$,

$$\mathbf{P}^0 - \mathbf{P}^* + \frac{L-\sigma}{L-\sigma t}t(\mathbf{A}^0 - \mathbf{P}^0) = \mathbf{P}^0 - \mathbf{P}^*, \tag{4.30}$$

based on Eq.(4.29), naturally we have

$$\begin{aligned}
&V_k + \frac{\sigma}{2}\|t\mathbf{P}^k - (\mathbf{P}^* + (t-1)\mathbf{P}^{k-1})\|^2 \\
\leq&(1 - \frac{1}{t})^k(V_0 + \frac{\sigma}{2}\|\mathbf{P}^0 - \mathbf{P}^*\|^2).
\end{aligned} \tag{4.31}$$

Thus we can get $V_k \leq (1 - \frac{1}{t})^k(V_0 + \frac{\sigma}{2}\|\mathbf{P}^0 - \mathbf{P}^*\|^2)$. The objective in Eq.(2.3) is not guaranteed to be monotonically non-increasing due to the existence of the $\frac{\sigma}{2}\|t\mathbf{P}^k - (\mathbf{P}^* + (t-1)\mathbf{P}^{k-1})\|^2$ term, but in general we can expect it can achieve optimal solution with a linear convergence rate. □

5 Experimental Results

5.1 Experiment Setup We compare the experimental results of the following MTL algorithms with our method: FedEM [18], DMTL [11], MTFL [12], MMTFL [26], MTRL [29], RMTL [6], CLMT [7], MKMTL [16]. The empirical studies are conducted on the following benchmark multi-task regression and classification datasets:

School [9]: there are exam scores of 15362 students from 139 schools in the dataset, each student is described with 28 attributes. Thus there are 139 related tasks, each sample has 28 features along with 1 output. We aim to perform multi-task regression to predict exam scores.

Sarcos [24]: it is collected for an inverse dynamics prediction problem for a seven degrees-of-freedom robot arm. The number of related tasks is 7, and there are 21 features for each sample. Following the work in [28] we sample 2000 random samples for each task.

Yale: it contains 165 images from 15 subjects, each image is scaled to 32 × 32 pixels. We use the first 8 subjects from it to construct related tasks, each task is defined as a binary classification problem of classifying two subjects, there are 28 binary classification tasks.

MNIST [15]: we use a subset from the MNIST dataset with 10000 samples of 10 handwritten digits. We cast the multi-task learning as 45 binary classification tasks to classify pairs of digits.

Letter [10]: it consists of handwritten letters from different writers, we construct 8 binary classification tasks from it to distinguish between pairs of letters.

ORL [23]: there are 10 different images of 40 distinct subjects. What's different for the ORL dataset is we construct 40 one-vs-all multi-class classification tasks from it rather than one-vs-one binary classification.

During the experiment process, each dataset is randomly split into a training set and a test set. In all classification tasks, the data is split according to a rough 60%-40% training-test split ratio, in the School regression task 20 random samples are used for training, and in the Sarcos regression task, 50 random samples are selected for training and the rest for testing. The prior knowledge matrix **D** can be initialized based on known prior, our common sense, and the correlation among features obtained with statistical methods. In the experiment, we have two methods to obtain the required **D**: the first method is to utilize the natural correlation

Table 1: Regression tasks: generalization performance measures over ten runs

Metric	**Dataset**	**FedEM**	**DMTL**	**MTFL**	**MMTFL**	**OURS-NATURAL**
VE (%)	School	38.7±3.2	28.8±5.6	29.7±2.1	31.5±4.2	**39.8±2.2**
	Sarcos	51.2±12.1	42.6±7.3	49.2±5.1	49.9±2.7	51.8±6.6
nMSE (%)	School	61.2±0.9	75.1±0.8	72.9±1.1	68.7±3.1	**60.2±0.5**
	Sarcos	15.7±3.1	20.9±0.8	19.1±2.7	17.1±2.2	14.9±0.8
Metric	**Dataset**	**MTRL**	**RMTL**	**CLMT**	**MKMTL**	**OURS-ART**
VE (%)	School	29.9±2.0	33.6±5.7	37.9±2.1	37.9±1.9	**39.8±2.5**
	Sarcos	42.5±8.2	49.9±7.2	50.1±9.9	50.1±1.7	**51.9±6.2**
nMSE (%)	School	73.1±0.9	68.9±2.7	63.7±0.7	62.7±0.9	**60.2±0.3**
	Sarcos	17.9±0.7	15.6±0.3	16.2±0.6	15.7±0.5	**14.7±0.9**

