Automatic Temporal Relation in Multi-Task Learning

Menghui Zhou
Department of Computer Science, The University of Sheffield
Sheffield, UK
menghuicn@sheffield.ac.uk

Po Yang
Department of Computer Science, The University of Sheffield
Sheffield, UK
po.yang@sheffield.ac.uk

ABSTRACT

Multi-task learning with temporal relation is a common prediction method for modelling the evolution of a wide range of systems. Considering the inherent relations between multiple time points, many works apply multi-task learning to jointly analyse all time points, with each time point corresponding to a prediction task. The most difficult challenge is determining how to fully explore and thus exploit the shared valuable temporal information between tasks to improve the generalization performance and robustness of model. Existing works are classified as temporal smoothness and mean temporal relations. Both approaches, however, utilize a predefined and symmetric task relation structure that is too rigid and insufficient to adequately capture the intricate temporal relations between tasks. Instead, we propose a novel mechanism named Automatic Temporal Relation (AutoTR) for directly and automatically learning the temporal relation from any given dataset. To solve the biconvex objective function, we adopt the alternating optimization and show that the two related sub-optimization problems are amenable to closed-form computation of the proximal operator. To solve the two problems efficiently, the accelerated proximal gradient method is used, which has the fastest convergence rate of any first-order method. We have preprocessed six public real-life datasets and conducted extensive experiments to fully demonstrate the superiority of AutoTR. The results show that AutoTR outperforms several baseline methods on almost all datasets with different training ratios, in terms of overall model performance and every individual task performance. Furthermore, our findings verify that the temporal relation between tasks is asymmetrical, which has not been considered in previous works. The implementation source can be found at https://github.com/menghui-zhou/AutoTR.

CCS CONCEPTS

• **Computing methodologies** → **Multi-task learning**; **Regularization**.

KEYWORDS

multi-task learning, automatic temporal relation

KDD '23, August 6–10, 2023, Long Beach, CA, USA

ACM ISBN 979-8-4007-0103-0/23/08…$15.00
https://doi.org/10.1145/3580305.3599261

ACM Reference Format:
Menghui Zhou and Po Yang. 2023. Automatic Temporal Relation in Multi-Task Learning. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23), August 6–10, 2023, Long Beach, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3580305.3599261

1 INTRODUCTION

Multi-task learning [16, 22, 26, 31] is a learning paradigm that aims to leverage valuable information present in multiple related tasks to increase the robustness and generalization performance of the model. Recently, multi-task learning with temporal relation has emerged as a popular and widely used numerical prediction method for the evolution of a wide range of systems, e.g., stock price movements prediction [3], robust key point tracking [32], ensemble forecasting [27], temporal survival analysis [23], road networks prediction [34], and disease progression model [7, 19].

Considering the inherent relations between multiple time points, using multi-task learning to jointly analyse all time points and thus take full advantage of the shared temporal information between tasks is supposed to significantly improve the performance of models [12, 15, 17, 39], especially when data size is limited but feature dimension is high [38, 40]. In this setting, each of the total m time points corresponds to a prediction task and the multi-task model coefficient matrix $W = [\mathbf{w_1}, \cdots, \mathbf{w_m}]$. As shown in Figure 1, the k-th time point represents the k-th task $\mathbf{w_k}$. The critical challenge of this type of temporal multi-task learning is to determine how to fully exploit the shared temporal information between tasks. Existing works are broadly classified as temporal smoothness and mean temporal relations.

The temporal smoothness relation assumes that the difference between two successive tasks is relatively small and thus investigates the temporal relation between multiple tasks. As a result, the methods of multi-task learning with temporal smoothness typically penalize the difference between adjacent tasks $\|\mathbf{w_k} - \mathbf{w_{k+1}}\|_2^2$ in order to achieve temporal smoothness at task level [7, 17, 23, 39]. Some works [34, 38, 40] assume that nearby time points have similar features, so they penalize $\sum_k |w_{i,k} - w_{i,k+1}|$ to pursue temporal smoothness at feature level. Clearly, both task-level and feature-level temporal smoothness seek the same outcome, i.e.,

$$\mathbf{w_k} \approx \mathbf{w_{k+1}}.$$

Unlike the temporal smoothness relation, some works [3, 27, 32] suggest all tasks share an explicit common term $\mathbf{w_0}$ and a task-specific weight $\mathbf{v_k}$, every task can be written as $\mathbf{w_k} = \mathbf{w_0} + \mathbf{v_k}$. This temporal relation structure has been proved in [8] that in the setting of support vector machine, every task has a trend to chase the mean value of all tasks $\sum_{i=1}^{m} \mathbf{w_i}/m$. In Section 3, we demonstrate

