Calibrated Multi-Task Learning

Feiping Nie
School of Computer Science
Center for OPTIMAL
Northwestern Polytechnical University
Xi'an, China
feipingnie@gmail.com

Zhanxuan Hu
School of Computer Science
Center for OPTIMAL
Northwestern Polytechnical University
Xi'an, China
huzhanxuan@mail.nwpu.edu.cn

Xuelong Li
OPTIMAL, Xian Institute of Optics and Precision Mechanics
Chinese Academy of Sciences
Xi'an, China
xuelong_li@opt.ac.cn

ABSTRACT

This paper proposes a novel algorithm, named Non-Convex Calibrated Multi-Task Learning (NC-CMTL), for learning multiple *related* regression tasks jointly. Instead of utilizing the nuclear norm, NC-CMTL adopts a non-convex low rank regularizer to explore the shared information among different tasks. In addition, considering that the regularization parameter for each regression task desponds on its noise level, we replace the least squares loss function by square-root loss function. Computationally, as proposed model has a non-smooth loss function and a non-convex regularization term, we construct an efficient re-weighted method to optimize it. Theoretically, we first present the convergence analysis of constructed method, and then prove that the derived solution is a stationary point of original problem. Particularly, the regularizer and optimization method used in this paper are also suitable for other rank minimization problems. Numerical experiments on both synthetic and real data illustrate the advantages of NC-CMTL over several state-of-the-art methods.

CCS CONCEPTS

• **Computing methodologies** → **Multi-task learning**; • **Information systems** → **Data mining**; • **Mathematics of computing** → **Regression analysis**;

KEYWORDS

Multi-Task Learning, Regression, Calibration, Loss function.

ACM Reference Format:
Feiping Nie, Zhanxuan Hu, and Xuelong Li. 2018. Calibrated Multi-Task Learning. In *KDD '18: The 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, August 19–23, 2018, London, United Kingdom.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3219819.3219951

KDD '18, August 19–23, 2018, London, United Kingdom

ACM ISBN 978-1-4503-5552-0/18/08…$15.00
https://doi.org/10.1145/3219819.3219951

1 INTRODUCTION

Multi-task learning, in essence, involves utilizing the information among different tasks to improve the generalization performance. It has been widely used in many applications, including computer version [1, 27, 33], Bioinformatics and Health Informatics [31, 35], and Web Applications [4, 7]. Recently, most research works mainly focus on the regularized model, which learns the underlying mutual information among different tasks by introducing a regularizer. According to the assumption used in model, these algorithms can be roughly grouped into five categories [32], including feature learning approach [3, 18], low-rank approach [2, 9, 10], task clustering approach [16, 17], dirty approach [14] and multi-level approach [15]. Taking a comprehensive investigation for all of them beyond the scope of this paper. Here, we mainly focus on the low rank approach due to its close connection with others.

Given K regression tasks $\{\mathcal{T}_i\}_1^K$ and corresponding training datasets $\{\boldsymbol{X}_i \in R^{n_i \times d}\}$ as well as target vectors $\{\boldsymbol{y}_i \in R^{n_i \times 1}\}$, the low rank approach aims at learning a coefficient matrix $\boldsymbol{W} \in R^{d \times K}$ with low rank structure, where $\boldsymbol{w}_i$ is the coefficient vector of i-th regression task. And the low rank structure of $\boldsymbol{W}$ interprets the relatedness of different tasks. In [2, 9], the authors assume that:

$$\boldsymbol{w}_i = \boldsymbol{U}\boldsymbol{v}_i\,, \tag{1}$$

where $\boldsymbol{U} \in R^{d \times r}$ $(r < d)$ is a low rank matrix, and its columns form a basis of the subspace where $\{\boldsymbol{w}_i\}_1^K$ lie. The similar ideas have been utilized by [29, 30]. A major limitation of these models is that the rank prior information of coefficient matrix $\boldsymbol{W}$ must be known, while preestimating the structure of $\boldsymbol{W}$ is time consuming in practical scenario.

To avoid this obstacle, a nuclear norm, i.e., trace norm based model was developed in [25]. Note that the trace norm is a convex relaxation for original rank minimization problem. Considering that multiple learning tasks may have discriminative features, Chen et al [8] further decomposed $\boldsymbol{W}$ to two matrices $\boldsymbol{L}$ and $\boldsymbol{E}$, where $\boldsymbol{L}$ is a low rank matrix representing the mutual information, and $\boldsymbol{E}$ is a sparse matrix representing the discriminative features. Subsequently, the sparse matrix $\boldsymbol{E}$ was replaced by a group-sparse matrix $\boldsymbol{C}$ in [10], which aims to eliminate the negative effect caused by outliers, i.e., the irrelevant tasks. In practice, however, the algorithms mentioned above may fail to learn the low rank structure of coefficient matrix $\boldsymbol{W}$ due to the punishment bias of nuclear norm. Recently, Han and Zhang [13] used a capped

trace norm regularizer, punishing only the singular values that are smaller than a fixed threshold τ, to cope with the Multi-Task Learning problems. Nevertheless, the performance of the method is very sensitive to the selection of parameters. Furthermore, the problem that different tasks generally have different noise level was ignored by the algorithms mentioned above.

For such an issue, a Calibrated Multivariate Regression (CMR) approach was developed in [18], and then it was improved in [12]. Nevertheless, both of them are based on Feature Learning and the optimization methods are time consuming. In this paper, we propose a novel model for multi-task learning, which utilizes a non-convex regularizer to approximate the original rank minimization problem. Particularly, motivated by the success of CMR, we choose the square-root loss function rather than the least squares loss function for each regression task. The contributions of this paper can be summarized as follows:

- We provide a novel calibrated multi-task learning model, named Non-Convex Calibrated Multi-Task Learning (NC-CMTL), which utilizes a new non-convex low-rank regularizer to alleviate the punishment bias of trace norm. In addition, the square-root loss function is used in NC-CMTL, which can help users cope with multiple tasks with different noise level.
- As NC-CMTL contains a non-smooth term and a non-convex term, we construct an efficient re-weighted method with convergence guarantee to solve it. The codes of our method can be downloaded from: https:// github.com/sudalvxin/Multi-task-Learning.git
- We give the convergence analysis for constructed optimization method and prove that the derived solution is a stationary point of original problem.

The remainder of this paper is organized as follows. We start in Sect 2 by describing the proposed model NC-CMTL. And the optimization of it is developed in Sect. 3. We provide the theory analysis of NC-CMTL in Sect. 4 and investigate its performance in Sect. 5. Finally, we conclude this paper and discuss the potential improvements in Sect. 6.

Notation: We present a summarization of notations used frequently in this paper in Table 1.

Table 1: The Summarization of Notations

Notation	Description
$\mathcal{T}_i$	The i-th regression task
d	Dimension of each training data
K	Number of tasks
n_i	Number of training data in $\mathcal{T}_i$
$\boldsymbol{X} \in R^{n\times d}$	Matrix with size $n \times d$
$\sigma_i(\boldsymbol{X}) = \sigma_i$	i-th singular value of $\boldsymbol{X}$
$\lambda_i(\boldsymbol{X}) = \lambda_i$	i-th eigenvalue of matrix $\boldsymbol{X}$
$\boldsymbol{u}_i(\boldsymbol{X}) = \boldsymbol{u}_i$	i-th left singular vector of $\boldsymbol{X}$
$\boldsymbol{v}_i(\boldsymbol{X}) = \boldsymbol{v}_i$	i-th right singular vector of $\boldsymbol{X}$
$\boldsymbol{x}^i \in R^d$,$\boldsymbol{x}_i \in R^n$	i-th row or column of $\boldsymbol{X}$
$\|\boldsymbol{x}\|_1 = \sum_{i=1}^n \|x_i\|$	ℓ_1-norm of vector $\boldsymbol{x}$
$\|\boldsymbol{x}\|_2 = \sqrt{\sum_{i=1}^n x_i^2}$	ℓ_2-norm of vector $\boldsymbol{x}$
$\|\boldsymbol{x}\|_2^2 = \sum_{i=1}^n x_i^2$	Squared ℓ_2-norm of $\boldsymbol{x}$
$\|\boldsymbol{X}\|_* = \sum_{i=1} \sigma_i(\boldsymbol{X})$	Nuclear norm of $\boldsymbol{X}$
$\|\boldsymbol{X}\|_F^2 = Tr(\boldsymbol{X}^T\boldsymbol{X})$	Squared Frobenius norm of $\boldsymbol{X}$
$\|\boldsymbol{X}\|_{2,1} = \sum_{i=1}^d \|\boldsymbol{x}\|_2$	The $\ell_{2,1}$-norm of $\boldsymbol{X}$

