Pattern Recognition 161 (2025) 111213

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Multi-level network Lasso for multi-task personalized learning

Jiankun Wang [a,*], Luhuan Fei [b], Lu Sun [b]

[a] Department of Computer Science and Engineering, College of Engineering, Michigan State University, East Lansing, United States of America
[b] School of Information Science and Technology, ShanghaiTech University, Shanghai, China

ARTICLE INFO

Keywords:
Multi-level network Lasso
Personalized learning
Multi-task learning

ABSTRACT

We propose the multi-level network Lasso, which aims to overcome the key limitations of existing personalized learning methods, such as ignoring sample homogeneity or heterogeneity, and over-parametrization. Multi-level network Lasso learns both sample-common model and sample-specific model, that are succinct and interpretable in the sense that model parameters are shared across neighboring samples based on only a subset of relevant features. To apply personalized learning in multi-task scenarios, we further extend the multi-level network Lasso for multi-task personalized learning by learning underlying task groups in the feature subspace. Additionally, we investigate a family of the multi-level network Lasso based on the ℓ_p quasi-norm ($0 < p < 1$), that helps prevent over-penalization on large group outliers. An alternating algorithm is developed to efficiently solve the proposed optimization problem. Experimental results on synthetic and real-world datasets demonstrate the effectiveness of the proposed method.

1. Introduction

Personalized learning (PL) aims to learn sample-specific models for individual samples (*sample heterogeneity*), in order to improve personalized generalization ability. For example, personalized medicine exploits patient heterogeneity to build patient-specific predictive models. PL is different from conventional supervised learning, which usually assumes that all samples share an identical model (*sample homogeneity*). One common issue of PL is that the excessive number of parameters makes it prone to overfitting. To alleviate this problem, several methods have been proposed in the literature. The network Lasso type methods [1–3] are proposed by borrowing strength from the neighborhoods of samples in the similarity network (graph). In addition, projecting the models into a shared subspace can further control model size [4]. PL can also benefit from the additional covariates [5] and pre-trained anchor models [6]. Despite of the success in various practical applications [7,8], existing methods usually ignore the sample homogeneity and the existence of noisy features, that may harm generalization performance.

In multi-task learning (MTL), multiple correlated tasks are learned jointly by sharing knowledge across tasks, leading to the improved predictive ability [9]. The major challenge in MTL is to adequately capture task correlations. One common way is to assume that the tasks are related by shared latent subspace, such as feature learning approaches [10,11] and low-rank approaches [12,13]. However, this assumption is restrictive in practice, as tasks may have their own specific features in addition to the shared ones. To this end, the decomposition approaches are proposed, which decompose the parameters into task-common and task-specific parts via summation [14] or multiplication [15,16]. Recently, task clustering approaches [17,18] have been proposed to learn task group structure and model parameters simultaneously. Extensive experimental results have shown these MTL approaches' superiority against baseline learners [9]. However, they typically estimate a global model shared by all samples within each task, preserving only sample homogeneity and ignoring sample heterogeneity—a critical aspect that PL is designed to capture. Integrating PL into the MTL setting is necessary to fully exploit its capabilities for handling diverse sample characteristics.

Recently, multi-task personalized learning (MTPL) has been proposed by considering both sample homogeneity and heterogeneity in multi-task scenarios. In [19], the first MTPL method is developed based on the network Lasso and low-rank matrix decomposition. It has achieved much success in a variety of real-world problems compared with the MTL and PL approaches. However, two challenges need to be addressed in the work. (1) Feature selection should be applied to choose interpretable features. The dense model learned by the existing method lacks interpretation and may suffer from performance degradation due to the existence of noisy features in real-world scenarios, especially with limited data. (2) The latent task group structure should be considered to improve generalization. The existing method saves task correlations by jointly building a common model for multiple tasks, that is difficult to explicitly capture the relationship when tasks indeed belong to different latent groups.

* Corresponding author.
E-mail addresses: wangj306@msu.edu (J. Wang), feilh2023@shanghaitech.edu.cn (L. Fei), sunlu1@shanghaitech.edu.cn (L. Sun).

https://doi.org/10.1016/j.patcog.2024.111213
Received 6 May 2024; Received in revised form 17 November 2024; Accepted 19 November 2024
Available online 26 November 2024

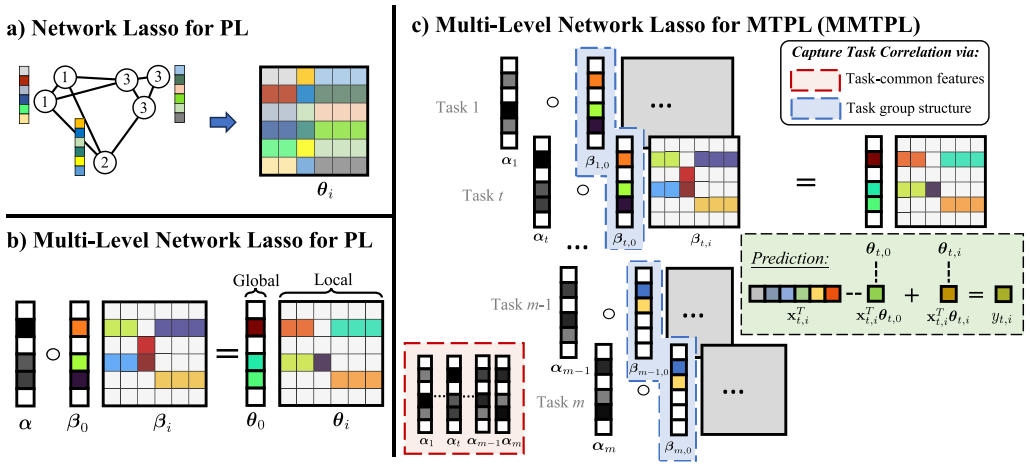


Fig. 1. Illustration of the MMTPL framework. (a) Network Lasso for PL: It leverages graph information for clustering and learning to derive personalized models. (b) Multi-Level Network Lasso for PL: We propose to decompose the personalized model for sample i into a global model $\boldsymbol{\theta}_0$ and a local model $\boldsymbol{\theta}_i$ to save homogeneity and heterogeneity, respectively. These models are further decomposed by $\boldsymbol{\theta}_0 = \boldsymbol{\alpha} \circ \boldsymbol{\beta}_0$ and $\boldsymbol{\theta}_i = \boldsymbol{\alpha} \circ \boldsymbol{\beta}_i$, with $\{\boldsymbol{\beta}_i\}_{i=1}^n$ exhibiting a sparse group structure. (c) Multi-Level Network Lasso for MTPL (MMTPL): We extend the framework to multi-task settings, applying a row-sparse constraint on $\mathbf{A} = [\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_m]$ to capture task-common features and promoting task group structure among $\{\boldsymbol{\beta}_{t,0}\}_{t=1}^m$. Predictions are computed as $y_{t,i} = \mathbf{x}_{t,i}^T(\boldsymbol{\theta}_{t,0} + \boldsymbol{\theta}_{t,i})$.

To cope with the above two challenges, we propose a novel method, namely **M**ulti-Level Network Lasso for **M**ulti-**T**ask **P**ersonalized **L**earning (**MMTPL**). Specifically, for challenge (1), we propose the multi-level network Lasso to select useful features and encourage sparse group structure among samples at the same time. It decomposes the model parameters $\boldsymbol{\theta}_i$ into a product of a component that selects common features of all samples within one task (i.e., $\boldsymbol{\alpha}$) and a component that captures sample-specific sparsity (i.e., $\boldsymbol{\beta}_i$) and promotes the latter components to group together according to a similarity graph. See Fig. 1(b) for details. For challenge (2), we extend multi-level network Lasso in multi-task scenarios by capturing task-common features and further learning underlying task groups in the feature subspace, such that the task-specific models belonging to the same group share parameters. See Fig. 1(c) for details. To prevent over-penalization on large group outliers, a family of multi-level network Lasso is investigated based on the ℓ_p quasi-norm. To solve the optimization problem, we develop an alternating algorithm. Empirical results on various synthetic and real-world datasets demonstrate its effectiveness. We highlight the contributions as follows:

- We propose the multi-level network Lasso that allows to group samples based a selected subset of features and extend it in multi-task scenarios by learning task groups in the feature subspace.
- We investigate a family of multi-level network Lasso based on the ℓ_p quasi-norm ($0 < p < 1$), which avoids over-penalization on outliers and generalizes the proposed method.
- We develop an alternating algorithm to solve the optimization problem and show its effectiveness on both synthetic and real-world datasets.

2. Related work

Personalized learning (PL) is proposed to learn sample-specific models by utilizing the heterogeneity of samples. Due to the possible large number of samples, the key challenge for PL is how to prevent overfitting. To reduce the model size, the network Lasso [1] learns personalized models that allow for simultaneously clustering and optimizing on graphs of samples, under the assumption that samples in the same group share similar parameters. Localized Lasso [2], a sparse variant of the network Lasso, introduces additional sample-wise exclusive regularizer to promote sparsity in the learned models. DTFLR [20] improves upon network Lasso by introducing an adaptive spanning-tree-based fusion penalty. FORMULA [4] controls the model size by projecting personalized models into a low-dimensional subspace via matrix factorization. By means of additional inputs, [5] proposes a novel distance-matching regularizer for PL based on the assumption that similar models share similar covariates. FALL [6] learns local models in a two-stage manner with the help of pre-computed anchor models. Recently, instance-wise feature selection [21,22] has been proposed to select a specific feature subset for each instance. Existing PL methods generally ignore the sample homogeneity. Although UPFS [3] considers both global and local models, it does so exclusively within an unsupervised context. Furthermore, current PL methods face challenges when adapting to multi-task settings because they primarily focus on sample correlations and neglect the critical task correlations that are pivotal in MTL scenarios.

Multi-task learning (MTL) assumes that the generalization performance for multiple prediction tasks can be enhanced by exploiting task correlation. By assuming that all tasks share a common subset of features, the $\ell_{p,q}$-norm ($p > 1$, $q \geq 1$) based regularizers [23,24] can be imposed to extract shared features among all tasks. Another way to explore task interdependence is to restrict the parameter matrix to be low-rank. This can be achieved by applying matrix factorization [25] or a trace norm based regularizer [12]. To further capture the specificity of tasks, model decomposition approaches are proposed, which decompose the parameter matrix into task-common and task-specific parts by summation [14] or multiplication [15]. Several recent studies focus on capturing group structures among tasks [26,27]. VSTG [28] performs variable selection and learns an overlapping group structure among tasks. GBDSP [17] learns the task group structure based on the block-diagonal task assignment matrix. CCMTL [29] integrates convex clustering into MTL, which is designed to perform parameter learning and task clustering simultaneously. HTEMTL [18] performs subspace clustering on task parameters by exploiting the effect of hidden tasks. EMTSL [30] combines subspace learning with discrete group structure constraint to avoid negative transfer in MTL. These MTL methods, despite their success, focus on sample homogeneity by learning a common model shared across all samples within a task, thus overlooking the critical heterogeneity among individual samples.