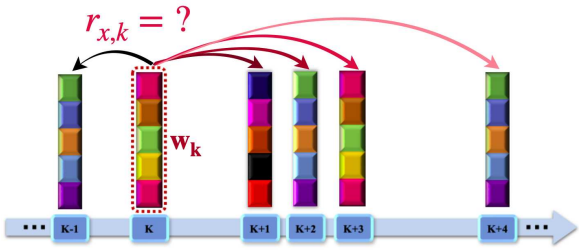

Figure 1: The time points are not evenly distributed. Every time point corresponds to a prediction task. The AutoTR mechanism directly and automatically learns the complex temporal relation between tasks.

that this mean temporal relation actually chases

$$\mathbf{w_k} \approx \frac{1}{m-1}(\mathbf{w_1} + \cdots + \mathbf{w_{k-1}} + \mathbf{w_{k+1}} + \cdots + \mathbf{w_m}).$$

It is concluded that in the temporal smoothness relation, only the relation between adjacent tasks is considered, and the weight of all temporal relations is fixed as 1. So temporal smoothness is a *local, predefined, and symmetric* temporal relation. The mean temporal relation is a global relation structure since it considers the difference between tasks. However, since the weight of all temporal relations is fixed as $(m-1)^{-1}$, it is still a *predefined and symmetric* relation structure.

Due to the utilization of temporal information shared between tasks, introducing temporal smoothness or mean temporal relation into multi-task learning methods has been shown to have a significant positive impact on model in terms of accuracy, robustness, and generalization performance [38]. However, the predefined and symmetric task relation structure is too rigid and insufficient to adequately capture the complex temporal relation among tasks. This work comes from a strong and clear motivation that none of these methods does consider the situation of uneven distribution of time points, as shown in Figure 1, which is usual in real-life applications. For example, based on the data from Alzheimer's Disease Neuroimaging Initiative (ADNI) database [10], many works [15, 38–40] use multi-task learning with temporal relation to predict Alzheimer's disease progression at a sequence of time points, $M00, M06, M12, M24, \cdots, M120$. The notation $M00$ is the baseline time point and Mx represents x months after $M00$. Clearly, the time points are not evenly distributed since the intervals between two successive time points are not the same, i.e., 6 months or a year. Furthermore, even when the time points are evenly distributed, the given time notation is frequently inaccurate. The data at M24 may come from M23, M25, or M26 in practice [24].

To deal with this common but extremely complicated situation, it should be far preferable to learn the complex temporal relation between tasks directly and automatically from the given data, rather than relying on a predefined temporal relation structure. So we name this idea ***Auto**matic **T**emporal **R**elation (**AutoTR**)* and mathematically formulate it as

$$\mathbf{w_k} \approx r_{1,k}\mathbf{w_1} + \cdots + r_{k-1,k}\mathbf{w_{k-1}} + r_{k+1,k}\mathbf{w_{k+1}} + \cdots + r_{m,k}\mathbf{w_m}.$$

Clearly, as shown in Figure 1, $\mathbf{w_k}$ is related to all other tasks $\mathbf{w_i}, \forall i \neq k$. The weight of temporal relation $r_{x,k}$ (the relation from task $\mathbf{w_k}$ to $\mathbf{w_x}$) is not fixed yet and needs to be learned from the data. Another important point is that in AutoTR, the temporal relation is not symmetric as predefined by temporal smoothness or mean temporal relation, since we do not constrain $r_{x,k} = r_{k,x}$. In fact, this asymmetry corresponds to the real-life temporal relation. For instance, refer to Figure 1, $r_{k-1,k}$ represents analyzing the past state of one patient in the current k-th time point, whereas $r_{k,k-1}$ represents predicting future state from $(k-1)$-th time point. They have completely different meanings in practice and should be allowed to have different values, rather than being predefined as the same value which is too strict in real-life applications.

To solve the nonsmooth and biconvex objective function, we adopt the alternating optimization method. The associated two subproblems are amenable to closed-form computation of the proximal operator, resulting in an efficient algorithm based on the accelerated proximal gradient method, which has the best convergence rate of all first-order methods [25]. Furthermore, since there is no theory to guarantee the convergence rate of alternating optimization [28], we design a simple but effective warm start strategy based on the Gaussian kernel to improve efficiency even further. According to the experimental results, this strategy effectively increases efficiency by 17.6 times in terms of computation time at best when compared to the zero initialization that is usually used in multi-task learning literature [11, 13, 28].

We have preprocessed six public real-life datasets and conducted extensive experiments to fully validate the superiority and generalization of the proposed AutoTR. Results show that AutoTR outperforms several baseline methods on almost all six datasets in terms of overall model performance and every individual task performance. It is worth noting that our findings also demonstrate that the temporal relation between tasks is asymmetric, which has not been considered in previous works with temporal smoothness or mean temporal relation.