2 RELATED WORK AND PROPOSED MODEL

2.1 Formulation and Related work

Given K regression tasks $\{\mathcal{T}_i\}_1^K$ accompanied by training datasets $\{\boldsymbol{X}_i \in R^{n_i\times d}\}$ as well as target vectors $\{\boldsymbol{y}_i \in R^{n_i\times 1}\}$, the target of multi-task regression learning is to learn K coefficient vectors $\{\boldsymbol{w}_i \in R^{d\times 1}\}_1^K$ jointly, where $\boldsymbol{w}_i$ is the coefficient vector for $\mathcal{T}_i$. And, all coefficient vectors form a coefficient matrix $\boldsymbol{W} \in R^{d\times K}$. Without loss of generality, we suppose $K \geq d$ in rest of this paper. Here, we provide a general formulation for multi-task learning based on regularization as follows:

$$\sum_{i=1}^{K} l(\boldsymbol{w}_i) + \mu\mathcal{R}(\boldsymbol{W}), \tag{2}$$

where $\mathcal{R}(\boldsymbol{W})$ is the regularization term which depends on the prior on $\boldsymbol{W}$, $l(\bullet)$ is the loss function, and least squares loss function $l(\boldsymbol{w}) = \|\boldsymbol{X}_i\boldsymbol{w}_i - \boldsymbol{y}_i\|_2^2$ is generally used. In addition, μ is a parameter balancing these two terms. In this paper, we mainly focus on the low rank prior where rank regularizer is used in Eq. (2). That is,

$$\sum_{i=1}^{K} l(\boldsymbol{w}_i) + \mu rank(\boldsymbol{W}), \tag{3}$$

where $rank(\boldsymbol{W})$ denotes the rank of $\boldsymbol{W}$, i.e., the number of non-zero singular values of $\boldsymbol{W}$. As the rank minimization problem is NP-hard, an alternative is utilizing the trace norm to approximate it [8, 10, 25]. We temporarily denote a vector $\boldsymbol{\sigma}(\boldsymbol{X})$, where σ_i is the i-th singular value of $\boldsymbol{X}$. Actually, the rank norm can be considered as the ℓ_0-norm of $\boldsymbol{\sigma}(\boldsymbol{X})$, and the trace norm can be considered as the ℓ_1-norm of $\boldsymbol{\sigma}(\boldsymbol{X})$. Under some conditions [5], $\|\boldsymbol{\sigma}(\boldsymbol{X})\|_{\ell_1}$ is the most compact convex envelope of $\|\boldsymbol{\sigma}(\boldsymbol{X})\|_{\ell_0}$. In practice, however, these conditions are too rigorous for some real scenarios, such that the gap between ℓ_0-norm and ℓ_1-norm will degenerate the performance of algorithm. This issue also occurs in compress sensing [6]. To address it, Han and Zhang [13] replaced the trace norm by a capped trace norm and reformulated the problem (3) as

$$\sum_{i=1}^{K} \|\boldsymbol{X}_i\boldsymbol{w}_i - \boldsymbol{y}_i\|_2^2 + \mu\sum_{i=1}^{d} min(\sigma_i(\boldsymbol{W}), \tau), \tag{4}$$

where τ is a constant, i.e., the cap of all singular values. Obviously, the capped trace norm punishes only the singular

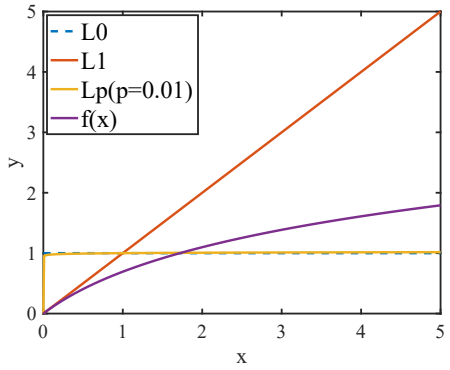


Figure 1: One dimensional illustrations with respect to four regularization terms. Here, $L0$ corresponds to the rank regularizer, $L1$ corresponds to the trace norm, and $f(x) = log(x+1)$ is the regularizer used in this paper.

values smaller than τ. And, a similar regularizer, Truncated Nuclear Norm Regularization, has been used to cope with the matrix completion problem [28]. Both of them are non-convex and have showed the advantages over trace norm, but the performance of these two regularization terms are sensitive to the selection of parameters. In addition, the least squares loss function is used in (4), which is unreasonable when different tasks have different noise level. Motivated by these issues, we propose a novel model for multi-task regression learning.

2.2 Proposed Model

To reduce the gap between relaxation problem and original rank minimization problem, in this paper we use a new regularizer [23], which is denoted as

$$\sum_{i=1}^{d} log(\sigma_i(\boldsymbol{W}) + 1)\,. \tag{5}$$

Correspondingly, the Eq. (4) can be rewritten as:

$$\sum_{i=1}^{K} \|\boldsymbol{X}_i\boldsymbol{w}_i - \boldsymbol{y}_i\|_2^2 + \mu \sum_{i=1}^{d} log(\sigma_i(\boldsymbol{W}) + 1)\,. \tag{6}$$

The model is named as NC-MTL (Multi-Task Learning based on Non-Convex relaxation). As presented in Fig. 1, one dimensional illustrations of widely used four regularization terms, the $f(x)$ used in this paper is closer to ℓ_0-norm comparing to ℓ_1-norm, i.e., the trace norm. Here, the readers may have a question: why don't we select the Lp-norm ($p = 0.01$) serve as regularizer, which is closer to ℓ_0-norm comparing to $f(x)$.

To answer the question, we provide a value comparison between three regularization terms in Table 2. It is well known that the model generally prefers to punish the large values in a minimization problem. Observing Table 2, we can find that $\frac{f_2(x_2)}{f_2(x_1)} \approx 1$ when $x_1 = 0.01$ and $x_2 = 1000$. Hence, the punishment degree of $f_2(x_1)$ is close to $f_2(x_2)$. Obviously, it is unreasonable, because $x_1 \ll x_2$ and x_1 may

Table 2: Comparison between three functions

x	0.010	0.100	10.00	100.0	1000
$f_1(x) = x$	0.010	0.100	10.00	100.0	1000
$f_2(x) = x^{0.01}$	0.955	0.977	1.023	1.047	1.071
$f_3(x) = log(x+1)$	0.010	0.095	2.397	4.615	6.908

be caused by noise. In addition, when $x_1 = 10$ and $x_2 = 1000$, for $f_1(x)$, $\frac{f_1(x_2)}{f_1(x_1)} = 100$, which demonstrates that $f_1(x)$ will pay more attention to the large values. Obviously, it is also unreasonable, because the large values generally represent the significant information we want to preserve in regularizer. Nevertheless, for $f_3(x)$ used in this paper, $\frac{f_3(x_2)}{f_3(x_1)} \approx 3$, which shows that the difference of punishment between x_2 and x_1 is small. In short, our regularizer can be seen as a neutralization between $f_1(x)$ and $f_2(x)$, which reduces the punishment on large values and enhances the punishment on small values simultaneously.

We further consider a general setting that different tasks have different noise level. Under this condition, each task should has a specific regularization parameter that depends on its noise level. However, we will show in Section 3 that the least squares loss function used in Eq. (6) enforces all tasks to share a common regularization parameter μ. To avoid this issue, we replace the least squares loss function by square-root loss function, and propose a calibrated version of model (6), named as NC-CMTL (Calibrated Multi-Task Learning based on Non-Convex relaxation). The concrete formulation of NC-CMTL is:

$$\min_{\boldsymbol{W}} \sum_{i=1}^{K} \|\boldsymbol{X}_i\boldsymbol{w}_i - \boldsymbol{y}_i\|_2 + \mu \sum_{i=1}^{d} log(\sigma_i(\boldsymbol{W}) + 1)\,. \tag{7}$$

A particular case of this problem is that all tasks share a common training dataset, and the corresponding model is:

$$\min_{\boldsymbol{W}} \|\boldsymbol{X}\boldsymbol{W} - \boldsymbol{Y}\|_{2,1} + \mu \sum_{i=1}^{d} log(\sigma_i(\boldsymbol{W}) + 1)\,. \tag{8}$$

In this paper, we mainly focus on the problem (7), because it is more complicated comparing to (8) and (6). But, one can find that Eq. (7) contains a non-smooth loss function and a non-convex regularization term, and optimizing it is a difficult work. Next, we develop an efficient re-weighted method to solve it. Particularly, the proposed optimization method is also suitable for problem (8) and problem (6).