By integrating PL and MTL, *multi-task personalized learning (MTPL)* seeks to jointly learn personalized models from multiple tasks. To the best of our knowledge, the only existing MTPL method is proposed in [19]. To reduce the model size and avoid overfitting, it encourages sparse group structures of local models in the latent feature subspace found by low-rank matrix factorization. However, it is unable to learn

a sparse and interpretable model, and thus noisy features might harm its generalization ability. Moreover, it implicitly models task correlation by sharing a common component with all tasks, which is too restrictive in practice. In contrast, our proposed MMTPL utilizes the multi-level network Lasso to capture both homogeneity and heterogeneity during parameter learning and feature selection. This approach not only enhances robustness to noise but also effectively reduces the risk of overfitting in personalized learning. Moreover, it explicitly finds the underlying task groups, enabling a more flexible capture of task correlations.

3. Preliminary

In this work, we use bold uppercase letters for matrices (e.g., $\mathbf{A}$), bold lowercase letters for vectors (e.g., $\mathbf{a}$), normal lowercase letters for scalars (e.g., a). Given m tasks, the tth task is associated with the training data $(\mathbf{X}_t, \mathbf{y}_t)$, $t = 1, 2, \dots, m$, that lies in the d-dimensional feature space. The ith row of the data matrix $\mathbf{X}_t \in \mathbb{R}^{n_t \times d}$ is denoted by $\mathbf{x}_{t,i} \in \mathbb{R}^d$, corresponding to the ith sample in the tth task, and the ith entry of the target vector $\mathbf{y}_t \in \mathbb{R}^{n_t}$ is denoted by $y_{t,i} \in \mathbb{R}$, where n_t is the number of samples in the tth task. There are a total of $N = \sum_t n_t$ samples. For an arbitrary matrix $\mathbf{A}$, $\|\mathbf{A}\|_1$ and $\|\mathbf{A}\|_{2,1}$ denote its ℓ_1-norm and $\ell_{2,1}$-norm, respectively. Specifically, $\|\mathbf{A}\|_1 = \sum_{i,j} |a_{ij}|$ and $\|\mathbf{A}\|_{2,1} = \sum_i \|\mathbf{a}_{i\cdot}\|_2$, where $\|\mathbf{a}_{i\cdot}\|_2$ is the ℓ_2-norm of the ith row of $\mathbf{A}$.

For PL, we employ the following model:

$$\min_{\boldsymbol{\Theta}} \sum_{i=1}^{n} \left(y_i - \mathbf{x}_i^T \boldsymbol{\theta}_i\right)^2 + \Omega(\boldsymbol{\Theta}), \tag{1}$$

where $\boldsymbol{\theta}_i \in \mathbb{R}^d$ represents the regression coefficients for the ith sample, and $\Omega(\boldsymbol{\Theta})$ denotes the regularization term on $\boldsymbol{\Theta} = [\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \dots, \boldsymbol{\theta}_n] \in \mathbb{R}^{d \times n}$. Unlike conventional supervised learning, where $\boldsymbol{\theta}_i = \boldsymbol{\theta}_j$ $(\forall i, j)$, learning personalized models for all samples is computationally expensive, and it is difficult to learn models with limited data. To alleviate this problem, the network Lasso [1] is proposed by borrowing strength across samples:

$$\Omega(\boldsymbol{\Theta}) = \lambda_1 \sum_{i,j=1}^{n} r_{ij} \left\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_j\right\|_2, \tag{2}$$

where λ_1 is the hyper-parameter, and r_{ij} measures the similarity between $\mathbf{x}_i$ and $\mathbf{x}_j$, which is usually computed by the RBF kernel [2,3]. In (2), the ℓ_2-norm penalty, instead of the squared ℓ_2-norm penalty, is imposed on the difference between $\boldsymbol{\theta}_i$ and $\boldsymbol{\theta}_j$, encouraging the difference to be exactly zero, rather than just close to zero. By imposing the network Lasso, similar models tend to be grouped together and share the same parameters, which helps to reduce the model size.

4. Methodology

4.1. Multi-level network Lasso

Although it is important to build personalized model for each sample, different samples more or less have something in common. To capture such sample homogeneity, we propose to decompose the personalized model in (1) into a sum of a global model $\boldsymbol{\theta}_0$ that is shared across samples, and a local model $\boldsymbol{\theta}_i$ $(i = 1, 2, \dots, n)$, which saves sample specificity, i.e.,

$$\min_{\boldsymbol{\Theta}} \sum_{i=1}^{n} \left(y_i - \mathbf{x}_i^T (\boldsymbol{\theta}_0 + \boldsymbol{\theta}_i)\right)^2 + \Omega(\boldsymbol{\Theta}), \tag{3}$$

where $\boldsymbol{\Theta}$ is redefined by $\boldsymbol{\Theta} = [\boldsymbol{\theta}_0, \boldsymbol{\theta}_1, \dots, \boldsymbol{\theta}_n] \in \mathbb{R}^{d \times (n+1)}$. Unlike (1), (3) enables to models sample homogeneity and heterogeneity by $\boldsymbol{\theta}_0$ and $\{\boldsymbol{\theta}_i\}_{i=1}^n$, respectively.

In the network Lasso, sparsity is used only to make the parameters similar, not for feature selection, which fails to eliminate noisy features. To address this problem, the localized Lasso [2] is proposed by incorporating the network Lasso with the exclusive Lasso [31]. However, it imposes sparsity induction only at the sample level and does not capture the across-sample sparsity, i.e., different samples may share a common sparse pattern in addition to their individual ones. To capture both commonality and specificity of sparse pattern, inspired by the multi-level Lasso [15,32], we propose the *multi-level network Lasso*:

$$\begin{aligned} &\Omega(\boldsymbol{\Theta}) = \lambda_1 \sum_{i,j=1}^{n} r_{ij} \left\|\boldsymbol{\beta}_i - \boldsymbol{\beta}_j\right\|_2 + \lambda_2 \sum_{i=0}^{n} \|\boldsymbol{\beta}_i\|_1 + \lambda_3 \|\boldsymbol{\alpha}\|_1 \\ &\text{s.t. } \boldsymbol{\theta}_i = \boldsymbol{\alpha} \circ \boldsymbol{\beta}_i, \ i = 0, 1, \dots, n, \end{aligned} \tag{4}$$

where $\circ$ is the element-wise product operator, $\boldsymbol{\alpha} \in \mathbb{R}^d$ controls the sample-common sparsity, while $\boldsymbol{\beta}_i \in \mathbb{R}^d$ models sample-specific sparsity. In this way, to exclude a feature from a sample, the multiplicative decomposition only requires one of the components to be zero, which brings more flexibility and interpretation, compared with the single-level counterparts, such as Lasso [33] and the exclusive Lasso [31]. By imposing multi-level network Lasso, the learned personalized model in (3) captures both homogeneity and heterogeneity of samples in parameter learning and feature selection, and meanwhile groups similar models together, encouraging the parameter sharing within each group. This approach not only enhances robustness to noise but also reduces the effective parameter size, thereby alleviating the risk of overfitting.

4.2. Extending to the multi-task setting

The multi-level network Lasso enables to promote sample sparse group structure, but it is unable to directly model the correlation among tasks. In this subsection, we extend it to handle the MTL problems by detecting the underlying task groups in a similar way.

A straightforward way is to directly incorporate (3) and (4) in the MTL setting, that yields:

$$\begin{aligned} &\min_{\boldsymbol{\Theta}} \sum_{t=1}^{m} \sum_{i=1}^{n_t} \left(y_{t,i} - \mathbf{x}_{t,i}^T (\boldsymbol{\theta}_{t,0} + \boldsymbol{\theta}_{t,i})\right)^2 + \Omega(\boldsymbol{\Theta}) \\ &\text{s.t. } \boldsymbol{\theta}_{t,i} = \boldsymbol{\alpha}_t \circ \boldsymbol{\beta}_{t,i}, \ t = 1, \dots, m, \ i = 0, 1, \dots, n_t, \end{aligned} \tag{5}$$

where $\boldsymbol{\Theta}$ is defined by $\boldsymbol{\Theta} = [\boldsymbol{\Theta}_1, \boldsymbol{\Theta}_2, \dots, \boldsymbol{\Theta}_m]$, with $\boldsymbol{\Theta}_t = [\boldsymbol{\theta}_{t,0}, \boldsymbol{\theta}_{t,1}, \dots, \boldsymbol{\theta}_{t,n_t}]$, and $\Omega(\boldsymbol{\Theta}) = \sum_{t=1}^{m} \Omega(\boldsymbol{\Theta}_t)$, i.e.,

$$\Omega(\boldsymbol{\Theta}) = \lambda_1 \sum_{t=1}^{m} \left(\sum_{i,j=1}^{n_t} r_{ij}^t \left\|\boldsymbol{\beta}_{t,i} - \boldsymbol{\beta}_{t,j}\right\|_2 + \lambda_2 \sum_{i=0}^{n_t} \|\boldsymbol{\beta}_{t,i}\|_1 + \lambda_3 \|\boldsymbol{\alpha}_t\|_1 \right). \tag{6}$$