We conclude this work has the following contributions:

- We propose a novel automatic temporal relation mechanism AutoTR to directly and automatically capture the complex temporal relation among tasks, rather than relying on predefined and symmetric temporal relation structures used in existing baseline methods. An efficient optimization algorithm has been designed based on alternating optimization and a simple but effective warm start strategy.
- We have preprocessed six widely used real-life datasets and conducted extensive experiments. The results demonstrate the superiority of AutoTR in terms of overall model performance and every individual task performance, compared to several baseline methods.
- To explore the complex temporal relation among tasks, we visualize the automatically learned matrix of the temporal relation between tasks. The results also confirm the temporal relation is asymmetry, which is not taken into account by baseline methods. It implies that using a predetermined structure, as existing methods, to investigate the temporal relation between tasks is insufficient.

Notations: $\mathbb{N}_m = \{1, \cdots, m\}$. x_i and $x_{i,j}$ denote the i-th element of a vector $\mathbf{x}$ and the (i,j)-th element of a matrix X. $\mathbf{x_i}$ ($\mathbf{x^i}$) denotes the i-th column (row) of a matrix X. Euclidean and Frobenius

norms are denoted by $\|\cdot\|_2$ and $\|\cdot\|_F$, $\langle A, B\rangle$ is the inner product, $A \odot B$ is component-wise multiplication of A and B. $\|X\|_{p,q} = (\sum_i (\sum_j x_{j,i}^p)^{q/p})^{1/q}$. The component-wise operator $\text{sgn}(\cdot)$ satisfies: $t < 0, \text{sgn}(t) = -1$; $t = 0, \text{sgn}(t) = 0$, and $t > 0, \text{sgn}(t) = 1$.

Organization: The remainder of this work is structured as follows. The related work is in Section 2. In Section 3, we thoroughly discuss the existing baseline works and our proposed automatic temporal relation. We go into great detail about the related optimization algorithm in Section 4. Section 5 presents the experimental findings. Section 6 serves as the conclusion of this paper.

2 RELATED WORK

In this section, we briefly discuss the multi-task learning methods with temporal smoothness or mean temporal relation.

2.1 Temporal Smoothness Relation

Temporal smoothness relation can be divided into two categories, task-level temporal smoothness, and feature-level temporal smoothness. The former assumes the difference between two successive time points is relatively small. Since every time point concerns a task of prediction, it penalizes the difference between two adjacent tasks over time $\|\mathbf{w_k} - \mathbf{w_{k+1}}\|_2^2$. Due to the property of differentiability, the task-level temporal relation will save some computational costs and hence has been widely used in a variety of scenarios, e.g., predicting disease progression [2, 7, 12, 17, 19, 30, 39], online ensemble forecasting [27], air quality inference [33], and tensor-based survival analysis [23]. Different from task-level temporal smoothness relation, several works [38, 40], motivated by predicting the progression of Alzheimer's disease, assume the two neighbouring time points have similar feature sets. They extend the famous fused Lasso [20] to multi-task learning setting to penalize the difference of each feature at two successive time points $\sum_k |w_{i,k} - w_{i,k+1}|$ to chase the feature-level temporal smoothness. Zheng et al. [34] also use the feature-level temporal smoothness for forecasting road travel costs. We conclude that both the task-level and feature-level temporal smoothness relations share the same goal, i.e., $\mathbf{w_k} \approx \mathbf{w_{k+1}}$.

2.2 Mean Temporal Relation

The mean temporal relation assumes every task $\mathbf{w_k}$ shares an explicit common term $\mathbf{w_0}$ and has a task-specific term $\mathbf{v_k}$ and both terms are penalized based on l_2 norm. It also arises in various applications, e.g., the prediction of day trading profit [3], online ensemble forecast [27] and robust key point tracking [32]. Evgeniou et al. [8] prove that in the setting of support vector machine, using the mean temporal relation means every task has a trend to chase the mean value of all tasks $\sum_{i=1}^m \mathbf{w_i}/m$, i.e., every task has the tendency to arrive $\mathbf{w_k} \approx \frac{1}{m-1}(\mathbf{w_1} + \cdots + \mathbf{w_{k-1}} + \mathbf{w_{k+1}} + \cdots + \mathbf{w_m})$.

Despite the fact that both temporal smoothness and mean temporal relations have been shown to significantly improve the capability of the model in terms of generalization performance and robustness, the main limitation is that both are predefined structures that are too rigid and insufficient to fully explore the complex temporal relation between tasks. At the same time, these two temporal relations are symmetric, which is too strict and inconsistent with reality. For example, given two time points $i < j$, the temporal relation from time point i to j likes predicting the state of one patient in the future, but the relation from time point j to i likes analyzing the historic data of the patient. They are clearly different. However, to the best of our knowledge, the asymmetry of temporal relation is not considered in all existing works.

Compared to the baseline methods, our proposed AutoTR directly and automatically learns the complex temporal relation between tasks from every dataset which can even be asymmetric.