3 OPTIMIZE THE PROPOSED MODEL

The re-weighted method has been widely used to solve various problems, such as RPCA [22], feature selection [21], Schatten p-norm minimization problem [20], and compress sensing [11]. Comparing to the models mentioned above, the Eq. (7) is more difficult to be solved. By defining two new

functions [1] $h(\boldsymbol{W}) = \boldsymbol{W}\boldsymbol{W}^T$ as well as $p(x) = x^{\frac{1}{2}}$, we can reformulate the Eq. (7) as:

$$\min_{\boldsymbol{W}} \sum_{i=1}^{K} p(l_i) + \mu \sum_{i=1}^{d} r(\lambda_i(h(\boldsymbol{W}))) \,, \tag{9}$$

where $l_i = \|\boldsymbol{X}_i\boldsymbol{w}_i - \boldsymbol{y}_i)\|_2^2$, $r(x) = log(x^{\frac{1}{2}} + 1)$. Further, it can be rewritten as:

$$\min_{\boldsymbol{W}} \mathcal{P}(\boldsymbol{W}) = \mathcal{L}(\boldsymbol{W}) + \mu\mathcal{R}(\boldsymbol{W}) \,, \tag{10}$$

where $\mathcal{L}(\boldsymbol{W}) = \sum_{i=1}^{K} p(l_i)$, and $\mathcal{R}(\boldsymbol{W}) = \mu \sum_{i=1}^{d} r(\lambda_i(h(\boldsymbol{W})))$. Before continuing, we introduce the Lemma 1.

LEMMA 1. *Given a symmetric $n \times n$ matrix $\boldsymbol{X}$, we denote its i-th eigenvalue and corresponding eigenvector by λ_i and $\boldsymbol{u}_i$, respectively [24]. Then, we have*

$$\frac{\partial \lambda_i}{\partial \boldsymbol{X}} = \boldsymbol{u}_i\boldsymbol{u}_i^T \,. \tag{11}$$

Suppose $\boldsymbol{W}\boldsymbol{W}^T = \sum_{i=1}^{d} \lambda_i \boldsymbol{u}_i\boldsymbol{u}_i^T$ is the eigenvalue decomposition of $h(\boldsymbol{W})$. According to the Lemma 1, we know that the derivative of $\mathcal{R}(\boldsymbol{W})$ is:

$$\frac{\partial \mathcal{R}(\boldsymbol{W})}{\partial \boldsymbol{W}} = 2\boldsymbol{D}^T\boldsymbol{W} \,, \tag{12}$$

where

$$\boldsymbol{D} = \sum_{i=1} r'(\lambda_i)\boldsymbol{u}_i\boldsymbol{u}_i^T \tag{13}$$

can be seen as the weighted matrix [2]. For $\mathcal{L}(\boldsymbol{W})$, we take the derivative of each column, and obtain:

$$\frac{\partial p(l_i)}{\partial \boldsymbol{w}_i} = \frac{\boldsymbol{X}_i^T(\boldsymbol{X}_i\boldsymbol{w}_i - \boldsymbol{y}_i)}{\|\boldsymbol{X}_i\boldsymbol{w}_i - \boldsymbol{y}_i)\|_2} = v_i\boldsymbol{X}_i^T(\boldsymbol{X}_i\boldsymbol{w}_i - \boldsymbol{y}_i) \,, \tag{14}$$

where

$$v_i = \frac{1}{\|\boldsymbol{X}_i\boldsymbol{w}_i - \boldsymbol{y}_i\|_2} \tag{15}$$

can be considered as the weight of task $\mathcal{T}_i$. Combining the Eq. (9), Eq. (12) and Eq. (14) gives:

$$\sum_{i=1}^{K}(v_i\boldsymbol{X}_i^T(\boldsymbol{X}_i\boldsymbol{w}_i - \boldsymbol{y}_i) + 2\mu\boldsymbol{D}^T\boldsymbol{w}_i) = 0 \,. \tag{16}$$

Further it can be written as:

$$\sum_{i=1}^{K}(\boldsymbol{X}_i^T(\boldsymbol{X}_i\boldsymbol{w}_i - \boldsymbol{y}_i) + \frac{2\mu}{v_i}\boldsymbol{D}^T\boldsymbol{w}_i) = 0 \,. \tag{17}$$

That is, for each i, we can obtain the optimal solution of $\boldsymbol{w}_i$ via solving the following problem:

$$\boldsymbol{X}_i^T(\boldsymbol{X}_i\boldsymbol{w}_i - \boldsymbol{y}_i) + \frac{2\mu}{v_i}\boldsymbol{D}^T\boldsymbol{w}_i = 0 \,. \tag{18}$$

Obviously, for task $\mathcal{T}_i$, we has a particular regularization parameter $\hat{\mu}_i = \frac{2\mu}{v_i}$ which depends on the regression error of $\mathcal{T}_i$. The mechanism is different from NC-MTL which enforces all tasks share a common regularization parameter 2μ (The derivation is similar to above).

[1] In experiment, we set $h(\boldsymbol{W}) = \boldsymbol{W}\boldsymbol{W}^T + \varepsilon_1\boldsymbol{I}$ and $p(x) = (x + \varepsilon_2^2)^{\frac{1}{2}}$ where ε_1 is a small positive constant, ε_2 is *eps* in MATLAB, and $\boldsymbol{I}$ is an identify matrix.

[2] The details w.r.t derivation can be found in Sect. 4

Algorithm 1 Reweighted method for solving proposed model

Input: The training data sets $\{\boldsymbol{X}_i, \boldsymbol{y}_i\}_i^K$.
Output: The learned coefficient matrix $\boldsymbol{W}$.
Initialization: Initialize $\{\boldsymbol{w}_i\}_1^K$ by $\{\boldsymbol{X}_i^T y_i\}_1^K$.
while not convergent **do**
1. Eigenvalue decomposition: $h(\boldsymbol{W}) = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T$;
2. Update $\boldsymbol{D}$ by Eq. (13);
3. Update $\{v_i\}_1^K$ by Eq. (15);
4. Update $\boldsymbol{W}$ via solving the Eq. (19).
5. Convergence checking.

end while

Observing Eq. (13) and Eq. (15), we can find that both $\boldsymbol{D}$ and $\{v_i\}_i^K$ depend on the variable $\boldsymbol{W}$, and them can be calculated when $\boldsymbol{W}$ is fixed. Conversely, each column of $\boldsymbol{W}$ can be computed by solving Eq. (18). Hence, we utilize an iterative method to find the solution of Eq. (9). And in $(t+1)$-th iteration, the step 4 is equivalent to solving the following problem:

$$\min_{\boldsymbol{W}} \mathcal{F}(\boldsymbol{W}) = \sum_{i=1}^{K} \frac{v_i^t}{2}\|\boldsymbol{X}_i\boldsymbol{w}_i - \boldsymbol{y}_i\|_2^2 + \mu Tr((\boldsymbol{D}^t)^T\boldsymbol{W}\boldsymbol{W}^T) \,, \tag{19}$$

where variables $\boldsymbol{D}^t$ and $\{v_i^t\}_1^K$ are defined as Eq. (13) and Eq. (15), respectively. One can easily verify that the derivative of Eq. (19) is consistent with the sum of Eq. (12) and Eq. (14). The entire iterative procedure is outlined in Algorithm (1), where the convergence condition is:

$$\frac{|\mathcal{F}(\boldsymbol{W}_t) - \mathcal{F}(\boldsymbol{W}_{t+1})|}{|\mathcal{F}(\boldsymbol{W}_t)} < 1e-6 \,. \tag{20}$$

We will prove in Sect. 4 that the solution derived by Algorithm 1 is a stationary point of the original problem (7).

4 THEORETICAL ANALYSIS

Before giving the theoretical analysis, we will introduce several significant Lemmas.

LEMMA 2. *For any two variables $a > 0$ and $b > 0$, we have that the following inequality holds [21].*

$$a - \frac{a^2}{2b} \leq b - \frac{b^2}{2b} \,. \tag{21}$$

LEMMA 3. *Von Neumanns trace inequality [19]: For two Hermitian matrices $\boldsymbol{A} \in R^{n\times n}$ and $\boldsymbol{B} \in R^{n\times n}$, suppose the eigenvalues of them are sorted with the **same** order, then we have the following inequality.*

$$\sum_{i=1}^{n} \lambda_i(\boldsymbol{A})\lambda_{n-i+1}(\boldsymbol{B}) \leq Tr(\boldsymbol{A}\boldsymbol{B}) \leq \sum_{i=1}^{n} \lambda_i(\boldsymbol{A})\lambda_i(\boldsymbol{B}) \,. \tag{22}$$

Particularly, the equality holds if and only if we can find a matrix $\boldsymbol{U}$ that satisfies

$$\boldsymbol{U}\boldsymbol{\Lambda}_A\boldsymbol{U}^T = \boldsymbol{A}, \quad and \quad \boldsymbol{U}\boldsymbol{\Lambda}_B\boldsymbol{U}^T = \boldsymbol{B} \,, \tag{23}$$

where $\boldsymbol{\Lambda}_A$ and $\boldsymbol{\Lambda}_B$ are the ordered eigenvalue matrices.