In MTL, modeling correlation among tasks is crucial to improve the generalization performance [9]. Based on the assumption that multiple tasks are correlated via task-common features, the $\ell_{2,1}$-norm penalty $\|\mathbf{A}\|_{2,1}$ is imposed on $\mathbf{A} = [\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \dots, \boldsymbol{\alpha}_m]$ to promote feature-wise group sparsity. Moreover, multiple tasks may exhibit group structure in the feature subspace, where task models belonging to the same group are related to each other. Similar with the sample grouping manner in (4), we integrate $\sum_{t,s} \|\boldsymbol{\beta}_{t,0} - \boldsymbol{\beta}_{s,0}\|_2$ into (6), by assuming that all tasks have the opportunity to become similar to each other. Therefore, the regularizer $\Omega(\boldsymbol{\Theta})$ is reformulated into:

$$\begin{aligned} \Omega(\boldsymbol{\Theta}) = \lambda_1 &\left(\sum_{t=1}^{m} \sum_{i,j=1}^{n_t} r_{ij}^t \left\|\boldsymbol{\beta}_{t,i} - \boldsymbol{\beta}_{t,j}\right\|_2 + \sum_{t,s=1}^{m} \|\boldsymbol{\beta}_{t,0} - \boldsymbol{\beta}_{s,0}\|_2 \right) \\ &+ \lambda_2 \sum_{t=1}^{m} \sum_{i=0}^{n_t} \|\boldsymbol{\beta}_{t,i}\|_1 + \lambda_3 \|\mathbf{A}\|_{2,1}. \end{aligned} \tag{7}$$

In (7), the global model $\boldsymbol{\theta}_{t,0} = \boldsymbol{\alpha}_t \circ \boldsymbol{\beta}_{t,0}$ is used to represent the model of the tth task, and $\sum_{t,s} \|\boldsymbol{\beta}_{t,0} - \boldsymbol{\beta}_{s,0}\|_2$ encourages task models that are close at ℓ_2 distance to become similar. Thanks to the multiplicative decomposition $\boldsymbol{\theta}_{t,0} = \boldsymbol{\alpha}_t \circ \boldsymbol{\beta}_{t,0}$, we have the chance to model negative correlation[1] [26] by penalizing the difference between $\boldsymbol{\beta}_{t,0}$ and $\boldsymbol{\beta}_{s,0}$, instead of $\boldsymbol{\theta}_{t,0}$ and $\boldsymbol{\theta}_{s,0}$.

[1] Grouping on the basis of ℓ_2 distance can put two tasks with parameters $\boldsymbol{\theta}$ and $-\boldsymbol{\theta}$ in separate groups while there is clearly a correlation between them.

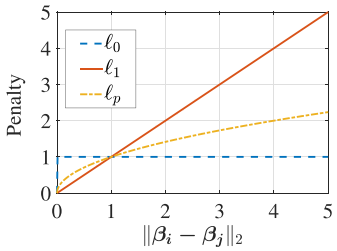


Fig. 2. Comparison of ℓ_0, ℓ_1 and $\ell_p\,(p = 1/2)$ by increasing the magnitude of the difference between $\boldsymbol{\beta}_i$ and $\boldsymbol{\beta}_j$.

By combining (5) and (7), the objective of the proposed **MMTPL** method can be defined in a compact form:

$$\min_{\boldsymbol{\Theta}} \left\|\mathbf{y} - \widetilde{\mathbf{X}}\boldsymbol{\theta}\right\|_2^2 + \lambda_1 \sum_{i,j} r_{ij} \left\|\boldsymbol{\beta}_i - \boldsymbol{\beta}_j\right\|_2 + \lambda_2 \|\mathbf{B}\|_1 + \lambda_3 \|\mathbf{A}\|_{2,1}$$
$$\text{s.t. } \boldsymbol{\Theta} = (\mathbf{AM})\circ\mathbf{B}, \tag{8}$$

where $\widetilde{\mathbf{X}} \in \mathbb{R}^{N\times d(N+m)}$ is a block diagonal matrix with the tth block being $\widetilde{\mathbf{X}}_t = [\mathbf{X}_t, \mathrm{Diag}(\mathbf{X}_t)]$, $\mathrm{Diag}(\mathbf{X}_t)$ represents a diagonal matrix with the ith diagonal element being $\mathbf{x}_{t,i}^T$, $\mathbf{y} = [\mathbf{y}_1; \mathbf{y}_2; \ldots; \mathbf{y}_m] \in \mathbb{R}^N$ is the response vector of all samples, and $\boldsymbol{\theta} = \mathrm{vec}(\boldsymbol{\Theta})$ represents the vectorization of $\boldsymbol{\Theta}$. Besides, $\mathbf{B} = [\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_m] \in \mathbb{R}^{d\times(N+m)}$ with $\mathbf{B}_t = [\boldsymbol{\beta}_{t,0}, \boldsymbol{\beta}_{t,1}, \ldots, \boldsymbol{\beta}_{t,n_t}]$, and $\mathbf{M} = [\mathbf{e}_1\mathbf{1}_{n_1+1}^T, \mathbf{e}_2\mathbf{1}_{n_2+1}^T, \ldots, \mathbf{e}_m\mathbf{1}_{n_m+1}^T]$ is an auxiliary matrix with $\mathbf{e}_t$ denoting a unit vector with the non-zero element at the tth entry. In the second term of (8), $\boldsymbol{\beta}_i$ denotes the ith column of $\mathbf{B}$, r_{ij} is the entry in the ith row and jth column of $\mathbf{R} \in \mathbb{R}^{(N+m)\times(N+m)}$, which is a block matrix with $m \times m$ blocks $\widetilde{\mathbf{R}}_{ts} = \begin{bmatrix} 1 & \mathbf{0}_{n_s}^T \\ \mathbf{0}_{n_t} & \mathbf{R}_{ts} \end{bmatrix}$, $t, s = 1, 2, \ldots, m$, and $\mathbf{R}_{ts} = \mathbf{R}_t$ ($(\mathbf{R}_t)_{ij} = r_{ij}^t$) if $t = s$ and $\mathbf{R}_{ts} = \mathbf{0}$ otherwise. The proposed MMTPL method in (8) successfully builds personalized models in the MTL setting, which considers both homogeneity and heterogeneity of samples during parameter learning and feature selection, promotes structured sparsity among samples, and exploits task correlations by capturing latent task group structure in the feature subspace.

4.3. Prediction

Once the models $\boldsymbol{\theta}_{t,0}$ and $\{\boldsymbol{\theta}_{t,i}\}_{i=1}^{n_t}$ of the tth task are learned based on (8), MMTPL makes the prediction of an unseen testing sample $\hat{\mathbf{x}}_{t,i}$ by $\hat{y}_{t,i} = \hat{\mathbf{x}}_{t,i}^T(\boldsymbol{\theta}_{t,0} + \hat{\boldsymbol{\theta}}_{t,i})$, where the ith unknown personalized model $\hat{\boldsymbol{\theta}}_{t,i}$ is obtained by solving the following Weber problem [34]:

$$\hat{\boldsymbol{\theta}}_{t,i} = \arg\min_{\boldsymbol{\theta}} \sum_{j\in k\mathrm{NN}(\hat{\mathbf{x}}_{t,i})} r_{i,j} \|\boldsymbol{\theta} - \boldsymbol{\theta}_{t,j}\|_2. \tag{9}$$

Here kNN denotes the k nearest neighbors of $\hat{\mathbf{x}}_{t,i}$ in training data from the tth task ($k = 5$ in our experiments), and $r_{i,j}$ measures the similarity between $\hat{\mathbf{x}}_{t,i}$ and its neighbor $\mathbf{x}_{t,j}$, which is calculated by the RBF kernel [2,3]. The Weber problem can be solved by an iterative algorithm [34]. Specifically, at the lth step of the iterative algorithm, the model is moved closer to the optimal solution by setting $\boldsymbol{\theta}^{(l+1)}$ to be the solution of a weighted least squares problem:

$$\min_{\boldsymbol{\theta}} \sum_{j\in k\mathrm{NN}(\hat{\mathbf{x}}_{t,j})} \frac{r_{i,j}}{\|\boldsymbol{\theta}^{(l)} - \boldsymbol{\theta}_{t,j}\|_2} \|\boldsymbol{\theta} - \boldsymbol{\theta}_{t,j}\|_2^2. \tag{10}$$

As the unique optimal solution to the above weighted least square problem, each successive is calculated by:

$$\boldsymbol{\theta}^{(l+1)} = \left(\sum_{j\in k\mathrm{NN}(\hat{\mathbf{x}}_{t,i})} \frac{r_{i,j}\boldsymbol{\theta}_{t,j}}{\|\boldsymbol{\theta}^{(l)} - \boldsymbol{\theta}_{t,j}\|_2}\right) \Big/ \left(\sum_{j\in k\mathrm{NN}(\hat{\mathbf{x}}_{t,i})} \frac{r_{i,j}}{\|\boldsymbol{\theta}^{(l)} - \boldsymbol{\theta}_{t,j}\|_2}\right). \tag{11}$$

4.4. Remarks on multi-level network Lasso

The use of the multi-level network Lasso is central to MMTPL, which captures both homogeneity and heterogeneity of samples as well as promotes sparse group structure among personalized models. Such group structure is essentially promoted by the ℓ_1-norm penalty on the difference $\|\boldsymbol{\beta}_i - \boldsymbol{\beta}_j\|_2$ between arbitrary pairwise models, which is weighted by the similarity r_{ij} of a neighborhood graph. But the ℓ_1-norm regularization typically results in over-penalization on large group outliers, as shown in Fig. 2. In fact, we do not care about the magnitude of the difference once the difference is non-zero, because in this case, the models $\boldsymbol{\beta}_i$ and $\boldsymbol{\beta}_j$ actually belong to different groups. Thus, a more natural penalty should be the ℓ_0-norm. However, using the ℓ_0-norm penalty results in an intractable combinatorial optimization problem, which is NP-hard.

Motivated by the superiority of dealing with sparsity inducing problems via the ℓ_p quasi-norm ($0 < p < 1$) [35], we generalize the proposed multi-level network Lasso, resulting in a family framework of MMTPL$_p$:

$$\min_{\boldsymbol{\Theta}} \left\|\mathbf{y} - \widetilde{\mathbf{X}}\boldsymbol{\theta}\right\|_2^2 + \lambda_1 \left\{\sum_{i,j} \left(r_{ij} \left\|\boldsymbol{\beta}_i - \boldsymbol{\beta}_j\right\|_2\right)^p\right\}^{\frac{1}{p}} + \lambda_2 \|\mathbf{B}\|_1 + \lambda_3 \|\mathbf{A}\|_{2,1}$$
$$\text{s.t. } \boldsymbol{\Theta} = (\mathbf{AM})\circ\mathbf{B}, \quad p \in (0, 1). \tag{12}$$

Since the ℓ_p quasi-norm ($0 < p < 1$) is closer to the ℓ_0-norm than the ℓ_1-norm, as illustrated in Fig. 2, it helps prevent over-penalization over large group outliers, and simultaneously yields a sparser solution where relatively few group differences are non-zero (as it penalizes more aggressively on small values). We compare the vanilla MMTPL and MMTPL$_p$ on real-world data in experiment section.