3 METHODS

In this section, we introduce the multi-task learning setting and fully explore the temporal smoothness and mean temporal relation. Then we propose our novel automatic temporal relation.

3.1 Multi-task Learning

Assume we're given a series of time points, the total number of which is m. Each time point is associated with a specific task. $\{(X_1, \mathbf{y_1}), \cdots, (X_m, \mathbf{y_m})\}$ is the data, $X_i \in \mathbb{R}^{n_i \times d}$ is the data matrix of the i-th task with each row representing an instance; d is the data dimension; n_i is the number of samples for the i-th task. $\mathbf{y_i} \in \mathbb{R}^{n_i}$ is the target of the i-th task, $\mathbf{y_i}$ has discrete values for classification and continuous values for regression. Denote $W = [\mathbf{w_1}, \cdots, \mathbf{w_m}] \in \mathbb{R}^{d \times m}$ as the weight matrix to be estimated, the empirical risk is given by

$$\mathcal{L}(W) = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{n_i}\sum_{j=1}^{n_i} l((\mathbf{x}_j^{(i)})\mathbf{w_i}, (y_i)_j),$$

where the loss function $l(\cdot,\cdot)$ is square loss for regression problem and logistic loss for binary classification problem.

3.2 Temporal Smoothness Relation

When using a multi-task learning approach, the biggest challenge is capturing and exploiting the complex task relation to improve the generalization performance and robustness of the model. The widely used temporal smoothness relation [17, 19, 38] assumes every time point is similar to its adjacent time points. If each task corresponds to a time point, each task has a tendency to be similar to its neighbouring tasks. To achieve this goal, the models based on temporal smoothness usually penalize the difference between two successive tasks $\|\mathbf{w_i} - \mathbf{w_{i+1}}\|_2^2$ to chase task-level temporal smoothness using the penalty of

$$\lambda \sum\nolimits_{i=1}^{m-1} \|\mathbf{w_i} - \mathbf{w_{i+1}}\|_2^2,$$

where λ is a fine-tuning hyperparameter to control the degree of the similarity between tasks. Some works, motivated by disease progression modeling [38, 40] or road cost forecasting [34], assume that the nearby time points have similar features, so they penalize $\sum_j |w_{i,j} - w_{i,j+1}|$ to chase the temporal smoothness at feature level using the penalty of

$$\lambda \sum\nolimits_{i=1}^{d} \sum\nolimits_{j=1}^{m-1} |w_{i,j} - w_{i,j+1}|.$$

We conclude that both task-level and feature-level temporal smoothness relations share the same goal, i.e.,

$$\mathbf{w_i} \approx \mathbf{w_{i+1}}.$$

Despite that many experiments have proved that the introduction of temporal smoothness can effectively enhance the model

performance, it is actually only a *local, predefined, and symmetric* temporal relation. To make our statement clear, we explain it from the perspective of graph theory. If each task is viewed as a node, the temporal relation between a pair of nodes is an edge, so all tasks and their temporal relation form a graph. However, the adjacency matrix of the temporal smoothness relation graph is a fixed and symmetric tridiagonal matrix as

$$\begin{bmatrix} 0 & 1 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix}.$$

3.3 Mean Temporal Relation

Unlike the temporal smoothness relation, some works [3, 27, 32] assume all tasks share an explicit common term $\mathbf{w_0}$ and a task-specific weight $\mathbf{v_i}$, every task can be written as $\mathbf{w_i} = \mathbf{w_0} + \mathbf{v_i}$. Both $\mathbf{w_0}$ and $\mathbf{v_i}$ are penalized to chase the complex temporal relation using

$$\frac{\lambda_1}{m} \sum_{i=1}^{m} \|\mathbf{v_i}\|_2^2 + \lambda_2 \|\mathbf{w_0}\|_2^2,$$

where λ_1 and λ_2 are two fine-tuning hyperparameters. Evgeniou et al. [8] prove that in the setting of support vector machine, this temporal relation structure is equivalent to

$$\lambda_1 \sum_{i=1}^{m} \|\mathbf{w_i}\|_2^2 + \lambda_2 \sum_{i=1}^{m} \|\mathbf{w_i} - \frac{1}{m} \sum_{j=1}^{m} \mathbf{w_j}\|_2^2.$$

It is clear that every task has a trend to chase the mean value of all tasks $\sum_{j=1}^{m} \mathbf{w_j}/m$. Actually, this mean temporal relation chases the following type of temporal relation:

$$\mathbf{w_k} \approx \frac{1}{m-1}(\mathbf{w_1} + \cdots + \mathbf{w_{k-1}} + \mathbf{w_{k+1}} + \cdots + \mathbf{w_m}).$$

The corresponding adjacency matrix of the mean temporal relation graph is

$$\frac{1}{m-1} \begin{bmatrix} 0 & 1 & \cdots & 1 & 1 \\ 1 & 0 & \cdots & 1 & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & 1 & \cdots & 0 & 1 \\ 1 & 1 & \cdots & 1 & 0 \end{bmatrix}.$$

So obviously, the mean temporal relation is a global relation structure since every task $\mathbf{w_k}$ is related to all other tasks $\mathbf{w_x}, \forall x \neq k$. However, all temporal relations have the same weight fixed as $(m-1)^{-1}$, the mean temporal relation is still a *predefined and symmetric* relation structure.