Lemma 4. *Suppose $\boldsymbol{W}^t$ is the optimal solution of Eq. (19) at t-th iteration of Algorithm 1, then we have*

$$\begin{aligned}&\sum_{i=1} r(\lambda_i(h(\boldsymbol{W}^{t+1}))) - Tr((\boldsymbol{D}^t)^T\boldsymbol{W}^{t+1}\boldsymbol{W}^{t+1^T})\\ \leq &\sum_{i=1} r(\lambda_i(h(\boldsymbol{W}^t))) - Tr((\boldsymbol{D}^t)^T\boldsymbol{W}^t(\boldsymbol{W}^t)^T)\end{aligned} \tag{24}$$

and

$$\begin{aligned}&\sum_{i=1}^{K}(\|\boldsymbol{X}_i\boldsymbol{w}_i^{t+1}-\boldsymbol{y}_i\|_2 - \frac{\|\boldsymbol{X}_i\boldsymbol{w}_i^{t+1}-\boldsymbol{y}_i\|_2^2}{2\|\boldsymbol{X}_i\boldsymbol{w}_i^t-\boldsymbol{y}_i\|_2})\\ \leq &\sum_{i=1}^{K}(\|\boldsymbol{X}_i\boldsymbol{w}_i^t-\boldsymbol{y}_i\|_2 - \frac{\|\boldsymbol{X}_i\boldsymbol{w}_i^t-\boldsymbol{y}_i\|_2^2}{2\|\boldsymbol{X}_i\boldsymbol{w}_i^t-\boldsymbol{y}_i\|_2})\end{aligned} \tag{25}$$

Proof. In rest of this section, we replace $\lambda_i(h(\boldsymbol{W}^t))$ by λ_i^t and $\lambda_i(h(\boldsymbol{W}^{t+1}))$ by λ_i^{t+1} for ease of notations. One can easily verify that $r(x)$ is a concave function. In this case, for each i, we know

$$r(\lambda_i^{t+1}) - r'(\lambda_i^t)\lambda_i^{t+1} \leq r(\lambda_i^t) - r'(\lambda_i^t)\lambda_i^t\,. \tag{26}$$

Further, we have

$$\sum_{i=1} r(\lambda_i^{t+1}) - \sum_{i=1} r'(\lambda_i^t)\lambda_i^{t+1} \leq \sum_{i=1} r(\lambda_i^t) - \sum_{i=1} r'(\lambda_i^t)\lambda_i^t\,. \tag{27}$$

Moreover, the Lemma 3 demonstrates

$$\sum_{i=1} r'(\lambda_i^t)\lambda_i^{t+1} \leq Tr((\boldsymbol{D}^t)^T\boldsymbol{W}^{t+1}\boldsymbol{W}^{t+1^T})\,. \tag{28}$$

where the order of $\{r'(\lambda_i^t)\}_{i=1}^d$ is contrary to $\{\lambda_i^{t+1}\}_{i=1}^d$ and $\{\lambda_i^t\}_{i=1}^d$. Besides, by combining the definition of $\boldsymbol{D}^t$ and Lemma 3, we achieve

$$\begin{aligned}&Tr((\boldsymbol{D}^t)^T\boldsymbol{W}^t(\boldsymbol{W}^t)^T) = Tr(\boldsymbol{U}^t r'(\boldsymbol{\Lambda}^t)(\boldsymbol{U}^t)^T\boldsymbol{U}^t\boldsymbol{\Lambda}^t(\boldsymbol{U}^t)^T)\\ &= Tr(r'(\boldsymbol{\Lambda}^t)\boldsymbol{\Lambda}^t) = \sum_{i=1} r'(\lambda_i^t)\lambda_i^t\,.\end{aligned} \tag{29}$$

where $\boldsymbol{U}^t\boldsymbol{\Lambda}^t(\boldsymbol{U}^t)^T = \boldsymbol{W}^t(\boldsymbol{W}^t)^T$ is the eigenvalue decomposition of $h(\boldsymbol{W}^t)$, and $r'(\boldsymbol{\Lambda}^t)$ denotes a diagonal matrices whose i-th diagonal element is $r'(\lambda_i^t)$.

Combining Eq. (27), Eq. (28), and Eq. (29), we have:

$$\begin{aligned}&\sum_{i=1} r(\lambda_i^{t+1}) - Tr((\boldsymbol{D}^t)^T\boldsymbol{W}^{t+1}(\boldsymbol{W}^{t+1})^T)\\ \leq &\sum_{i=1} r(\lambda_i^t) - Tr((\boldsymbol{D}^t)^T\boldsymbol{W}^t(\boldsymbol{W}^t)^T)\end{aligned} \tag{30}$$

In addition, for each i, we know that $\|\boldsymbol{X}_i\boldsymbol{w}_i^{t+1}-\boldsymbol{y}_i\|_2 > 0$, and $\|\boldsymbol{X}_i\boldsymbol{w}_i^t-\boldsymbol{y}_i\|_2 > 0$. Hence, according to Lemma 2 we can directly obtain:

$$\begin{aligned}&\|\boldsymbol{X}_i\boldsymbol{w}_i^{t+1}-\boldsymbol{y}_i\|_2 - \frac{\|\boldsymbol{X}_i\boldsymbol{w}_i^{t+1}-\boldsymbol{y}_i\|_2^2}{2\|\boldsymbol{X}_i\boldsymbol{w}_i^t-\boldsymbol{y}_i\|_2}\\ \leq &\|\boldsymbol{X}_i\boldsymbol{w}_i^t-\boldsymbol{y}_i\|_2 - \frac{\|\boldsymbol{X}_i\boldsymbol{w}_i^t-\boldsymbol{y}_i\|_2^2}{2\|\boldsymbol{X}_i\boldsymbol{w}_i^t-\boldsymbol{y}_i\|_2}\end{aligned} \tag{31}$$

It shows that the inequality (25) holds. □

Theorem 1. *The Algorithm 1 will decrease the objective function value of Eq. (10) until it converges. That is, at $t+1$-th iteration,*

$$\mathcal{L}(\boldsymbol{W}^{t+1}) + \mu\mathcal{R}(\boldsymbol{W}^{t+1}) \leq \mathcal{L}(\boldsymbol{W}^t) + \mu\mathcal{R}(\boldsymbol{W}^t) \tag{32}$$

Proof. Since $\boldsymbol{W}^{t+1}$ is the optimal solution of Eq. (19), we can conclude that

$$\begin{aligned}&\sum_{i=1}^{K}(\frac{\|\boldsymbol{X}_i\boldsymbol{w}_i^{t+1}-\boldsymbol{y}_i\|_2^2}{2\|\boldsymbol{X}_i\boldsymbol{w}_i^t-\boldsymbol{y}_i\|_2}) + Tr((\boldsymbol{D}^t)^T\boldsymbol{W}^{t+1}\boldsymbol{W}^{t+1^T})\\ \leq &\sum_{i=1}^{K}(\frac{\|\boldsymbol{X}_i\boldsymbol{w}_i^t-\boldsymbol{y}_i\|_2^2}{2\|\boldsymbol{X}_i\boldsymbol{w}_i^t-\boldsymbol{y}_i\|_2}) + Tr((\boldsymbol{D}^t)^T\boldsymbol{W}^t(\boldsymbol{W}^t)^T)\,.\end{aligned} \tag{33}$$

Then, by summing Eq. (24), Eq. (25), and Eq. (33) we achieve

$$\begin{aligned}&\sum_{i=1}^{K}(\|\boldsymbol{X}_i\boldsymbol{w}_i^{t+1}-\boldsymbol{y}_i\|_2) + \sum_{i=1} r(\lambda_i(h(\boldsymbol{W}^{t+1})))\\ \leq &\sum_{i=1}^{K}(\|\boldsymbol{X}_i\boldsymbol{w}_i^t-\boldsymbol{y}_i\|_2) + \sum_{i=1} r(\lambda_i(h(\boldsymbol{W}^t)))\,.\end{aligned} \tag{34}$$

It shows that the Theorem 1 holds.