5. Optimization

We consider the optimization of MMTPL$_p$ ($0 < p < 1$) in (12), and the vanilla MMTPL problem in (8) can be simply solved by setting $p = 1$. Since the optimization problem is not jointly convex, we develop an alternating algorithm to solve it. The algorithm repeats the following steps until convergence.

5.1. Updating A

With fixed $\mathbf{B}$, the problem w.r.t. $\mathbf{A}$ becomes:

$$\min_{\mathbf{A}} \left\|\mathbf{y} - \widetilde{\mathbf{X}}\mathrm{vec}\left((\mathbf{AM})\circ\mathbf{B}\right)\right\|_2^2 + \lambda_3 \|\mathbf{A}\|_{2,1}. \tag{13}$$

The above problem is non-smooth, and proximal gradient descent method [36] can be applied to update $\mathbf{A}$. The gradient of the smooth part of the objective function in (13) w.r.t. $\mathrm{vec}(\mathbf{A})$ is given by:

$$\nabla f(\mathrm{vec}(\mathbf{A})) = \left(\mathbf{M}\otimes\mathbf{I}_d\right)\left(-2\widetilde{\mathbf{X}}^T(\mathbf{y} - \widetilde{\mathbf{X}}\mathrm{vec}(\boldsymbol{\Theta}))\circ\mathrm{vec}(\mathbf{B})\right), \tag{14}$$

where $\mathbf{I}_d \in \mathbb{R}^{d\times d}$ is the identity matrix, and $\otimes$ is the Kronecker product. Then the update rule w.r.t. $\mathbf{A}$ is:

$$\left(\mathbf{A}^*\right)_i = \mathrm{prox}_{\eta\lambda_3\|\cdot\|_2}\left((\mathbf{A})_i - \eta\left(\nabla f\left(\mathbf{A}\right)\right)_i\right), \ \forall i \in \mathbb{N}_d, \tag{15}$$

where $\mathbb{N}_d = \{1, 2, \ldots, d\}$, $(\mathbf{A})_i$ denotes the ith row of $\mathbf{A}$, $\nabla f(\mathbf{A})$ is obtained by reshaping $\nabla f(\mathrm{vec}(\mathbf{A}))$, η is the learning rate, and $\mathrm{prox}_{\gamma\|\cdot\|_2}(\mathbf{x}) = \left(1 - \frac{\gamma}{\max\{\|\mathbf{x}\|_2, \gamma\}}\right)\mathbf{x}$ is the proximal operator of the ℓ_2-norm.

5.2. Updating B

With fixed $\mathbf{A}$, the problem w.r.t. $\mathbf{B}$ becomes:

$$\min_{\mathbf{B}} \left\|\mathbf{y} - \widetilde{\mathbf{X}}\mathrm{vec}\left((\mathbf{AM})\circ\mathbf{B}\right)\right\|_2^2 + \lambda_1 \left\{\sum_{i,j} \left(r_{ij} \left\|\boldsymbol{\beta}_i - \boldsymbol{\beta}_j\right\|_2\right)^p\right\}^{\frac{1}{p}} + \lambda_2 \|\mathbf{B}\|_1. \tag{16}$$

The formulation in $\mathbf{B}$ is non-differentiable and non-convex when $0 < p < 1$. Here we solve an equivalent problem according to the following proposition.

Proposition 1. *Let $p \in (0,2)$ and $q = \frac{p}{2-p}$. For any vectors $\boldsymbol{\beta}_i, \boldsymbol{\beta}_j \in \mathbb{R}^d$, the optimization problem in (16) has an equivalent formulation w.r.t.* $\mathbf{B}$, *that is:*

$$\min_{\mathbf{B},\mathbf{Z}} \left\|\mathbf{y} - \widetilde{\mathbf{X}} vec\left((\mathbf{AM})\circ\mathbf{B}\right)\right\|_2^2 + \lambda_1\left(\sum_{i,j}\frac{r_{ij}^2\left\|\boldsymbol{\beta}_i-\boldsymbol{\beta}_j\right\|_2^2}{2z_{ij}} + \frac{1}{2}\|\mathbf{Z}\|_q\right) + \lambda_2\|\mathbf{B}\|_1, \tag{17}$$

where $(\mathbf{Z})_{ij} = z_{ij} \in \mathbb{R}_+$ is the introduced auxiliary variable, and the minimum of $\mathbf{B}$ *is uniquely attained when*

$$z_{ij} = \left(r_{ij}\left\|\boldsymbol{\beta}_i-\boldsymbol{\beta}_j\right\|_2\right)^{2-p}\left\{\sum_{i,j}\left(r_{ij}\left\|\boldsymbol{\beta}_i-\boldsymbol{\beta}_j\right\|_2\right)^p\right\}^{\frac{p-1}{p}}. \tag{18}$$

Proof. To verify Proposition 1, we need to prove the correctness of the following Lemma first.

Lemma 1 (*Variational Formulation*). *Let $p \in (0,2)$ and $q = \frac{p}{2-p}$. For any vector $\mathbf{x} \in \mathbb{R}^d$, we have the following equivalent:*

$$\|\mathbf{x}\|_p = \min_{\mathbf{z}\in\mathbb{R}_+^d}\left[\frac{1}{2}\sum_{i=1}^d\frac{x_i^2}{z_i} + \frac{1}{2}\|\mathbf{z}\|_q\right],$$

and the minimum is uniquely attained for $z_i = |x_i|^{2-p}\|\mathbf{x}\|_p^{p-1}$, $\forall i \in \{1,2,\ldots,d\}$.

Proof. Let $\phi : \mathbf{z} \to \sum_{i=1}^d x_i^2 z_i^{-1} + \|\mathbf{z}\|_q$ be the continuously differentiable function defined on $(0,+\infty)$. We have $\lim_{\|\mathbf{z}\|_q\to+\infty}\phi(\mathbf{z}) = +\infty$ and $\lim_{z_i\to 0}\phi(\mathbf{z}) = +\infty$ if $x_i \neq 0$ (for $x_i = 0$, note that $\min_{\mathbf{z}\in\mathbb{R}_+^d}\phi(\mathbf{z}) = \min_{\mathbf{z}\in\mathbb{R}_+^d, z_i=0}\phi(\mathbf{z})$). Thus, the infimum exists and it is attained. Taking the derivative w.r.t. z_i (for $z_i > 0$) leads to the expression of the unique minimum, and the expression is still correct for $z_i = 0$.

According to Lemma 1 and the fact

$$\left\{\sum_{i,j}\left(r_{ij}\left\|\boldsymbol{\beta}_i-\boldsymbol{\beta}_j\right\|_2\right)^p\right\}^{\frac{1}{p}} = \left\|\left(r_{ij}\left\|\boldsymbol{\beta}_i-\boldsymbol{\beta}_j\right\|_2\right)_{(i,j)\in[\![1,N+m]\!]\times[\![1,N+m]\!]}\right\|_p,$$

where $[\![1,N+m]\!] = \{1,2,\ldots,N+m\}$, the tuple composed of different $r_{ij}\left\|\boldsymbol{\beta}_i-\boldsymbol{\beta}_j\right\|_2$ is denoted by $\left(r_{ij}\left\|\boldsymbol{\beta}_i-\boldsymbol{\beta}_j\right\|_2\right)_{(i,j)\in[\![1,N+m]\!]\times[\![1,N+m]\!]} \in \mathbb{R}^{(N+m)^2\times 1}$. We have the following equality:

$$\left\{\sum_{i,j}\left(r_{ij}\left\|\boldsymbol{\beta}_i-\boldsymbol{\beta}_j\right\|_2\right)^p\right\}^{\frac{1}{p}} = \min_{\mathbf{Z}\in\mathbb{R}_+^{(N+m)\times(N+m)}}\left[\sum_{i,j}\frac{r_{ij}^2\left\|\boldsymbol{\beta}_i-\boldsymbol{\beta}_j\right\|_2^2}{2z_{ij}} + \frac{1}{2}\|\mathbf{Z}\|_q\right].$$

Then the optimization problem Eq. (16) has an equivalent w.r.t. $\mathbf{B}$, that is:

$$\min_{\mathbf{B},\mathbf{Z}}\left\|\mathbf{y}-\widetilde{\mathbf{X}}\mathrm{vec}\left((\mathbf{AM})\circ\mathbf{B}\right)\right\|_2^2 + \lambda_1\left(\sum_{i,j}\frac{r_{ij}^2\left\|\boldsymbol{\beta}_i-\boldsymbol{\beta}_j\right\|_2^2}{2z_{ij}} + \frac{1}{2}\|\mathbf{Z}\|_q\right) + \lambda_2\|\mathbf{B}\|_1,$$

and the minimum of $\mathbf{B}$ is uniquely attained when

$$z_{ij} = \left(r_{ij}\left\|\boldsymbol{\beta}_i-\boldsymbol{\beta}_j\right\|_2\right)^{2-p}\left\{\sum_{i,j}\left(r_{ij}\left\|\boldsymbol{\beta}_i-\boldsymbol{\beta}_j\right\|_2\right)^p\right\}^{\frac{p-1}{p}}. \quad \square$$

Given the closed-form solution of z_{ij}, we reformulate the problem (17) into a matrix form as follows:

$$\min_{\mathbf{B}}\left\|\mathbf{y}-\widetilde{\mathbf{X}}\mathrm{vec}\left((\mathbf{AM})\circ\mathbf{B}\right)\right\|_2^2 + 2\lambda_1\,\mathrm{Tr}\left(\mathbf{B}\left(\mathbf{D}-\mathbf{W}\right)\mathbf{B}^T\right) + \lambda_2\|\mathbf{B}\|_1, \tag{19}$$

where $\mathbf{W}$ is a symmetric matrix with $w_{ij} = r_{ij}^2/2z_{ij}$ $(i,j = 1,\ldots,N+m)$, $\mathbf{D}$ is a diagonal matrix with the ith diagonal element $d_{ii} = \sum_{j=1}^{N+m} w_{ij}$ $(i = 1,\ldots,N+m)$, and $\mathrm{Tr}(\cdot)$ is the trace of a matrix. Then, proximal gradient descent method is applied to update $\mathbf{B}$. The gradient of the smooth part of the objective in (19) w.r.t. $\mathrm{vec}(\mathbf{B})$ is:

$$\nabla f(\mathrm{vec}(\mathbf{B})) = \left(-2\widetilde{\mathbf{X}}^T\left(\mathbf{y}-\widetilde{\mathbf{X}}\mathrm{vec}(\boldsymbol{\Theta})\right)\right)\circ\left(\left(\mathbf{M}^T\otimes\mathbf{I}_d\right)\mathrm{vec}(\mathbf{A})\right) + 4\lambda_1\left(\left(\mathbf{D}-\mathbf{W}\right)^T\otimes\mathbf{I}_d\right)\mathrm{vec}(\mathbf{B}). \tag{20}$$

Then the update rule w.r.t. $\mathbf{B}$ is defined as:

$$\mathbf{B}^* = S_{\mu\lambda_2}\left(\mathbf{B} - \mu\nabla f(\mathbf{B})\right), \tag{21}$$

where $\nabla f(\mathbf{B})$ is obtained by reshaping $\nabla f(\mathrm{vec}(\mathbf{B}))$, μ is the learning rate, and $S_\tau(x) = \mathrm{sign}(x)\max(|x|-\tau,0)$ is the element-wise soft thresholding operator.