3.4 Automatic Temporal Relation

To overcome these drawbacks, first of all, we assume that each task is connected to every other task and the weight of temporal relation should be learned directly and automatically from data, rather than being predefined. So we formulate this type of temporal relation mathematically as

$$\mathbf{w_k} \approx r_{1,k}\mathbf{w_1} + \cdots + r_{k-1,k}\mathbf{w_{k-1}} + r_{k+1,k}\mathbf{w_{k+1}} + \cdots + r_{m,k}\mathbf{w_m}.$$

Obviously, $\mathbf{w_k}$ is related to all other tasks $\mathbf{w_i}, \forall i \neq k$. The weight of temporal relation $r_{x,k}$ (the relation from task $\mathbf{w_k}$ to $\mathbf{w_x}$) is not fixed yet and needs to be learned from data. Another key point is in this structure, the temporal relation is not symmetric as predefined in temporal smoothness or mean temporal relation, since we do not constrain $r_{x,k} = r_{k,x}$. In fact, this asymmetry is consistent with the real-life temporal relation. For example, analysing the past state of one patient in the present moment is not the same as predicting the future state, i.e., $r_{x,k} \neq r_{k,x}, \forall x \neq k$.

Not only that, we do not assume that tasks are necessarily similar to others, i.e., we do not constrain $r_{x,k} \geqslant 0$. In fact, as the results show in Section 5, we found that sometimes tasks will have a negative relation, although very slightly, with $r_{x,k} < 0$. It means they slightly repel, rather than approximate each other. This negative temporal relation has never been studied in all existing works.

After integrating the temporal relation between all tasks, we have

$$W \approx W \begin{bmatrix} 0 & r_{1,2} & \cdots & r_{1,m} \\ r_{2,1} & 0 & \cdots & r_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ r_{m-1,1} & r_{m-1,2} & \cdots & r_{m-1,m} \\ r_{m,1} & r_{m,2} & \cdots & 0 \end{bmatrix} = WR, \tag{1}$$

where R is the adjacency matrix of temporal relation between tasks.

Based on the above descriptions, we propose a novel mechanism, termed ***Auto**matic **T**emporal **R**elation* (AutoTR), to automatically capture the complex temporal relation among tasks:

$$\min_{W,R} \mathcal{L}(W) + \lambda_1 \|W - WR\|_F^2 + \lambda_2 \|R\|_{1,1},$$
$$\text{s.t.} \quad r_{i,i} = 0, i \in \mathbb{N}_m. \tag{2}$$

The first penalty $\|W - WR\|_F^2$ is applied to chase the complex temporal relation among all tasks. We use the second penalty $\|R\|_{1,1}$ to encourage only tasks that are most pertinent to share common temporal information.

Note that the penalty $\|W - WR\|_{1,1}$ is an alternate option to chase the temporal relation, however, with extremely expensive computational costs. Please refer to Section 4 for the detailed discussion about the reason for using $\|W - WR\|_F^2$, rather than $\|W - WR\|_{1,1}$.

In order to constrain $r_{i,i} = 0$, we need to penalize the main diagonal elements of R much more heavily than other entries. So we introduce the auxiliary matrix S which is formulated as

$$S = (s-1) \cdot I_{m \times m} + \mathbf{1}_{m \times m}.$$

The optimization problem (2) becomes

$$\min_{W,R} \mathcal{L}(W) + \lambda_1 \|W - WR\|_F^2 + \lambda_2 \|R \odot S\|_{1,1}. \tag{3}$$

It is worth noting that s is only a "pseudo" hyperparameter, not a true hyperparameter like λ_1 and λ_2. We just need to give s an enough large number to constrain $r_{i,i} = 0, \forall i \in \mathbb{N}_m$. In our experimental setting, we let $s = 10^9$ to achieve the constraint of $r_{i,i} = 0$. It is concluded that introducing the auxiliary matrix S will not increase the computational complexity of the associated optimization problem.

Note that the optimization problem (2) is biconvex which implies we can employ the alternating optimization algorithm to update both variables W and R. Based on AutoTR, we have the capability of

automatically and directly learning the complex temporal relation among tasks from every specific dataset.