□

Theorem 2. *The solution sequence $\{\boldsymbol{W}^t\}$ generated by Algorithm 1 is a convergent sequence, i.e.,*

$$\lim_{t\to\infty} \|\boldsymbol{W}^t - \boldsymbol{W}^{t+1}\| = 0\,. \tag{35}$$

Proof. We first prove that the solution sequence $\{\boldsymbol{W}^t\}$ is bounded. For any t, we have

$$\sum_i r(\lambda_i^t) \leq \mathcal{P}(\boldsymbol{W}^t) \leq \mathcal{P}(\boldsymbol{W}^1)\,. \tag{36}$$

It demonstrates that $\{h(\boldsymbol{W}^t)\}$ is bounded. Further, we can conclude that $\{\boldsymbol{W}^t\}$ and $\{\boldsymbol{D}^t\}$ are bounded. Next, we prove the Eq. (35). As $\boldsymbol{W}^t$ is the optimal solution of Eq. (19) at t-th iteration, we have:

$$\begin{aligned}&\mathcal{P}(\boldsymbol{W}^t) - \mathcal{P}(\boldsymbol{W}^{t+1})\\ &= \mu(\mathcal{R}(\boldsymbol{W}^t) - \mathcal{R}(\boldsymbol{W}^{t+1})) + \mathcal{L}(\boldsymbol{W}^t) - \mathcal{L}(\boldsymbol{W}^{t+1})\,.\end{aligned} \tag{37}$$

For the first term, we know that

$$\begin{aligned}&\mathcal{R}(\boldsymbol{W}^t) - \mathcal{R}(\boldsymbol{W}^{t+1})\\ &= \sum_i r(\lambda_i^t) - \sum_i r(\lambda_i^{t+1}) \geq \sum_i r'(\lambda_i^t)(\lambda_i^t - \lambda_i^{t+1})\\ &= Tr(r'(\boldsymbol{\Lambda}^t)(\boldsymbol{\Lambda}^t - \boldsymbol{\Lambda}^{t+1}))\\ &\geq Tr((\boldsymbol{D}^t)^T(\boldsymbol{W}^t(\boldsymbol{W}^t)^T - \boldsymbol{W}^{t+1}(\boldsymbol{W}^{t+1})^T))\\ &= Tr((\boldsymbol{D}^t)^T(\boldsymbol{W}^t - \boldsymbol{W}^{t+1})(\boldsymbol{W}^t - \boldsymbol{W}^{t+1})^T)\\ &+ 2Tr((\boldsymbol{D}^t)^T\boldsymbol{W}^{t+1}(\boldsymbol{W}^t - \boldsymbol{W}^{t+1})^T)\\ &= Tr((\boldsymbol{D}^t)^T(\boldsymbol{W}^t - \boldsymbol{W}^{t+1})(\boldsymbol{W}^t - \boldsymbol{W}^{t+1})^T)\\ &+ 2\sum_i((\boldsymbol{w}_i^t - \boldsymbol{w}_i^{t+1})^T(\boldsymbol{D}^t)^T\boldsymbol{w}_i^{t+1})\,.\end{aligned} \tag{38}$$

Note that we derive the second inequality by utilizing the Lemma 3, where vector $r'(\boldsymbol{\Lambda}^t)$ is increasing order when vector $\boldsymbol{\Lambda}^t$ is descending order.

For the second term, letting $\boldsymbol{E} = (\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_K)$, where $\boldsymbol{e}_i = \boldsymbol{X}_i\boldsymbol{w}_i - \boldsymbol{y}_i$, we can obtain:

$$\begin{aligned}
&\mathcal{L}(\boldsymbol{W}^t) - \mathcal{L}(\boldsymbol{W}^{t+1}) = \boldsymbol{E}_{2,1}^t - \boldsymbol{E}_{2,1}^{t+1} \\
&\geq \frac{1}{2}Tr(((\boldsymbol{E}^t)^T\boldsymbol{E}^t - (\boldsymbol{E}^{t+1})^T\boldsymbol{E}^{t+1})\boldsymbol{K}^t) \\
&= \frac{1}{2}Tr((\boldsymbol{E}^t - \boldsymbol{E}^{t+1})^T(\boldsymbol{E}^t - \boldsymbol{E}^{t+1})\boldsymbol{K}^t) \\
&+ Tr((\boldsymbol{E}^t - \boldsymbol{E}^{t+1})^T\boldsymbol{E}^{t+1}\boldsymbol{K}^t) \\
&= \frac{1}{2}Tr((\boldsymbol{E}^t - \boldsymbol{E}^{t+1})^T(\boldsymbol{E}^t - \boldsymbol{E}^{t+1})\boldsymbol{K}^t) \\
&+ \sum_{i=1}^{K}(\boldsymbol{e}_i^t - \boldsymbol{e}_i^{t+1})^T\boldsymbol{e}_i^{t+1}K_{ii}^t\,,
\end{aligned} \tag{39}$$

where $\boldsymbol{K}^t$ is a diagonal matrix, and $K_{ii}^t = \frac{1}{\|\boldsymbol{e}_i\|_2}$. Since each column of $\boldsymbol{W}_{k+1}$ is the optimal solution of Eq. (18), the following equality holds.

$$v_i^t\boldsymbol{X}_i^T(\boldsymbol{X}_i\boldsymbol{w}_i^{t+1} - \boldsymbol{y}_i) + 2\mu(\boldsymbol{D}^t)^T\boldsymbol{w}_i^{t+1} = 0\,. \tag{40}$$

Multiplying by $(\boldsymbol{w}_i^t - \boldsymbol{w}_i^{t+1})^T$ on both sides of Eq. (40), we have:

$$\begin{aligned}
&v_i^t(\boldsymbol{X}_i\boldsymbol{w}_i^t - \boldsymbol{X}_i\boldsymbol{w}_i^{t+1})^T(\boldsymbol{X}_i\boldsymbol{w}_i^{t+1} - \boldsymbol{y}_i) \\
&+ 2\mu(\boldsymbol{w}_i^t - \boldsymbol{w}_i^{t+1})^T(\boldsymbol{D}^t)^T\boldsymbol{w}_i^{t+1} = 0 \\
&\Rightarrow v_i^t(\boldsymbol{e}_i^t - \boldsymbol{e}_i^{t+1})\boldsymbol{e}_i^{t+1} \\
&+ 2\mu(\boldsymbol{w}_i^t - \boldsymbol{w}_i^{t+1})^T(\boldsymbol{D}^t)^T\boldsymbol{w}_i^{t+1} = 0\,.
\end{aligned} \tag{41}$$

It demonstrates that

$$v_i^t(\boldsymbol{e}_i^t - \boldsymbol{e}_i^{t+1})\boldsymbol{e}_i^{t+1} = -2\mu(\boldsymbol{w}_i^t - \boldsymbol{w}_i^{t+1})^T(\boldsymbol{D}^t)^T\boldsymbol{w}_i^{t+1} \tag{42}$$

Recalling the definition of v_i^t, we find that $v_i^t = K_{ii}^t$. Further, by combining the Eq. (38), Eq. (39), and Eq. (42), we achieve

$$\begin{aligned}
&\mathcal{P}(\boldsymbol{W}^t) - \mathcal{P}(\boldsymbol{W}^{t+1}) \\
&= \mu(\mathcal{R}(\boldsymbol{W}^t) - \mathcal{R}(\boldsymbol{W}^{t+1})) + \mathcal{L}(\boldsymbol{W}^t) - \mathcal{L}(\boldsymbol{W}^{t+1}) \\
&\geq \mu Tr((\boldsymbol{D}^t)^T(\boldsymbol{W}^t - \boldsymbol{W}^{t+1})(\boldsymbol{W}^t - \boldsymbol{W}^{t+1})^T) \\
&+ \frac{1}{2}Tr((\boldsymbol{E}^t - \boldsymbol{E}^{t+1})^T(\boldsymbol{E}^t - \boldsymbol{E}^{t+1})\boldsymbol{K}^t)\,.
\end{aligned} \tag{43}$$