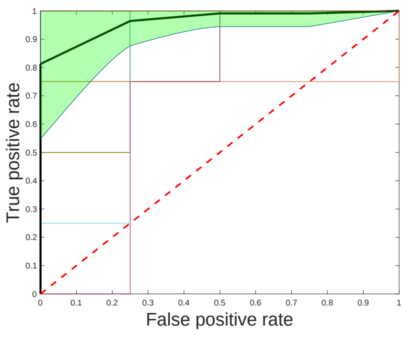
Table 2: Classification tasks: generalization performance measures over ten runs

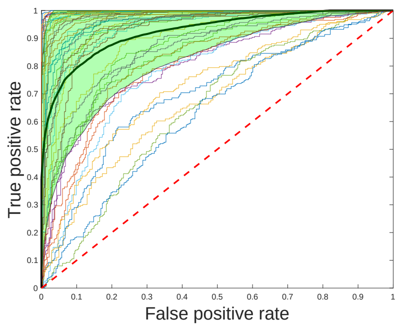
Metric	**Dataset**	**FedEM**	**DMTL**	**MTFL**	**MMTFL**	**OURS-NATURAL**
AUC (%)	Yale	96.9±2.7	95.7±3.2	86.8±1.6	92.7±1.1	**97.7±1.7**
	MNIST	**98.1±3.2**	90.9±3.1	91.2±1.9	91.5±2.3	96.5±1.2
	Letter	63.2±2.9	61.9±1.8	62.1±2.2	61.8±1.8	63.5±1.1
	ORL	81.3±5.8	72.9±3.7	77.2±5.2	77.9±3.2	82.5±2.2
Metric	**Dataset**	**MTRL**	**RMTL**	**CLMT**	**MKMTL**	**OURS-ART**
AUC (%)	Yale	96.1±3.5	97.2±1.9	96.7±2.3	95.7±1.8	**97.7±1.7**
	MNIST	93.7±7.1	92.5±3.1	94.7±3.8	93.2±5.2	97.6±1.2
	Letter	60.3±2.1	62.7±3.6	61.5±7.2	60.9±7.1	**65.5±1.3**
	ORL	77.6±3.9	80.2±9.1	78.9±6.3	78.8±3.2	**84.3±1.8**

among features, we calculate the covariance matrix for all the features, and select the strongest ones as the information contained in $\mathbf{D}$, we call the prior knowledge matrix obtained in this way the **natural D**; the second method is we create the strong correlation among features manually by appending features repeatedly on purpose, for example, we transform the original feature (x_1, x_2, x_3) into (x_1, x_2, x_3, x_1) to enhance the correlation among features (this may introduce multicollinearity, but since we only care about the predictive result, it should be fine), the matrix obtained is called as the **artificial D**. Parameters in all the methods are tuned using 5-fold cross-validation, and for each method, we stop the experiment when the objective change is $< 10^{-3}$. We use Matlab R2019a on a laptop with a 1.4 GHz QuadCore Intel Core i5 processor.

5.2 Results Results of the experiment are presented in Table 1 and Table 2, for regression tasks and classification tasks respectively, our method with a natural $\mathbf{D}$ is denoted as OURS-NATURAL, with an artificial $\mathbf{D}$ is denoted as OURS-ART. In OURS-ART, we manually repeat about 5% features to construct $\mathbf{D}$. In the case of regression tasks, we report the variance explained (VE) and the normalized mean squared error (nMSE) following previous studies [6, 12], whereas the receiver operating characteristic (ROC) curve and the area under the ROC curve (AUC) are employed as the classification performance measurements as used in previous studies [6, 12]. ROC curves show the performance of a binary classification model at all classification thresholds by plotting the true positive rate against the false positive rate. Higher explained variance and AUC indicate better performance, and the opposite for nMSE reported. Each experiment with one specific dataset is repeated 10 times and we report the averaged performance and the standard deviation, the best performances are in **bold**.