4 OPTIMIZATION

The objective function (3) is not easy to solve, since it is nonsmooth and biconvex. In this section, we first introduce the whole alternating optimization for solving (3). Then we show how to adopt the accelerated proximal gradient method (APM) [25] to solve the associated two suboptimization problems about W and R with high efficiency.

The alternating optimization is widely used for solving the biconvex objective function [11, 13, 28]. We need to optimize W and R alternately. The optimization procedure is stopped when the relative change of objective function value ΔF at two successive iterations is not bigger than the threshold ϵ.

4.1 Accelerated Proximal Gradient Method

To update W and R efficiently, we use the accelerated proximal gradient method (APM). Because of the fastest convergence rate for the class of first-order methods, APM has been widely used to address issues with multi-task learning [9, 36]. It has the following form:

$$\min_W F(W) = f(W) + g(W), \tag{4}$$

where $f(W)$ is smooth and convex, and $g(W)$ is nonsmooth and convex. APM is built on two sequences, the search point $\{S^k\}$ and the approximation point $\{W^k\}$. S^k is a linear combination of W^{k-1} and W^k.

$$S^{k+1} = W^k + \alpha_k(W^k - W^{k-1}),$$

where α_k is the combination coefficient. According to [1], let $\alpha_k = \frac{(t_{k-1}-1)}{t_k}$, $t_0 = 1$ and $t_k = \frac{1}{2}(1 + \sqrt{4t_{k-1}^2 + 1})$ for $k \geqslant 1$.

The approximation point W^k is computed as

$$W^k = \pi(S^k - \eta_k \nabla f(S^k)), \tag{5}$$

where η_k is the chosen step size, $\pi(V)$ is the proximal operator of V. The global convergence of APM is dependent on an appropriate step size of η_k. Many sophisticated line search schemes [4] can estimate the step size η_k. Updates are made to the value of η_k up until the following condition is met

$$\begin{aligned} f(W^k) \leqslant & f_\eta(W^k, S^k) \\ = & f(S^k) + \langle \nabla f(S^k), W^k - S^k \rangle + \frac{1}{2\eta_k}\|W^k - S^k\|_F^2. \end{aligned} \tag{6}$$

Please refer to Appendix for the pseudocode of APM.

Note that the computation of the proximal operator (5) is the crucial step in using APM. The complexity for solving (5) dominates the whole complexity of APM-based algorithms. As usual, the proximal operator of the nonsmooth part is not easy to solve, e.g., [38, 40]. However, in our proposed AutoTR (3), updating W does not involve computing the proximal operator of the nonsmooth term. When updating R, the proximal operator admits a closed-form solution, which enables to the design of efficient algorithm.

4.2 Fix R, Update W

When updating W, we fix R, the suboptimization problem is

$$\min_W \mathcal{L}(W) + \lambda_1\|W - WR\|_F^2. \tag{7}$$

Both two terms are smooth and differentiable, we can directly use the accelerated gradient descent to solve (7) efficiently.

4.3 Fix W, Update R

When updating R, we fix W and the sub-optimization problem is

$$\min_R \lambda_1\|W - WR\|_F^2 + \lambda_2\|R \odot S\|_{1,1}. \tag{8}$$

To obtain the proximal operator of $\lambda_2\|R \odot S\|_{1,1}$, we need to resolve

$$\pi(R) = \arg\min_Q \frac{1}{2}\|Q - R\|_F^2 + \lambda_2\|R \odot S\|_{1,1}. \tag{9}$$

Clearly, (9) is an extension of Lasso problem, we also apply the soft-thresholding method to arrive at the closed-form solution:

$$\pi(R) = \max(|R| - \lambda_2 S, 0) \odot sgn(R). \tag{10}$$

So we need the complexity of $O(m^2)$ to solve (9).

4.4 The Reason for Using $\|W - WR\|_F^2$

Based on the above discussion, here we explain the reason why we choose $\|W - WR\|_F^2$, rather than $\|W - WR\|_{1,1}$, to capture the complex temporal relation between tasks.

If we apply $\|W - WR\|_{1,1}$, the associated optimization problem for updating W becomes from (7) to

$$\min_W \mathcal{L}(W) + \lambda_1\|W - WR\|_{1,1}. \tag{11}$$

We need to minimize the following problem to compute the proximal operator of $\lambda_1\|W - WR\|_{1,1}$ at each iteration:

$$\pi(W) = \arg\min_V \frac{1}{2}\|V - W\|_F^2 + \lambda_1\|V - VR\|_{1,1}. \tag{12}$$

This problem (12) no longer admits a closed-form solution. In fact, we can solve (12) using the alternating direction method of multipliers (ADMM) [5]. Despite ADMM being widely used [11, 28, 40, 40], for a desired accuracy ϵ, the worst-case convergence rate of ADMM is only $O(1/\epsilon^2)$. It is quite slow, and the actual speed of implementation of ADMM may be affected by the penalty parameter ρ chosen [29]. It is concluded that applying $\|W - WR\|_{1,1}$ will result in expensive computational costs for updating W.