In addition, the Lemma 3 shows that:

$$\begin{aligned}
&\mu Tr((\boldsymbol{D}^t)^T(\boldsymbol{W}^t - \boldsymbol{W}^{t+1})(\boldsymbol{W}^t - \boldsymbol{W}^{t+1})^T) \\
&\geq \mu\sum_{i}^{d}\lambda_{d-i+1}(\boldsymbol{D}^t)\lambda_i((\boldsymbol{W}^t - \boldsymbol{W}^{t+1})(\boldsymbol{W}^t - \boldsymbol{W}^{t+1})^T) \\
&\geq \mu\alpha\|\boldsymbol{W}^t - \boldsymbol{W}^{t+1}\|_F^2\,,
\end{aligned} \tag{44}$$

and

$$\begin{aligned}
&Tr((\boldsymbol{E}^t - \boldsymbol{E}^{t+1})^T(\boldsymbol{E}^t - \boldsymbol{E}^{t+1})\boldsymbol{K}^t) \\
&\geq \sum_{i}^{d}\lambda_{d-i+1}(\boldsymbol{K}^t)\lambda_i((\boldsymbol{E}^t - \boldsymbol{E}^{t+1})(\boldsymbol{E}^t - \boldsymbol{E}^{t+1})^T) \\
&\geq \beta\|\boldsymbol{E}^t - \boldsymbol{E}^{t+1}\|_F^2\,,
\end{aligned} \tag{45}$$

where $\alpha > 0$ and $\beta > 0$ are the least eigenvalues of $\boldsymbol{D}^t$ and $\boldsymbol{K}^t$, respectively. By summing Eq. (44) and Eq. (45), we get:

$$\begin{aligned}
&\mathcal{P}(\boldsymbol{W}^t) - \mathcal{P}(\boldsymbol{W}^{t+1}) \\
&\geq \mu\alpha\|\boldsymbol{W}^t - \boldsymbol{W}^{t+1}\|_F^2 + \beta\|\boldsymbol{E}^t - \boldsymbol{E}^{t+1}\|_F^2\,.
\end{aligned} \tag{46}$$

Further using the observation that $P(\boldsymbol{W}^t) > 0$ and the Theorem 1, we can know that $\{P(\boldsymbol{W}^t)\}$ is a convergent sequence, i.e., $\lim_{t\to\infty}\|\mathcal{P}(\boldsymbol{W}^t) - \mathcal{P}(\boldsymbol{W}^{t+1})\|_F^2 = 0$. Naturally, we can conclude that

$$\begin{aligned}
&\lim_{t\to\infty}\|\boldsymbol{W}^t - \boldsymbol{W}^{t+1}\|_F^2 = 0 \\
&\lim_{t\to\infty}\|\boldsymbol{E}^t - \boldsymbol{E}^{t+1}\|_F^2 = 0\,.
\end{aligned} \tag{47}$$

□

Theorem 3. *The final solution provided by Algorithm 1 (a limit point of sequence $\{\boldsymbol{W}^t\}$) is a stationary point of the original problem* (10).

Proof. According to Eq. (47), we know that the weighted matrices $\boldsymbol{D}$ and $\boldsymbol{K}$ will keep constant when Algorithm 1 converges. In addition, the first-order optimal condition of problem (10) is:

$$\mathbf{0} = \frac{\partial\mathcal{P}(\boldsymbol{W})}{\partial\boldsymbol{W}} = \frac{\partial\mathcal{L}(\boldsymbol{W})}{\partial\boldsymbol{W}} + \mu\frac{\partial\mathcal{R}(\boldsymbol{W})}{\partial\boldsymbol{W}}\,. \tag{48}$$

And, it is consistent with the derivative of problem (19) when $\boldsymbol{W}$ converges to a limit point $\boldsymbol{W}^*$. Such that the limit point $\boldsymbol{W}^*$ satisfies the first-order optimal conditions of problem (19) and problem (10) simultaneously. Hence, it is a stationary point of problem (10). □

4.1 Extension to a general algorithm

It is worth noting that the optimization method used in this paper can be used to solve various rank minimization problems. A general formulation for them is:

$$\min_{\boldsymbol{W}\in\mathcal{C}}\mathcal{L}(\boldsymbol{W}) + \mu\sum_{i=1}^{d}r(\lambda_i(h(\boldsymbol{W})))\,. \tag{49}$$

where $r(x)$ is an arbitrary concave function satisfying $r'(x) \geq 0$ when $x > 0$. And, $r'(x)$ represents the derivative of $r(x)$ at x. For different tasks, loss function $\mathcal{L}(\boldsymbol{W})$ have different formulations. The details for solving the problem (49) has been outlined in Algorithm 2. The theoretical analysis is similar to the content mentioned above.

5 NUMERICAL SIMULATIONS

The performance evaluation consists of the following two parts. Firstly, we compare the performance of NC-CMTL and NC-MTL on synthetic data. The aim of implementing this experiment is to verify the advantage of introducing calibration. Then, we evaluate the performance of NC-CMTL with serval state-of-the-art multi-task learning methods on two benchmark datasets.

Algorithm 2 Reweighted method for solving problem (49)

Input: The known information
Output: The matrix $\boldsymbol{W}$
Initialization: Initialize $\boldsymbol{W} \in \mathcal{C}$.
while not convergent **do**
1. Eigenvalue decomposition: $h(\boldsymbol{W}) = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T$;
2. Update $\boldsymbol{D}$ by $\boldsymbol{D} = \sum_{i=1} r'(\lambda_i)\boldsymbol{u}_i\boldsymbol{u}_i^T$;
3. Update $\boldsymbol{W}$ by solving

$$\min_{\boldsymbol{W}\in\mathcal{C}} \mathcal{L}(\boldsymbol{W}) + \mu Tr(\boldsymbol{D}^T h(\boldsymbol{W}))\,. \tag{50}$$

4. Convergence checking.
end while

Table 3: Numerous comparison results between NC-MTL and NC-CMTL on synthetic data. Here d_i represents the noise level of different tasks. The reported results are the average and the standard errors of 100 times simulations, respectively. NC-CMTL significantly outperforms NC-MTL when only several tasks are corrupted by gross noise ($\boldsymbol{d}_3$,$\boldsymbol{d}_4$).

Noise	Method	nMSE	E.E.
$\boldsymbol{d}_1$	NC-MTL	0.8406(0.1621)	0.1247(0.0223)
	NC-CMTL	0.7977(0.1537)	0.1236(0.0211)
$\boldsymbol{d}_2$	NC-MTL	0.9982(0.1982)	0.1237(0.0319)
	NC-CMTL	0.9992(0.1983)	0.1237(0.0328)
$\boldsymbol{d}_3$	NC-MTL	8.2795(1.7524)	0.3721(0.0721)
	NC-CMTL	0.6001(0.1226)	0.0947(0.0126)
$\boldsymbol{d}_4$	NC-MTL	4.3439(0.9857)	0.3641(0.0695)
	NC-CMTL	0.2035(0.0234)	0.0783(0.0158)

5.1 Calibration Investigation

The synthetic data is generated by combining the schemes used in [18] and [34]. Concretely, we use the following procedure to generate the data. Here, we fix $K = 101$, $n_i = 400$ $(i = 1, 2, \ldots, K)$, $d = 200$, $r = 3$ and $\sigma_{max} = 2$.

- Generate training data $\{\boldsymbol{X}_i\}_1^K$. Each row of $\boldsymbol{X}_i \in R^{n_i\times d}$ is generated from a d-dimensional $\mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma})$, where $\Sigma_{i,i} = 1$ and $\Sigma_{i,j} = 0.5$ otherwise.
- Generate weighted coefficient matrix $\boldsymbol{W} \in R^{d\times K}$ with rank r. We generate two matrices $\boldsymbol{U} \in R^{d\times r}$ and $\boldsymbol{V} \in R^{K\times r}$ to obtain the weighted coefficient matrix $\boldsymbol{W} = \boldsymbol{U}\boldsymbol{V}^T$, and all elements of $\boldsymbol{U}$ and $\boldsymbol{V}$ are sampled from $\mathcal{N}(0, 0.05)$.
- Generate noise matrix $\boldsymbol{Z} \in R^{n\times K}$. The noise matrix $\boldsymbol{Z}$ is a product of $\boldsymbol{S}$ and $\boldsymbol{D}$, i.e., $\boldsymbol{Z} = \boldsymbol{S}\boldsymbol{D}$, where $\boldsymbol{S} \in R^{n\times K}$ and the elements of it are sampled from $\mathcal{N}(0, 1)$. In addition, $\boldsymbol{D} \in R^{K\times K}$ is a diagonal matrix whose diagonal elements are equal to one of the following vectors.