Algorithm 1 MMTPL$_p$: Optimization procedure

Input: $\widetilde{\mathbf{X}}$, $\mathbf{y}$, λ_1, λ_2, λ_3.
Output: $\boldsymbol{\Theta} = (\mathbf{AM})\circ\mathbf{B}$.
1: Initialize $\mathbf{A}$, $\mathbf{B}$.
2: Compute $\mathbf{R}$ by RBF kernel using $\widetilde{\mathbf{X}}$.
3: **repeat**
4: Update $\mathbf{A}$ via APG based on Eq. (15).
5: Update z_{ij} based on Eq. (18).
6: Update $\mathbf{B}$ via APG based on Eq. (21).
7: **until** *Convergence*

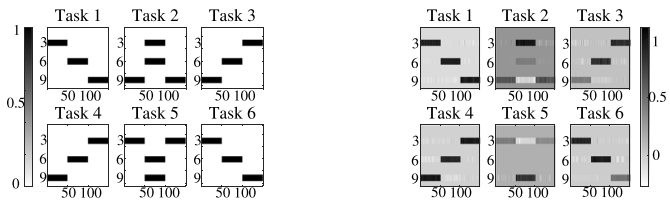
5.3. Analysis on time complexity

In practice, we apply accelerated proximal method (APG) [36] to accelerate the optimization algorithm. In term of time complexity analysis, updating $\mathbf{A}$ comprises two major steps, gradient calculation and soft thresholding operation, with time complexities of $\mathcal{O}(dN(N+m))$ and $\mathcal{O}(dm)$, respectively. Similarly, on updating $\mathbf{B}$, we perform gradient descent and soft thresholding operations again, resulting in time complexities of $\mathcal{O}(d(N+m)^2)$ and $\mathcal{O}(d(N+m))$, respectively. Therefore, the total time complexity of each iteration is $\mathcal{O}(d(N+m)^2)$. Although the time cost is somehow high, we find that the complexity is linear w.r.t. the number of features, and the algorithm usually converges within 50 iterations in real-world experiments. We provide the pseudo code of the alternating optimization procedure in Algorithm 1. In practice, the similarity matrix $\mathbf{R}$ is computed in the same way with [2,3], by using the RBF kernel. It consumes constant time and thus is not considered in the time complexity analysis. For further exploration and replication, our code is available at: https://www.dropbox.com/scl/fi/nx0ry03m5l30xxs52fjmz/MMTPL_code.zip?rlkey=jeydcjf44kc6d5mqz7ib5wsee&st=zyc2o683&dl=0.

6. Experiment

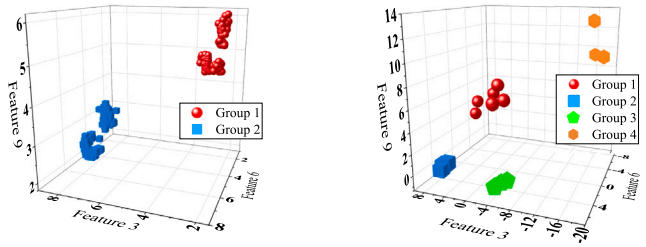
6.1. Experimental setting

Synthetic dataset. We set the number of tasks as $m = 6$ and each task has $n = 150$ labeled samples. The dimensionality of samples is set as $d = 10$ for all tasks. The ground truth personalized model for each sample is represented in a multiplicative way, i.e., $\boldsymbol{\theta}_{t,i} = \boldsymbol{\alpha}_t\circ(\boldsymbol{\beta}_{t,0} + \boldsymbol{\beta}_{t,i})$, $t = 1,\ldots,m$, $i = 1,\ldots,n_t$. Elements of $\boldsymbol{\alpha}_t \in \mathbb{R}^d$ are randomly sampled from normal distribution $\mathcal{N}(0,1)$. To make the matrix $\mathbf{A} = [\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\ldots,\boldsymbol{\alpha}_m]$ row-wise sparse, we further set all rows of $\mathbf{A}$ except the 3rd, 6th, and 9th rows to be 0. Elements of $\boldsymbol{\beta}_{t,0} \in \mathbb{R}^d$ are sampled from uniform distribution $\mathcal{U}(0,10)$. Moreover, $\boldsymbol{\beta}_{1,0}$, $\boldsymbol{\beta}_{2,0}$ and $\boldsymbol{\beta}_{3,0}$ share the identical values, and the same goes for $\boldsymbol{\beta}_{4,0}$, $\boldsymbol{\beta}_{5,0}$ and $\boldsymbol{\beta}_{6,0}$, which indicates that there are two task groups. The local sparse model $\boldsymbol{\beta}_{t,i} \in \mathbb{R}^d$ is specially defined, where every 50 samples in a task lie in the same cluster and thus share the same sparse pattern, as illustrated in Fig. 3(a). The similarity matrix $\mathbf{R}_t \in \mathbb{R}^{n_t\times n_t}$ is designed as a block diagonal matrix with the main-diagonal blocks being all-one matrices. The data $\mathbf{x}_{t,i} \in \mathbb{R}^d$ is randomly sampled from normal distribution $\mathcal{N}(0,25)$. Finally, the target is calculated by $y_{t,i} = \mathbf{x}_{t,i}^T\boldsymbol{\theta}_{t,i} + \delta_{t,i}$, where $\delta_{t,i}$ is zero-mean Gaussian noise sampled from $\mathcal{N}(0,1)$.



(a) Designed pattern $\beta^*_{t,i}$ (b) Learned pattern $\beta_{t,i}$

Fig. 3. Illustration of group sparse structure recovered by MMTPL on the synthetic dataset. (a): designed sparse model $\beta^*_{t,i}$; (b): learned sparse model $\beta_{t,i}$.



(a) Learned by MMTPL (b) Learned by MMTPL$_{\alpha=1}$

Fig. 4. Comparison of MMTPL and MMTPL$_{\alpha=1}$ on modeling task group structure. For clarity, we show $\beta_{t,0} + \beta_{t,i}$ with non-zero features, i.e., the 3rd, 6th, 9th features.

Table 1
The statistics of used six real-world datasets.

Datasets	#Samples	#Features	#Tasks
SARCOS	48 933	21	7
Parkinsons	5875	16	42
Computer	20	13	190
Housing	985	6	5
WQ	1060	14	16
RF1	9125	64	8

Real-world datasets. We select SARCOS,[2] Parkinsons,[3] Computer,[4] Housing,[5] WQ[6] and RF1[6] for performance evaluation. The first three are benchmark multi-task datasets, that have been widely used in previous MTL works [9,15]. The last two are multi-target regression datasets, and we treat each target as a learning task. The SARCOS dataset relates to an inverse dynamics problem. The Parkinsons dataset is to predict the disease symptom scores of Parkinson. The Computer dataset aims to predict students' purchase intention on computers. The housing dataset refers to house price forecasts in the Greater Sacramento area for May 2008. The WQ dataset refers to the problem of inferring chemical parameters of water quality. The RF1 dataset concerns the prediction of river network flows for 48h in the future. Their statistics are summarized in Table 1.

Comparing methods. We compare MMTPL with three types of learning methods, i.e., PL, MTL and MTPL. For PL, FORMULA, the Network Lasso and the Localized Lasso are selected. FORMULA [4] treats the prediction of each sample as a task and utilizes a multi-task type approach to solve the PL problems. The Network Lasso [1] estimates personalized models by clustering and optimizing the parameters in graph, and the Localized Lasso [2] is a sparse variant of the Network Lasso. For MTL, we compare MMTPL with MMTFL, VSTG, GBDSP, HTEMTL and AutoTR. The MMTFL [15] method decomposes model parameters into a multiplication of two components: the across-task feature indicator and task-specific part, in a similar way with multi-level Lasso [32]. The VSTG [28], GBDSP [17] and HTEMTL [18] method are proposed based on the assumption that tasks may have a latent group structure. The AutoTR [37] automatically captures complex correlations among tasks, assuming each task model can represent every other task. For MTPL, the only existing method proposed by [19] is used. In addition, Lasso [33] is selected as a baseline for sparse learning.

Parameter setting. For grid search, the number of latent bases in FORMULA, MTPL, VSTG and GBDSP is selected from $\{3, 5, 7, 9, 11\}$. The number of transfer groups in GBDSP is selected from $\{3, 5, 7, 9, 11\}$. The value k of k-support norm in VSTG is selected from $\{1, 2, 3\}$. As for other hyper-parameters, the search grid is set to be $\{2^{-10}, 2^{-8}, \ldots, 2^{8}, 2^{10}\}$. The maximum number of iterations is 1000, and the algorithm will terminate once the relative change of the objective is below 10^{-3}.

Evaluation. In experiments, we randomly select 70%, 20% and 10% of total samples as the training set, the validation set and the testing set, respectively. We repeat this process ten times, and report the mean value with standard deviation in terms of three metrics, root mean squared error (RMSE), mean absolute error (MAE) and normalized mean squared error (NMSE).

6.2. Experiments on synthetic data

Illustration of group sparse structure. The recovery of group sparse structure on the synthetic dataset is illustrated in Fig. 3, where $\beta^*_{t,i}$ denotes the designed sparse model at sample level, and $\beta_{t,i}$ is the sparse model learned by MMTPL with the setting $\lambda_1 = 2^{-4}$, $\lambda_2 = 2^{-2}$ and $\lambda_3 = 2^6$. As shown is Fig. 3, MMTPL successfully recovers the sparse group structure among the sparse models $\beta_{t,i}$ $(\forall t, i)$. This demonstrates the effectiveness of multi-level network Lasso used in MMTPL, that enables to jointly perform sample grouping and feature selection.