Similarly, if $\|W - WR\|_{1,1}$ is applied, the associated optimization problem for updating R becomes from (8) to

$$\min_R \lambda_1\|W - WR\|_{1,1} + \lambda_2\|R \odot S\|_{1,1}. \tag{13}$$

Due to the all non smooth terms, (13) is challenging to solve. The Subgradient method [4] is a viable option. However, the low convergence rate of the subgradient method, say $O(1/\epsilon^2)$ for a desired accuracy ϵ, will also lead to extremely expensive computational costs for updating R.

We conclude that the utilization of $\|W - WR\|_F^2$ is for reducing the computational cost. $\|W - WR\|_{1,1}$ is an alternative option for capturing the complex temporal relation between tasks, however, only from the perspective of theory. In practice, we can hardly accept such expensive computational costs leading by the use of $\|W - WR\|_{1,1}$.

4.5 Complexity Analysis

For simplicity, we make an assumption that each task has identical n training samples. The computational cost of our proposed optimization algorithm is composed of two parts, the complexity of updating W and R.

4.5.1 The Complexity of Updating W. When optimizing W, each iteration needs to compute the gradient of $\mathcal{L}(W)$ with the complexity of $O(nmd + m^2(m+d))$. So in the procedure of updating W, each iteration has the complexity of $O(nmd + m^2(m+d))$. Here we emphasize that in our implementation MATLAB code, we compute the loss part $\mathcal{L}(W)$ parallelly with the complexity of $O(nd)$, so the complexity of every iteration reduces to $O(nd + m^2(m+d))$. The convergence rate of APM is proved to be $O(1/\sqrt{\epsilon})$ iterations for a desired accuracy ϵ [14, 25], so the overall complexity for updating W is $O\left((nd + m^2(m+d))/\sqrt{\epsilon}\right)$.

4.5.2 The Complexity of Updating R. When optimizing R, each iteration needs to compute the gradient of smooth part $\lambda_1\|W - WR\|_F^2$ and the proximal gradient of non-smooth part $\lambda_2\|R \odot S\|_{1,1}$. The complexity for computing the gradient is $O(m^2 d)$. The cost for computing the proximal operator of $\lambda_2\|R \odot S\|_{1,1}$ is $O(m^2)$. So for updating R, each iteration has the complexity of $O(m^2 d)$ and the overall complexity for updating R is $O\left(m^2 d/\sqrt{\epsilon}\right)$.

4.5.3 The Overall Complexity of Algorithm 1. In the alternating optimization procedure, W and R will be updated once each as a full iteration. Therefore, a full iteration has a complexity of

$$O\left(\frac{nd + m^2(m+d)}{\sqrt{\epsilon}}\right).$$

4.6 A Warm Start Strategy

Note that, there is currently no theory work that can guarantee the convergence rate of the alternating optimization [28]. In order to further improve the efficiency, we propose a warm start technique to initialize R. Specifically, this strategy starts from an intuitive idea that the larger the interval between two time points, the less similar they are. We use a variant of the Gaussian kernel to measure the similarity between two time points i and j. The corresponding weight of the temporal relation is initialized as

$$r_{i,j} \xlongequal{\text{Initialize}} \begin{cases} \dfrac{e^{-|i-j|}}{\sum_{i=1, i\neq j}^{m} e^{-|i-j|}}, \forall i \neq j \\ 0, \qquad i = j. \end{cases}$$

This simple warm start strategy has been shown to effectively enhance the efficiency by 17.6 times at best, compared to initialization using the zero matrix in our experiments. It is worth noting that we can use $e^{-|i-j|^\alpha}$ to propose different initialization strategies. The parameter α adjusts the decay of the temporal relation. In fact, we have tried $\alpha \in \{0.5, 1, 2, e, 5, 10\}$ and it works best when $\alpha = 1$. As a result, in this work, we uniformly set $\alpha = 1$.

5 EXPERIMENT

In this section, we will first introduce the six datasets we used, the method of preprocessing, and the detailed information for each dataset. We test the effect of the proposed warm start strategy. Then, we compare the performance of our proposed AutoTR to baseline methods on six datasets. We also visualize the learned temporal relation matrix. The hardware condition is an Apple M1 Max chip with 32 GB memory. The implementation code runs on MATLAB.

Table 1: Comparing the iteration number of AutoTR-0 and AutoTR-w on six datasets.