$$\begin{aligned}
\boldsymbol{d}_1 &= \sigma_{max}(2^{0/100}, 2^{-3/100}, \ldots, 2^{-297/100}, 2^{-300/100});\\
\boldsymbol{d}_2 &= \sigma_{max}(1, 1, \ldots, 1);\\
\boldsymbol{d}_3 &= \sigma_{max}(2^{0/25}, 2^{-3/25}, \ldots, 2^{-297/25}, 2^{-300/25});\\
\boldsymbol{d}_4 &= \sigma_{max}(2^{0/4}, 2^{-1/4}, \ldots, 2^{-99/4}, 2^{-100/4})\,,
\end{aligned} \tag{51}$$

where $\boldsymbol{d}_1$ represents the case that all tasks are corrupted by gross noise with different noise degree; $\boldsymbol{d}_2$ represents the case that all tasks are corrupted by gross noise with same degree; $\boldsymbol{d}_3$ represents the case that only several tasks are corrupted by gross noise, and the residual tasks are corrupted by light noise; $\boldsymbol{d}_4$ represents the case that only several tasks are corrupted by gross noise, and the residual tasks are noiseless. Note that 60% data are selected randomly serve as training data, and the residual server as test data. In addition, the best regularization parameter μ is selected by cross validation. And the predication precision is evaluated by nMSE, which is defined as:

$$nMSE = \frac{MSE}{\|\boldsymbol{y}_i\|_2}\,, \tag{52}$$

where $MSE = \|\boldsymbol{X}_i\boldsymbol{w}_i - \boldsymbol{y}_i\|_2^2/n_i$, $var(\boldsymbol{y}_i)$ is the variance of $\boldsymbol{y}_i$. Since the ground truth of the weighted coefficient matrix $\boldsymbol{W}^*$ is known, we introduce an additional criteria $E.E$, which is denoted as:

$$E.E. = \frac{\|\boldsymbol{W} - \boldsymbol{W}^*\|_F^2}{K}\,. \tag{53}$$

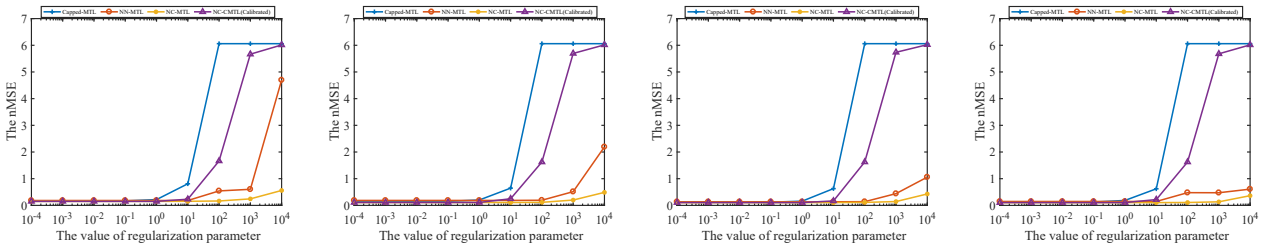
We summarize the final experimental results, the average errors and standard errors of 100 times simulations, in Tab. 3.

Observing Tab. 3, we can find that NC-CMTL obtains similar precision to NC-MTL on the setting that all tasks are corrupted by gross noise ($\boldsymbol{d}_1$ and $\boldsymbol{d}_2$). While NC-CMTL provides a huge advantage over NC-MTL on the setting that some tasks have small noise or are noiseless ($\boldsymbol{d}_3$ and $\boldsymbol{d}_4$). Further, the performance of NC-CMTL on $\boldsymbol{d}_4$ has a significant improvement comparing to $\boldsymbol{d}_3$. Hence, we can conclude that introducing calibration for each task is necessary for multi-task learning.

5.2 Experiments on real data

In this section, we evaluate NC-CMTL on two benchmark datasets including SCHOOL and SARCOS data, and measure the predication precision by nMSE. Note that in SCHOOL data different tasks have different training samples, while in SARCOS data all tasks share a common training dataset.

For reference, we compare NC-CMTL with five state-of-the-art multi-task regression learning algorithms including: NN-MTL [9], Capped-MTL [13], and Robust-MTL [10], three low rank based models; Lasso [26] and Calibrated-MTL [12], two feature selection based models. Note that Calibrated-MTL [12] is a representative method for multi-task feature learning with calibration. As MRC [18] is developed for solving the problem that all tasks share a common training data, we omit it in this paper. The codes of NN-MTL, Robust-MTL, CMTL and Calibrated-MTL are provided by Malsar [34], and the code of Capped-MTL is provided by its authors.



(a) $n = 100$ **(b)** $n = 200$ **(c)** $n = 300$ **(d)** $n = 500$

Figure 2: The performance of our methods (NC-MTL as wll as NC-CMTL) and two low rank based algorithms (NN-MTL as well as Capped-MTL) on SARCOS Data. Here, we vary the regularization parameter μ in the set $\{1e-4, 1e-3, \ldots, 1e+4\}$. NC-CMTL outperforms other methods including NC-MTL when μ takes a small value.

Table 4: The experimental results of all test methods on benchmark data. The evaluation criteria are nMSE with standard deviation. The parameters of alternative methods are selected by 5-fold cross-validation.

Method	training ratio	Future Based		Low Rank Based			
		Lasso	Calibrated-MTL	NN-MTL	Capped-MTL	Robust-MTL	NC-CMTL
SCHOOL	10%	1.0238(0.0018)	0.9512(0.0022)	0.9364(0.0034)	0.9215(0.0037)	0.9134(0.0040)	**0.8864(0.0035)**
	20%	0.8729(0.0041)	0.8407(0.0064)	0.8331(0.0028)	0.8223(0.0037)	0.8115(0.0056)	**0.7822(0.0048)**
	30%	0.8132(0.0021)	0.8122(0.0031)	0.7870(0.0016)	0.7692(0.0024)	0.7612(0.0033)	**0.7539(0.0022)**
SARCOS	100	0.1621(0.0155)	0.1572(0.0022)	0.1531(0.0024)	0.1516(0.0017)	0.1472(0.0026)	**0.1375(0.0015)**
	200	0.1569(0.0061)	0.1523(0.0143)	0.1491(0.0021)	0.1447(0.0029)	0.1368(0.0031)	**0.1116(0.0022)**
	300	0.1547(0.0038)	0.1507(0.0062)	0.1456(0.0025)	0.1438(0.0033)	0.1324(0.0027)	**0.1106(0.0018)**

5.2.1 Parameter selection. Before reporting the results of all alternative methods, we prefer to test the effect caused by regularization parameter μ to the performance of our methods (NC-MTL as wll as NC-CMTL) and two low rank based algorithms (NN-MTL as well as Capped-MTL). The value of μ varies in the set $\{1e-4, 1e-3, \ldots, 1e+4\}$. Here, we do not implement Robust-MTL, because it have two parameters needed to be adjust. The final comparison results are presented in Fig 2, which shows that NC-CMTL generally outperforms NN-MTL as well as Capped-MTL when μ takes a small value. While μ taking a large value leads to the regularization parameter $(\frac{2\mu}{v_i})$ of Eq. (18) being large, such that NC-CMTL will have a overfitting problem. The reason is also suitable for Capped-norm. The regularization parameter of NC-MTL is fixed, thus a large value of μ will have a small effect on its performance. In practice, we suggest users select a small value for μ. In addition, the performance of NC-MTL is more stable than NN-MTL when the number of training data is small, which shows the advantage of proposed non-convex regularizer.

5.2.2 Comparison with all alternative methods. We report the results of all alternative approaches and our method in Tab. 4. The regularization parameters are determined by 5-fold Cross validation for all methods mentioned above. As the noise degree of real data is unknown, we omit the results of NC-MTL. According the results shown in Tab 4, we can conclude that low rank based methods generally outperform feature selection based approaches, and learning multiple tasks jointly can significantly improve the prediction precision comparing to learning multiple tasks separately (Lasso algorithm). Further, NC-CMTL provides a huge improvement over NN-MTL and other low rank based algorithms.

5.3 Computational Complexity

In this section, we analysis the computational complexity of Algorithm 1. The main time consuming of Algorithm 1 occur in step 1 and step 4. In step 1, as $h(\boldsymbol{W})$ is a symmetric matrix with size $d \times d$, the time complexity of performing Eigenvalue Decomposition is $O(d^3)$ when $d < K$. While the Eigenvalue Decomposition can be replaced by SVD to $\boldsymbol{W}$ when $d > K$. The time complexity of performing SVD is $O(dK^2)$. In step 4, we need to solve the problem (18) K times, the time complexity is $O(Kd^3)$. Hence the total time complexity of Algorithm 1 is $O((K+1)d^3)$ $(d < K)$ or $O(dK^2 + Kd^3)$ $(d > K)$.

To the best of our knowledge, none work has been provided to the theoretical analysis of the convergence speed for re-weighted method. In practice, however, we find that Algorithm 1 will converge within $5-10$ times iterations as presented in Fig. 3. This test is conducted on School Data with 20% training data. We report the time consuming of four low rank based algorithms in Tab. 5. Obviously, the time consuming of NC-CMTL is far less than others. Hence, NC-CMTL can cope with large scale data efficiently.