Case study on modeling negative correlation. The MMTPL has the ability to capture negative correlation among task models due to the multiplicative decomposition $\theta_{t,i} = \alpha_t \circ \beta_{t,i}$. We compare the proposed MMTPL method with a special variant MMTPL$_{\alpha=1}$ ($\theta_{t,i} = \mathbf{1} \circ \beta_{t,i}$), which fixes α_t to be all-one vector during the learning process. Fig. 4 illustrates $\beta_{t,0} + \beta_{t,i}$ learned by MMTPL and MMTPL$_{\alpha=1}$, in the same parameter setting with Fig. 3. As we can see in Fig. 4, MMTPL successfully detects the two underlying task groups as we designed. In contrast, MMTPL$_{\alpha=1}$ fails to find the correct task groups, due to the constraint $\alpha = \mathbf{1}$, which makes it impossible to group two negatively correlated tasks together. The ability on capturing negative correlations makes MMTPL flexible enough to efficiently save task correlations.

[2] http://www.gaussianprocess.org/gpml/data.
[3] https://archive.ics.uci.edu/ml/datasets.php.
[4] https://github.com/probml/pmtk3/tree/master/data.
[5] http://support.spatialkey.com/spatialkey-sample-csv-data.
[6] http://mulan.sourceforge.net/datasets-mtr.html.

Table 2
Experimental results on six real-world datasets. The best results of each dataset are highlighted in boldface.

Dataset	Metric	LASSO	FORMULA	Network Lasso	Localized Lasso	MMTFL	VSTG	GBDSP	HTEMTL	AutoTR	MTPL	MMTPL
Computer	RMSE	1.7292(0.0032)	2.2385(0.0033)	2.3389(0.0082)	1.7951(0.0054)	1.6211(0.0015)	1.5895(0.0052)	1.6335(0.0034)	1.6151(0.0060)	1.6919(0.0052)	1.6265(0.0066)	**1.5179(0.0038)**
	NMSE	2.2177(0.0863)	3.4466(0.6021)	4.5117(0.2946)	2.2729(0.1055)	2.0084(0.1714)	2.0477(0.0847)	2.0288(0.1209)	1.9169(0.0683)	2.2181(0.1118)	1.9568(0.1352)	**1.7951(0.0686)**
	MAE	1.5386(0.0027)	1.9506(0.0038)	2.1011(0.0073)	1.6000(0.0034)	1.4471(0.0019)	1.4161(0.0037)	1.4697(0.0051)	1.4503(0.0050)	1.5100(0.0040)	1.4472(0.0046)	**1.3569(0.0029)**
Parkinsons	RMSE	2.0841(0.0168)	2.2495(0.2307)	1.9935(0.0423)	1.9191(0.0057)	1.9154(0.0039)	1.9053(0.0104)	1.9149(0.0205)	1.8929(0.0060)	1.9253(0.0143)	1.8960(0.0092)	**1.8671(0.0061)**
	NMSE	1.4448(0.1141)	1.4404(0.0625)	1.1471(0.1261)	0.9872(0.0093)	1.0100(0.0147)	0.9765(0.0151)	1.0301(0.0281)	1.0009(0.0254)	1.0660(0.0418)	**0.9605(0.0162)**	0.9712(0.0118)
	MAE	1.7151(0.0095)	1.8702(0.2333)	1.5886(0.0077)	1.5670(0.0044)	1.6277(0.0027)	1.5940(0.0069)	1.5740(0.0033)	1.5825(0.0040)	1.5916(0.0055)	1.5577(0.0036)	**1.5495(0.0066)**
Housing	RMSE	0.1700(0.0013)	0.1630(0.0011)	0.1583(0.0011)	0.1577(0.0013)	0.1677(0.0014)	0.1684(0.0014)	0.1766(0.0011)	0.1702(0.0013)	0.1726(0.0013)	0.1597(0.0011)	**0.1561(0.0011)**
	NMSE	0.7110(0.0484)	0.6873(0.0540)	0.6010(0.0317)	0.5966(0.0330)	0.6674(0.0272)	0.6977(0.0442)	0.7641(0.0425)	0.6988(0.0371)	0.7224(0.0331)	0.6253(0.0330)	**0.5824(0.0181)**
	MAE	0.1361(0.0006)	0.1271(0.0005)	0.1246(0.0005)	**0.1228(0.0005)**	0.1358(0.0007)	0.1365(0.0007)	0.1383(0.0006)	0.1361(0.0006)	0.1449(0.0014)	0.1291(0.0005)	0.1250(0.0007)
WQ	RMSE	0.8561(0.0053)	0.8551(0.0022)	0.8178(0.0019)	0.8201(0.0017)	0.8329(0.0015)	0.8215(0.0015)	0.8213(0.0015)	0.8210(0.0015)	0.8232(0.0015)	0.8134(0.0016)	**0.8118(0.0017)**
	NMSE	0.8848(0.0032)	0.9322(0.0051)	0.8305(0.0002)	0.8364(0.0002)	0.8619(0.0001)	0.8406(0.0002)	0.8399(0.0002)	0.8395(0.0002)	0.8443(0.0002)	0.8204(0.0002)	**0.8186(0.0002)**
	MAE	0.5466(0.0026)	0.5171(0.0001)	0.4842(0.0001)	0.4854(0.0001)	0.5186(0.0001)	0.5101(0.0001)	0.5101(0.0001)	0.5097(0.0001)	0.5043(0.0001)	0.4805(0.0001)	**0.4792(0.0001)**
SARCOS	RMSE	2.8154(0.0139)	3.5347(0.0257)	3.1932(0.0160)	4.0727(0.0266)	2.7925(0.0115)	2.7027(0.0126)	2.6982(0.0117)	2.9609(0.0096)	2.7020(0.0105)	**1.9940(0.0250)**	2.0772(0.0030)
	NMSE	0.1269(0.0002)	0.2855(0.0008)	0.1002(0.0000)	0.1734(0.0001)	0.1290(0.0002)	0.1202(0.0001)	0.1206(0.0001)	0.1320(0.0001)	0.1243(0.0005)	**0.0629(0.0001)**	0.0701(0.0001)
	MAE	2.0675(0.0067)	2.5906(0.0119)	2.1028(0.0072)	2.7519(0.0155)	2.0347(0.0075)	1.9556(0.0062)	1.9581(0.0064)	2.1275(0.0053)	1.9580(0.0056)	**1.4035(0.0031)**	1.4571(0.0024)
RF1	RMSE	7.0194(0.2333)	5.7904(0.7353)	5.5321(0.2195)	5.8621(0.2280)	7.0376(0.2332)	6.6570(0.1913)	6.6813(0.1732)	6.7618(0.1872)	6.7933(0.2092)	4.1147(0.1423)	**3.9003(0.1085)**
	NMSE	0.1671(0.0019)	0.1833(0.0154)	0.0409(0.0000)	0.0481(0.0000)	0.1686(0.0028)	0.1330(0.0006)	0.1266(0.0003)	0.1310(0.0003)	0.1328(0.0002)	**0.0366(0.0001)**	0.0461(0.0001)
	MAE	4.4970(0.0303)	3.1150(0.0219)	2.3334(0.0191)	2.3558(0.0170)	4.4297(0.0332)	4.1871(0.0229)	4.1904(0.0264)	4.2690(0.0239)	4.3257(0.0258)	**2.0222(0.0180)**	2.1893(0.0117)

Table 3
Statistical test on comparing methods in RMSE. Here, •/∘/ ∗ indicates whether MMTPL is statistically superior/inferior/similar to the comparing method (pairwise t-test at 5% significance level).

Method	Dataset						win/tie/loss counts
	Computer	WQ	Parkinsons	Housing	RF1	SARCOS	for MMTPL
MMTPL	1.5179 (0.0038)	0.8118 (0.0017)	1.8671 (0.0061)	0.1561 (0.0011)	3.9003 (0.1085)	2.0772 (0.003)	/
MTPL	1.6265• (0.0066)	0.8134∗ (0.0016)	1.896∗ (0.0092)	0.1597∗ (0.0011)	4.1147∗ (0.1423)	1.994∗ (0.025)	1/5/0
MMTFL	1.6211• (0.0015)	0.8329∗ (0.0015)	1.9154∗ (0.0039)	0.1677∗ (0.0014)	7.0376• (0.2332)	2.7925• (0.0115)	3/3/0
GBDSP	1.6335• (0.0034)	0.8213∗ (0.0015)	1.9149∗ (0.0205)	0.1766∗ (0.0011)	6.6813• (0.1732)	2.6982• (0.0117)	3/3/0
VSTG	1.5895• (0.0052)	0.8215∗ (0.0015)	1.9053∗ (0.0104)	0.1684∗ (0.0014)	6.6570• (0.1913)	2.7027• (0.0126)	3/3/0
HTEMTL	1.6151• (0.0060)	0.8210∗ (0.0015)	1.8929∗ (0.0060)	0.1702∗ (0.0013)	6.7618• (0.1872)	2.9609• (0.0096)	3/3/0
AutoTR	1.6919• (0.0052)	0.8232∗ (0.0015)	1.9253∗ (0.0143)	0.1726∗ (0.0013)	6.7933• (0.2092)	2.7020• (0.0150)	3/3/0
FORMULA	2.2385• (0.0033)	0.8551• (0.0022)	2.2495• (0.2307)	0.1630∗ (0.0011)	5.7904• (0.7353)	3.5347• (0.0257)	5/1/0
Network Lasso	2.3389• (0.0082)	0.8178∗ (0.0019)	1.9935• (0.0423)	0.1583∗ (0.0011)	5.5321• (0.2195)	3.1932• (0.0160)	4/2/0
Localized Lasso	1.7951• (0.0054)	0.8201∗ (0.0017)	1.9191∗ (0.0057)	0.1577∗ (0.0013)	5.8621• (0.2280)	4.0727• (0.0266)	3/3/0

Table 4
Comparison of MMTPL and MMTPL$_p$ ($p = 1/2$) on SARCOS and RF1 by varying the number of training samples.