In Table 1 and Table 2, our method can achieve the best performance in most cases, while the FedEM method achieves better results with the MNIST dataset. The reason perhaps is that neural networks still have


(a) Yale


(b) MNIST

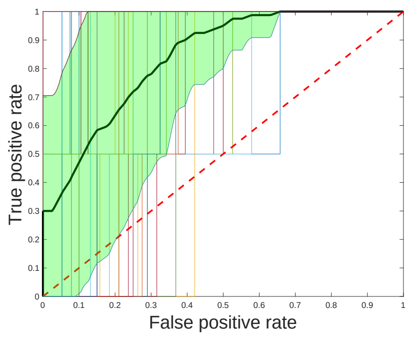
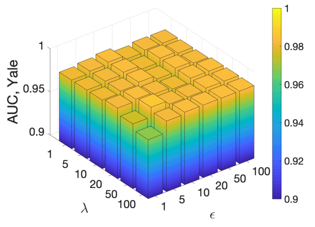
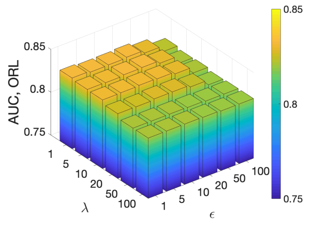
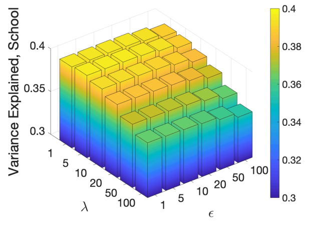

(c) ORL

Figure 2: The ROC curve for Yale, MNIST, ORL dataset with tuned parameters.


(a) AUC, Yale


(b) AUC, ORL


(c) VE, School

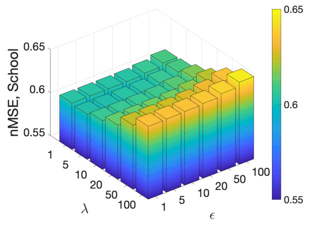

(d) nMSE, School

Figure 3: Ablation study – the influence of regularization parameters on learning performance.

unique advantages when dealing with large-scale complex image datasets, and prior knowledge about features is hard to establish with complex image data. Also, in regression tasks, our method works well with both natural prior knowledge and artificial prior knowledge. In image classification tasks, the improvement in performance with artificial prior knowledge matrices is more significant than that with natural ones, the reason may be due to the fact that the natural correlations between features in regression tasks are stronger than those involved in image classification tasks.

We present the ROC curves with a natural prior knowledge matrix for the classification tasks on Yale, MNIST, and ORL datasets in Figure 2, the slim colorful lines are the ROC curves for each task, and the weighted black line is the averaged ROC curve over all the tasks, and the green area represents the range between the mean $\pm$ standard deviation. While for each dataset there is a small number of tasks with performances that are not so perfect on some level, the overall performance is satisfactory with a pretty high AUC, and the mean $-$ standard deviation curve is almost always over the diagonal line, especially for the one-vs-one classification tasks on Yale and MNIST. We also provide Figure 3 to illustrate the influence of regularization parameters. With a super wide tuning range for each parameter, which is from 1 to 100, the effect of each parameter on the performance is not significant until the value reaches a threshold, there is a generous range for each parameter to be able to provide stable and great performance. We can roughly draw the conclusion that for classification tasks, no matter whether it's one-vs-one binary classification tasks or one-vs-all binary classification tasks, the values of regularization parameters have a marginal influence on the results as long as the value is within a reasonable range. While in a regression situation the performance is much more sensitive to the value of parameters, especially to the group Lasso penalty parameter, which is in accordance with common sense that underfitting happens with large regularization term parameters.

6 Conclusion

We propose a convex formulation of multi-task learning utilizing the prior information, including the prior-known relationships between certain coefficients and the fact that related tasks should share similar coefficients. Our main technical contribution is a new optimization algorithm to solve the formulated problem with a linear convergence rate and we provide detailed proof of the convergence rate. Results on benchmark datasets with regression and classification tasks demonstrate the effectiveness of the proposed multi-task learning formulation and algorithm. Also, we are interested in identifying outlier tasks using the framework we propose in the paper, this remains to be our future work.

References

[1] Rie Kubota Ando, Tong Zhang, and Peter Bartlett. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6(11), 2005.

[2] BJ Bakker and TM Heskes. Task clustering and gating for bayesian multitask learning. 2003.

[3] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

[4] Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

[5] Antonin Chambolle and Ch Dossal. On the convergence of the iterates of the "fast iterative shrinkage/thresholding algorithm". *Journal of Optimization theory and Applications*, 166(3):968–982, 2015.