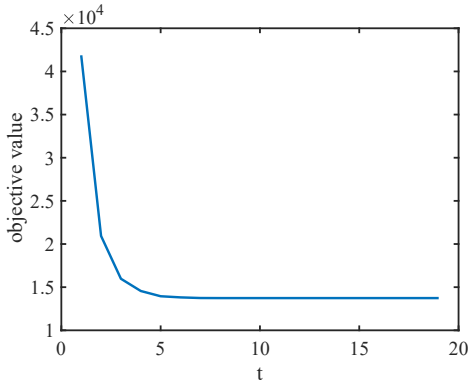


Figure 3: The convergence analysis of our method on SCHOOL with 20% training data. Obviously, Algorithm 1 will converge within $5-10$ times iterations.

Table 5: The time consuming of four algorithms based on low rank regularization. The proportion of training data are 20% and 300, respectively.

Data	NN-MTL	Capped-MTL	Robust-MTL	NC-CMTL
SCHOOL	$0.32s$	$1.52s$	$0.93s$	$\mathbf{0.15s}$
SARCOS	$0.23s$	$1.16s$	$0.87s$	$\mathbf{0.10s}$

6 CONCLUSION

This paper mainly solve the problem of Multi-Task regression Learning with calibration. To reduce the gap between convex relaxation and original rank minimization problem, we introduce a new non-convex regularizer. Furthermore, we replace the least squares loss function by square-root loss function to cope with the problem that different tasks have different noise degree. As proposed model is non-convex and non-smooth, we construct an efficient re-weighted method to solve it. Particularly, the constructed optimization method has sound convergence guarantee. Particularly, the regularizer, the optimization method and relevant theoretical analysis are suit for other rank minimization problems such as subspace clustering and Robust principal component analysis. Investigating the performance of regularizer as well as optimization method on other machine learning fields is a direction in our future work.

ACKNOWLEDGMENTS

The authors would like to thank Dr. Lei Han for providing the MATLAB code of the *Capped-MTL* method. The authors would also like to thank the Dr.Jiayu Zhou for providing MATLAB softwar *Malsar*. This work was supported in part by the National Natural Science Foundation of China grant under numbers 61772427, 61751202 and 61761130079.

REFERENCES

[1] Qi An, Chunping Wang, Ivo Shterev, Eric Wang, Lawrence Carin, and David B Dunson. 2008. Hierarchical kernel stick-breaking process for multi-task image analysis. In *Proceedings of the 25th international conference on Machine learning*. ACM, 17–24.

[2] Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research* 6, Nov (2005), 1817–1853.

[3] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. 2008. Convex multi-task feature learning. *Machine Learning* 73, 3 (2008), 243–272.

[4] Jing Bai, Ke Zhou, Guirong Xue, Hongyuan Zha, Gordon Sun, Belle Tseng, Zhaohui Zheng, and Yi Chang. 2009. Multi-task learning for learning to rank in web search. In *Proceedings of the 18th ACM conference on Information and knowledge management*. ACM, 1549–1552.

[5] Emmanuel J Candès and Terence Tao. 2010. The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory* 56, 5 (2010), 2053–2080.

[6] Emmanuel J Candes, Michael B Wakin, and Stephen P Boyd. 2008. Enhancing sparsity by reweighted 1 minimization. *Journal of Fourier analysis and applications* 14, 5 (2008), 877–905.

[7] Olivier Chapelle, Pannagadatta Shivaswamy, Srinivas Vadrevu, Kilian Weinberger, Ya Zhang, and Belle Tseng. 2010. Multi-task learning for boosting with application to web search ranking. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1189–1198.

[8] Jianhui Chen, Ji Liu, and Jieping Ye. 2012. Learning incoherent sparse and low-rank patterns from multiple tasks. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 5, 4 (2012), 22.

[9] Jianhui Chen, Lei Tang, Jun Liu, and Jieping Ye. 2009. A convex formulation for learning shared structures from multiple tasks. In *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM, 137–144.

[10] Jianhui Chen, Jiayu Zhou, and Jieping Ye. 2011. 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*. ACM, 42–50.

[11] Ingrid Daubechies, Ronald DeVore, Massimo Fornasier, and C Sinan Güntürk. 2010. Iteratively reweighted least squares minimization for sparse recovery. *Communications on Pure and Applied Mathematics* 63, 1 (2010), 1–38.

[12] Pinghua Gong, Jiayu Zhou, Wei Fan, and Jieping Ye. 2014. Efficient multi-task feature learning with calibration. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 761–770.

[13] Lei Han and Yu Zhang. 2016. Multi-Stage Multi-Task Learning with Reduced Rank.. In *AAAI*. 1638–1644.

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

[15] Pratik Jawanpuria and J. Saketha Nath. 2012. A Convex Feature Learning Formulation for Latent Task Structure Discovery. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*.

[16] Abhishek Kumar and Hal Daume III. 2012. Learning task grouping and overlap in multi-task learning. *arXiv preprint arXiv:1206.6417* (2012).

[17] Giwoong Lee, Eunho Yang, and Sung Hwang. 2016. Asymmetric multi-task learning based on task relatedness and loss. In *International Conference on Machine Learning*. 230–238.

[18] Han Liu, Lie Wang, and Tuo Zhao. 2014. Multivariate regression with calibration. In *Advances in neural information processing systems*. 127–135.

[19] Albert W Marshall, Ingram Olkin, and Barry C Arnold. 1979. *Inequalities: theory of majorization and its applications*. Vol. 143. Springer.

[20] Karthik Mohan and Maryam Fazel. 2012. Iterative reweighted algorithms for matrix rank minimization. *Journal of Machine Learning Research* 13, Nov (2012), 3441–3473.

[21] Feiping Nie, Heng Huang, Xiao Cai, and Chris H Ding. 2010. Efficient and robust feature selection via joint $\ell_{2,1}$-norms minimization. In *Advances in neural information processing systems*. 1813–1821.

[22] Feiping Nie, Jianjun Yuan, and Heng Huang. 2014. Optimal mean robust principal component analysis. In *International Conference on Machine Learning*. 1062–1070.

[23] Chong Peng, Zhao Kang, Huiqing Li, and Qiang Cheng. 2015. Subspace clustering using log-determinant rank approximation. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 925–934.

[24] Kaare Brandt Petersen, Michael Syskind Pedersen, et al. 2008. The matrix cookbook. *Technical University of Denmark* 7, 15 (2008), 510.
[25] Ting Kei Pong, Paul Tseng, Shuiwang Ji, and Jieping Ye. 2010. Trace norm regularization: Reformulations, algorithms, and multi-task learning. *SIAM Journal on Optimization* 20, 6 (2010), 3465–3489.
[26] Robert Tibshirani. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)* (1996), 267–288.
[27] Hua Wang, Feiping Nie, Heng Huang, Shannon Risacher, Chris Ding, Andrew J Saykin, Li Shen, et al. 2011. Sparse multi-task regression and feature selection to identify brain imaging predictors for memory performance. In *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 557–562.
[28] Debing Zhang, Yao Hu, Jieping Ye, Xuelong Li, and Xiaofei He. 2012. Matrix completion by truncated nuclear norm regularization. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2192–2199.
[29] Jian Zhang, Zoubin Ghahramani, and Yiming Yang. 2006. Learning multiple related tasks using latent independent component analysis. In *Advances in neural information processing systems*. 1585–1592.
[30] Jian Zhang, Zoubin Ghahramani, and Yiming Yang. 2008. Flexible latent variable models for multi-task learning. *Machine Learning* 73, 3 (2008), 221–242.
[31] Kai Zhang, Joe W Gray, and Bahram Parvin. 2010. Sparse multi-task regression for identifying common mechanism of response to therapeutic targets. *Bioinformatics* 26, 12 (2010), i97–i105.
[32] Yu Zhang and Qiang Yang. 2017. A survey on multi-task learning. *arXiv preprint arXiv:1707.08114* (2017).
[33] Yu Zhang and Dit-Yan Yeung. 2010. Multi-task warped gaussian process for personalized age estimation. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2622–2629.
[34] Jiayu Zhou, Jianhui Chen, and Jieping Ye. 2011. Malsar: Multi-task learning via structural regularization. *Arizona State University* 21 (2011).
[35] Jiayu Zhou, Lei Yuan, Jun Liu, and Jieping Ye. 2011. A multi-task learning formulation for predicting disease progression. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 814–822.