SARCOS	100	200	300	500	800	1200	1800
MMTPL	3.0837 (0.0712)	2.5927 (0.0036)	2.4362 (0.0041)	2.2288 (0.0082)	**2.0393 (0.0087)**	**1.9208 (0.0103)**	**1.7985 (0.0075)**
MMTPL$_{1/2}$	**3.0559 (0.0928)**	**2.5918 (0.0037)**	**2.4237 (0.0046)**	**2.2225 (0.0073)**	2.0468 (0.0060)	1.9220 (0.0103)	1.8078 (0.0086)
RF1	100	200	300	500	800	1200	1800
MMTPL	8.8365 (0.8825)	7.4002 (0.6157)	7.2874 (0.2378)	6.4148 (0.2345)	**5.5916 (0.2371)**	**4.6986 (0.4031)**	**3.9003 (0.1085)**
MMTPL$_{1/2}$	**8.6090 (0.3508)**	**7.3738 (0.5964)**	**7.2702 (0.2475)**	**6.3822 (0.2616)**	5.6427 (0.2510)	4.7125 (0.3962)	3.9019 (0.1069)

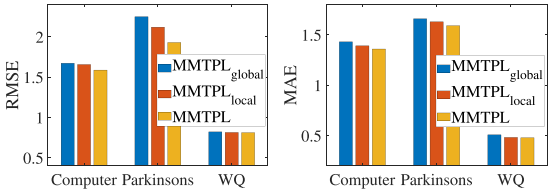


Fig. 5. Analysis on the effect of model decomposition in MMTPL on three datasets. MMTPL$_{global}$ and MMTPL$_{local}$ are two variants of MMTPL, only considering the global model and the local model, respectively.

6.3. Experiments on real-world data

Evaluation of comparing methods. Experimental results are reported in Table 2. We can see that MMTPL performs the best in 11 out of 18 cases. Such performance advantage can be attributed to two aspects. First, MMTPL sufficiently exploits the correlation among samples by considering both sample homogeneity and heterogeneity, and encouraging sparse group structure among samples. Second, MMTPL enables to save task relatedness by capturing task group structure in the feature subspace. MTL methods outperform PL methods on three multi-task datasets, i.e., SARCOS, Parkinsons and Computer. For PL methods in multi-tasking scenarios, MMTPL outperforms MTPL in most cases. The PL methods, in both single-task and multi-task scenarios, performs significantly better than other methods on the Housing and RF1 datasets. It is probably due to the high heterogeneity of samples in the Housing and RF1 datasets. We also provide the statistical test results in Table 3.

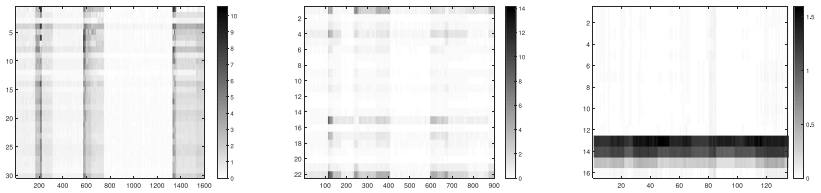
Analysis on the effect of model decomposition. In MMTPL, the personalized model is decomposed into a sum of a global model $\boldsymbol{\theta}_{t,0}$ and a local model $\boldsymbol{\theta}_{t,i}$. To evaluate the effect of model decomposition, an experiment is conducted to compare MMTPL with two special variants: MMTPL$_{global}$ and MMTPL$_{local}$, considering only the global part $\boldsymbol{\theta}_{t,0}$ and the local part $\boldsymbol{\theta}_{t,i}$, respectively. We illustrate the result on the Computer, Parkinsons and WQ datasets in Fig. 5. The results for the other three datasets are similar. From Fig. 5, we can see that MMTPL outperforms MMTPL$_{local}$ and MMTPL$_{global}$ on all three datasets. The superior performance of MMTPL in all cases indicates the importance of saving both heterogeneity and homogeneity of samples by model decomposition.

Comparison of MMTPL and MMTPL$_p$. To avoid severe penalization on the large group outlier, we propose MMTPL$_p$ based on the ℓ_p quasi-norm ($0 < p < 1$) in Section 4.4. To evaluate its effectiveness, we conduct experiments on the SARCOS and RF1 datasets. The two datasets are chosen because there are a sufficient number of samples for each task in both datasets. We fix the number of testing samples to 200 and vary the number of training samples from 100 to 1800. Experimental results are reported in terms of RMSE, and similar results can be observed in other metrics. As shown in Table 4, MMTPL$_p$ ($p = 1/2$) performs better than vanilla MMTPL in the case of fewer training samples, probably because the ℓ_p quasi-norm is a more appropriate approximation to the ideal ℓ_0-norm, compared with the ℓ_1-norm used in MMTPL. In this sense, MMTPL$_p$ will lead to a more aggressive model grouping (as it will more aggressively penalize small model difference $\|\boldsymbol{\beta}_i - \boldsymbol{\beta}_j\|_2$), such that the number of learned parameters is significantly controlled, and thus the generalization ability is guaranteed with less data. However, as the number of samples exceeds 800, MMTPL achieves better performance than MMTPL$_p$. That is because excessive model grouping may weaken the generalization ability, once the sample size is enough.

Group sparse structure on real-world datasets. To visualize the grouping structures of real-world datasets learned by MMTPL, we conduct an experiment on three datasets and illustrate the sparse structure of the learned local model matrix $\boldsymbol{\Theta}_t = [\boldsymbol{\theta}_{t,1}, \boldsymbol{\theta}_{t,2}, \ldots, \boldsymbol{\theta}_{t,n_t}]$ of a specific task ($t = 1$) in Fig. 6. The grouping structure is learned by fixing $\lambda_1 = 2^6$ and $\lambda_3 = 2^0$ ($\lambda_2 = 2^2$ for RF1, $\lambda_2 = 2^0$ for Sarcos and Parkinsons), and then performing k-means ($k = 15$) clustering on the columns of $\boldsymbol{\Theta}_t$. As shown in Fig. 6, a clear grouping structure and row-wise sparse pattern can be observed, which demonstrates the effectiveness of multi-level network Lasso in terms of sample-level model clustering and feature selection. Such ability not only improves the generalization of MMTPL when real-world data exhibits clustering structures in personalized models, but also enhances interpretability by capturing key features crucial in real setting. For example, in the Parkinsons case, MMTPL significantly weights feature 13–15, which include Harmonic-to-Noise Ratio (HNR), Recurrence Period Density Entropy (RPDE), and Detrended Fluctuation Analysis (DFA). These features have been validated in prior research [38,39], demonstrating their significant contributions to predicting scores on the Unified Parkinson's Disease Rating Scale (UPDRS).

Sensitivity analysis. The sensitivity analysis on λ_1, λ_2 and λ_3 is conducted on the SARCOS and Computer datasets. Specifically, λ_1 controls the group sharing among individual models, λ_2 controls the sparsity within each task and λ_3 controls the row-wise sparsity among tasks. Values of λ_1, λ_2 and λ_3 are selected from $\{2^{-10}, 2^{-8}, \ldots, 2^8, 2^{10}\}$. Experiments are conducted to evaluate the pairwise correlation between parameters. The experiment on λ_2 and λ_3 is conducted by fixing $\lambda_1 = 1$, and similar setting is applied for the remains. Fig. 7 shows the results on SARCOS and Computer in RMSE. We can see that the performance is more sensitive to the value change of λ_2 compared to λ_1 and λ_3. We recommend using smaller values for λ_1 and λ_2 ($< 2^2$) and a larger value for λ_3 ($> 2^4$) in practice.

Running time analysis. According to the time complexity analysis in Section 5.3, the theoretical time complexity of MMTPL is quadratic w.r.t the number of samples N. To investigate its practical runtime and potential scalability issues, we plot the execution times for varying numbers of training samples on the SARCOS and RF1 datasets, averaging the results over five runs. The results, as shown in Fig. 8, indicate that the runtime of MMTPL increases significantly with the number of training samples, yet it performs better than the strong competitor, MTPL. Scalability remains a limitation of the current MMTPL; future work will explore more efficient algorithms to improve computational efficiency.



(a) RF1 (b) Sarcos (c) Parkinsons

Fig. 6. Illustration of group sparse structure learned by MMTPL on three datasets with $\lambda_1 = 2^6$ and $\lambda_3 = 2^0$ ($\lambda_2 = 2^2$ for RF1, $\lambda_2 = 2^0$ for Sarcos and Parkinsons). We show the learned local model $\theta_{t,i}$ in tth task ($t = 1$) after k-means ($k = 15$) clustering.

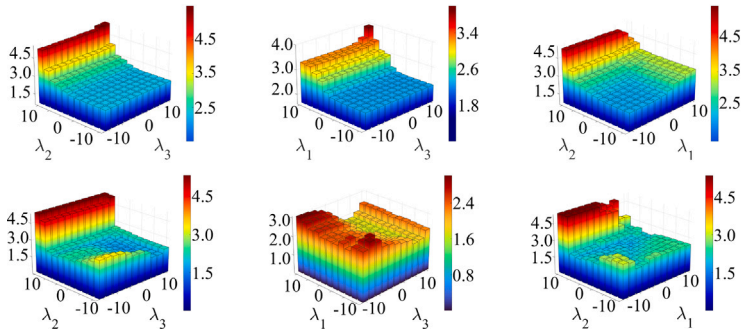
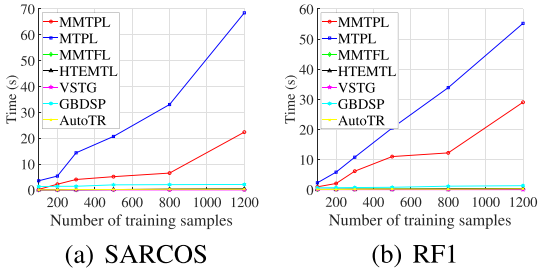
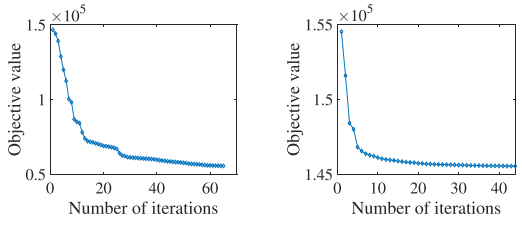


Fig. 7. Sensitivity analysis of λ_1, λ_2 and λ_3. The 1st row shows the results on SARCOS, while the 2nd row shows the results on Computer. The values of λ_1, λ_2 and λ_3 are shown in the logarithmic scale.