[6] Jianhui Chen, Jiayu Zhou, and Jieping Ye. Integrating low-rank and group-sparse structures for robust multi-task learning. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 42–50, 2011.

[7] Carlo Ciliberto, Youssef Mroueh, Tomaso Poggio, and Lorenzo Rosasco. Convex learning of multiple tasks and their structure. In *International Conference on Machine Learning*, pages 1548–1557. PMLR, 2015.

[8] Kevin Clark, Minh-Thang Luong, Urvashi Khandelwal, Christopher D Manning, and Quoc V Le. Bam! born-again multi-task networks for natural language understanding. *arXiv preprint arXiv:1907.04829*, 2019.

[9] Paulo Cortez and Alice Maria Gonçalves Silva. Using data mining to predict secondary school student performance. 2008.

[10] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[11] Ali Jalali, Sujay Sanghavi, Chao Ruan, and Pradeep Ravikumar. A dirty model for multi-task learning. *Advances in neural information processing systems*, 23, 2010.

[12] Pratik Jawanpuria and J Saketha Nath. A convex feature learning formulation for latent task structure discovery. *arXiv preprint arXiv:1206.4611*, 2012.

[13] Zhuoliang Kang, Kristen Grauman, and Fei Sha. Learning with whom to share in multi-task feature learning. In *ICML*, 2011.

[14] Matthieu Kowalski. Thresholding rules and iterative shrinkage/thresholding algorithm: A convergence study. In *2014 IEEE International Conference on Image Processing (ICIP)*, pages 4151–4155. IEEE, 2014.

[15] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

[16] Xiaoli Liu, Peng Cao, Jinzhu Yang, and Dazhe Zhao. Linearized and kernelized sparse multitask learning for predicting cognitive outcomes in alzheimer's disease. *Computational and mathematical methods in medicine*, 2018, 2018.

[17] Aakarsh Malhotra, Surbhi Mittal, Puspita Majumdar, Saheb Chhabra, Kartik Thakral, Mayank Vatsa, Richa Singh, Santanu Chaudhury, Ashwin Pudrod, and Anjali Agrawal. Multi-task driven explainable diagnosis of covid-19 using chest x-ray images. *Pattern Recognition*, 122:108243, 2022.

[18] Othmane Marfoq, Giovanni Neglia, Aurélien Bellet, Laetitia Kameni, and Richard Vidal. Federated multi-task learning under a mixture of distributions. *Advances in Neural Information Processing Systems*, 34:15434–15447, 2021.

[19] Sahand Negahban and Martin J Wainwright. Estimation of (near) low-rank matrices with noise and high-dimensional scaling. *The Annals of Statistics*, pages 1069–1097, 2011.

[20] Neal Parikh and Stephen Boyd. Proximal algorithms. *Foundations and Trends in optimization*, 1(3):127–239, 2014.

[21] Ting Kei Pong, Paul Tseng, Shuiwang Ji, and Jieping Ye. Trace norm regularization: Reformulations, algorithms, and multi-task learning. *SIAM Journal on Optimization*, 20(6):3465–3489, 2010.

[22] Novi Quadrianto, James Petterson, Tibério Caetano, Alex Smola, and Svn Vishwanathan. Multitask learning without label correspondences. *Advances in Neural Information Processing Systems*, 23:1957–1965, 2010.

[23] FS Samaria and AC Harter. In: Proceedings of the 2nd ieee workshop on applications of computer vision. december 1994, sarasota (florida). parameterisation of a stochastic model for human face identification.

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

[25] Nelson Vithayathil Varghese and Qusay H Mahmoud. A survey of multi-task deep reinforcement learning. *Electronics*, 9(9):1363, 2020.

[26] Xin Wang, Jinbo Bi, Shipeng Yu, and Jiangwen Sun. On multiplicative multitask feature learning. *Advances in Neural Information Processing Systems*, 27, 2014.

[27] Haibin Zhang, Jiaojiao Jiang, and Zhi-Quan Luo. On the linear convergence of a proximal gradient method for a class of nonsmooth convex minimization problems. *Journal of the Operations Research Society of China*, 1(2):163–186, 2013.

[28] Yu Zhang and Dit-Yan Yeung. Semi-supervised multi-task regression. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 617–631. Springer, 2009.

[29] Yu Zhang and Dit-Yan Yeung. A convex formulation for learning task relationships in multi-task learning. *arXiv preprint arXiv:1203.3536*, 2012.