(a) SARCOS (b) RF1

Fig. 8. Comparison of methods in running time on the SARCOS and RF1 datasets.



(a) SARCOS (b) Computer

Fig. 9. Convergence analysis of Algorithm 1 on two real-world datasets. The algorithm converges at the 65th and 44th iteration on SARCOS and Computer, respectively.

Convergence analysis. To evaluate the convergence ability of Algorithm 1, we conduct experiment on two real-world datasets, Computer and SARCOS. In this experiment, we set the parameters of MMTPL as $\lambda_1 = \lambda_2 = \lambda_3 = 1$. We terminate Algorithm 1 once the relative change of its objective is below 10^{-3}. Fig. 9 shows the convergence curves of the objective function value by MMTPL. Fig. 9 shows that the objective function value converges after a few number of iterations, demonstrating the efficiency of the proposed algorithm.

7. Conclusion and future work

We propose the multi-level network Lasso used in personalized learning, and further extend it to the MTL scenarios. The proposed multi-level network Lasso has the ability to learn interpretable personalized models by capturing both homogeneity and heterogeneity of samples in feature selection and parameter learning. Moreover, it is extended to handle MTL problems by detecting the latent task group structure in the feature subspace, leading to the MMTPL method. We also investigate a family of multi-level network Lasso based on the ℓ_p quasi-norm, which helps prevent over-penalization on large group outliers. We develop an alternating algorithm to optimize the objective function of MMTPL. Experiments on synthetic and real-world datasets demonstrate its superiority.

One limitation of our multi-level network Lasso is that sample similarity r_{ij} is computed using an RBF kernel with manually tuned parameters, reducing efficiency. Additionally, as noted in , scalability issues emerge with increasing dataset size. Future work will explore using end-to-end deep models like the Siamese Network [40] for more automated similarity calculations and developing more efficient algorithms to address scalability challenges.

CRediT authorship contribution statement

Jiankun Wang: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Luhuan Fei:** Writing – review & editing, Visualization, Validation. **Lu Sun:** Writing – review & editing, Supervision, Resources, Project administration, Methodology, Formal analysis.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work has been partially supported by Startup Funding [2019F0203-000-06] at ShanghaiTech University to L.S.. The authors have no relevant financial or non-financial interests to disclose.

Data availability

Data will be made available on request.

References

[1] D. Hallac, J. Leskovec, S. Boyd, Network lasso: Clustering and optimization in large graphs, in: Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2015, pp. 387–396.
[2] M. Yamada, T. Koh, T. Iwata, J. Shawe-Taylor, S. Kaski, Localized lasso for high-dimensional regression, in: Proceedings of AISTATS 2017, PMLR, pp. 325–333.
[3] J. Li, L. Wu, H. Dani, H. Liu, Unsupervised personalized feature selection, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 32, 2018.
[4] J. Xu, J. Zhou, P.-N. Tan, FORMULA: factorized multi-task learning for task discovery in personalized medical models, in: Proceedings of the 2015 SIAM International Conference on Data Mining, 2015, pp. 496–504.
[5] B. Lengerich, B. Aragam, E.P. Xing, Learning sample-specific models with low-rank personalized regression, in: Advances in Neural Information Processing Systems, Vol. 32, 2019.
[6] M. Petrovich, M. Yamada, Fast local linear regression with anchor regularization, 2020, arXiv:2003.05747.
[7] B. Lengerich, Sample-Specific Models for Precision Medicine (Ph.D. thesis), Carnegie Mellon University, 2020.
[8] S. Dey, P. Zhang, D. Sow, K. Ng, PerDREP: Personalized drug effectiveness prediction from longitudinal observational data, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2019, pp. 1258–1268.
[9] Y. Zhang, Q. Yang, A survey on multi-task learning, IEEE Trans. Knowl. Data Eng. 34 (12) (2022) 5586–5609.
[10] P. Gong, J. Ye, C.-s. Zhang, Multi-stage multi-task feature learning, in: Advances in Neural Information Processing Systems, Vol. 25, 2012.
[11] P. Cao, X. Liu, J. Yang, D. Zhao, M. Huang, O. Zaiane, $\ell_{2,1}-\ell_1$ regularized nonlinear multi-task representation learning based cognitive performance prediction of Alzheimer's disease, Pattern Recognit. 79 (2018) 195–215.
[12] T.K. Pong, P. Tseng, S. Ji, J. Ye, Trace norm regularization: Reformulations, algorithms, and multi-task learning, SIAM J. Optim. 20 (6) (2010) 3465–3489.
[13] P. Cao, X. Shan, D. Zhao, M. Huang, O. Zaiane, Sparse shared structure based multi-task learning for MRI based cognitive performance prediction of Alzheimer's disease, Pattern Recognit. 72 (2017) 219–235.
[14] L. Han, Y. Zhang, Learning multi-level task groups in multi-task learning, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 29, 2015.
[15] X. Wang, J. Bi, S. Yu, J. Sun, M. Song, Multiplicative multitask feature learning, J. Mach. Learn. Res. 17 (80) (2016) 1–33.
[16] C. Liu, C.-T. Zheng, S. Qian, S. Wu, H.-S. Wong, Encoding sparse and competitive structures among tasks in multi-task learning, Pattern Recognit. 88 (2019) 689–701.
[17] Z. Yang, Q. Xu, Y. Jiang, X. Cao, Q. Huang, Generalized block-diagonal structure pursuit: Learning soft latent task assignment against negative transfer, in: Advances in Neural Information Processing Systems, Vol. 32, 2019.
[18] J. Jin, J. Wang, L. Sun, J. Zheng, M. Kudo, Grouped multi-task learning with hidden tasks enhancement, in: Proceedings of the 26th European Conference on Artificial Intelligence, 2023, pp. 1164–1171.
[19] J. Wang, L. Sun, Multi-task personalized learning with sparse network lasso, in: Proceedings of the 31st International Joint Conference on Artificial Intelligence, 2022, pp. 3516–3522.
[20] X. Zhang, J. Liu, Z. Zhu, Learning coefficient heterogeneity over networks: A distributed spanning-tree-based fused-lasso regression, J. Amer. Statist. Assoc. 119 (545) (2024) 485–497.
[21] N. Jethani, M. Sudarshan, Y. Aphinyanaphongs, R. Ranganath, Have we learned to explain?: How interpretability methods can learn to encode predictions in their interpretations, in: Proceedings of AISTATS 2021, PMLR, pp. 1459–1467.
[22] J. Yang, O. Lindenbaum, Y. Kluger, Locally sparse neural networks for tabular biomedical data, in: Proceedings of the 39th International Conference on Machine Learning, 2022, pp. 25123–25153.
[23] G. Obozinski, B. Taskar, M. Jordan, Multi-Task Feature Selection, Tech. Rep, Vol. 2, Statistics Department, UC Berkeley, 2006, p. 2.
[24] H. Liu, M. Palatucci, J. Zhang, Blockwise coordinate descent procedures for the multi-task lasso, with applications to neural semantic basis discovery, in: Proceedings of the 26th International Conference on Machine Learning, 2009, pp. 649–656.
[25] J. Chen, L. Tang, J. Liu, J. Ye, A convex formulation for learning shared structures from multiple tasks, in: Proceedings of the 26th International Conference on Machine Learning, 2009, pp. 137–144.
[26] A. Kumar, H. Daumé, Learning task grouping and overlap in multi-task learning, in: Proceedings of the 29th International Conference on Machine Learning, 2012, pp. 1723–1730.
[27] A. Barzilai, K. Crammer, Convex multi-task learning by clustering, in: Proceedings of AISTATS 2015, PMLR, pp. 65–73.
[28] J.-Y. Jeong, C.-H. Jun, Variable selection and task grouping for multi-task learning, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2018, pp. 1589–1598.
[29] X. He, F. Alesiani, A. Shaker, Efficient and scalable multi-task regression on massive number of tasks, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33, 2019, pp. 3763–3770.
[30] W. Chang, F. Nie, R. Wang, X. Li, Elaborate multi-task subspace learning with discrete group constraint, Pattern Recognit. 139 (2023) 109515.
[31] Y. Zhou, R. Jin, S.C.-H. Hoi, Exclusive lasso for multi-task feature selection, in: Proceedings of AISTATS 2010, PMLR, pp. 988–995.
[32] A.C. Lozano, G. Swirszcz, Multi-level lasso for sparse multi-task regression, in: Proceedings of the 29th International Conference on Machine Learning, 2012, pp. 595–602.
[33] R. Tibshirani, Regression shrinkage and selection via the lasso, J. R. Stat. Soc. Ser. B Stat. Methodol. 58 (1) (1996) 267–288.
[34] R.E. Kuenne, H.W. Kuhn, An efficient algorithm for the numerical solution of the generalized weber problem in spatial economics, in: General Equilibrium Economics: Space, Time and Money, 1992, pp. 223–240.
[35] R. Jenatton, G. Obozinski, F. Bach, Structured sparse principal component analysis, in: Proceedings of AISTATS 2010, PMLR, pp. 366–373.
[36] Y. Nesterov, Introductory Lectures on Convex Optimization: A Basic Course, vol. 87, Springer Science & Business Media, 2013.
[37] M. Zhou, P. Yang, Automatic temporal relation in multi-task learning, in: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2023, pp. 3570–3580.
[38] A. Tsanas, M. Little, P. McSharry, L. Ramig, Accurate telemonitoring of Parkinson's disease progression by non-invasive speech tests, Nat. Preced. (2009) 1.
[39] M. Little, P. McSharry, E. Hunter, J. Spielman, L. Ramig, Suitability of dysphonia measurements for telemonitoring of Parkinson's disease, Nat. Preced. (2008) 1.
[40] G. Koch, R. Zemel, R. Salakhutdinov, et al., Siamese neural networks for one-shot image recognition, in: ICML Deep Learning Workshop, Vol. 2, 2015.

Jiankun Wang received his M.S. degree in computer science from ShanghaiTech University, China, in 2023. He is currently pursuing the Ph.D. degree in computer science at Michigan State University. His research interest is multi-task learning.

LuHuan Fei received her bachelor's degree in computer science from ShanghaiTech University, China, in 2023. She is currently pursuing the M.S. degree in computer science at ShanghaiTech University. Her research interest is machine learning.

Lu Sun received his Ph.D. degree in Information Engineering from Hokkaido University in 2017. He is an assistant professor in ShanghaiTech University. His research interests include pattern recognition, data mining and machine learning.