Dataset	Method	Stopping Criterion ($\leqslant \tau$) 10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}
Motor UPDRS	AutoTR-0	3.4	5.6	27.4	416.6	605
	AutoTR-w	2	3.6	4.4	19.8	242.6
	Rate (%)	41 ↓	36 ↓	84 ↓	95 ↓	60 ↓
Total UPDRS	AutoTR-0	3.2	5.4	11.2	405.2	558.4
	AutoTR-w	2.6	3	6.2	14.8	351.2
	Rate (%)	19 ↓	44 ↓	45 ↓	**96** ⇓	37 ↓
Weather	AutoTR-0	3.8	4.4	8.8	24	66.2
	AutoTR-w	2.4	5	13.8	16.4	43.8
	Rate (%)	37 ↓	14 ↑	57 ↑	32 ↓	34 ↓
MMSE	AutoTR-0	4.8	16.4	45.6	173.2	217.6
	AutoTR-w	2.2	4	16	23.6	81.8
	Rate (%)	54 ↓	76 ↓	65 ↓	86 ↓	62 ↓
ADAS-Cog	AutoTR-0	3.4	6	23.2	84.4	193
	AutoTR-w	2.4	3.2	7.6	42.4	85
	Rate (%)	29 ↓	47 ↓	67 ↓	50 ↓	56 ↓
RAVLT	AutoTR-0	4.2	8	28.2	71	206.6
	AutoTR-w	2.2	4.2	5.8	26	67.6
	Rate (%)	48 ↓	48 ↓	79 ↓	63 ↓	67 ↓

Table 2: Comparing the computation time (second) of AutoTR-0 and AutoTR-w on six datasets.

Dataset	Method	Stopping Criterion ($\leqslant \tau$) 10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}
Motor UPDRS	AutoTR-0	0.02	0.05	0.28	2.82	4.3
	AutoTR-w	0.02	0.04	0.05	0.16	1.65
	Rate (%)	7 ↓	30 ↓	82 ↓	**94** ⇓	62 ↓
Total UPDRS	AutoTR-0	0.03	0.04	0.11	2.58	4.13
	AutoTR-w	0.03	0.03	0.08	0.17	2.14
	Rate (%)	5 ↓	35 ↓	25 ↓	93 ↓	48 ↓
Weather	AutoTR-0	10.86	15.81	23.92	103.61	296.59
	AutoTR-w	8.13	22.68	51.01	64.51	208.14
	Rate (%)	25 ↓	43 ↑	113 ↑	38 ↓	30 ↓
MMSE	AutoTR-0	9.76	22.93	65.66	266.76	431.03
	AutoTR-w	7.89	9.35	42.94	86.46	200.1
	Rate (%)	19 ↓	59 ↓	35 ↓	68 ↓	54 ↓
ADAS-Cog	AutoTR-0	5.76	15.14	80.11	370.58	760.03
	AutoTR-w	7.2	10.75	25.49	171.41	350.22
	Rate (%)	25 ↑	29 ↓	68 ↓	54 ↓	54 ↓
RAVLT	AutoTR-0	7.49	20.45	102.52	109.87	489.14
	AutoTR-w	5.69	11.5	16.45	76.5	189.43
	Rate (%)	24 ↓	44 ↓	84 ↓	30 ↓	61 ↓

Table 3: The average rMSE over 5 repetitions with various training ratios is displayed. The bold font highlights superior models.

Dataset	Ratio β	Ridge	Lasso	TaskTS	FeaTS	MeanTR	AutoTR
Motor UPDRS	0.6	10.088 ± 0.779	9.037 ± 0.592	7.763 ± 0.663	7.814 ± 0.916	7.737 ± 0.694	**7.534 ± 0.571**
	0.8	9.303 ± 1.128	8.458 ± 0.227	7.051 ± 0.116	6.976 ± 0.320	7.006 ± 0.204	**6.924 ± 0.232**
Total UPDRS	0.6	13.587 ± 1.145	12.047 ± 0.634	10.095 ± 0.353	9.832 ± 0.094	9.914 ± 0.127	**9.810 ± 0.096**
	0.8	12.906 ± 1.325	11.417 ± 0.738	10.060 ± 0.594	9.715 ± 0.688	9.891 ± 0.559	**9.677 ± 0.643**
Weather	0.6	0.568 ± 0.040	0.577 ± 0.039	0.523 ± 0.022	0.577 ± 0.078	0.550 ± 0.041	**0.466 ± 0.020**
	0.8	0.553 ± 0.038	0.561 ± 0.035	0.504 ± 0.033	0.518 ± 0.030	0.509 ± 0.030	**0.463 ± 0.053**
MMSE	0.6	9.641 ± 0.552	3.785 ± 0.050	3.606 ± 0.116	**3.347 ± 0.081**	3.477 ± 0.046	3.349 ± 0.102
	0.8	8.259 ± 0.228	3.827 ± 0.151	3.673 ± 0.092	3.413 ± 0.127	3.518 ± 0.157	**3.320 ± 0.118**
ADAS-Cog	0.6	12.310 ± 0.241	9.874 ± 0.289	8.389 ± 0.214	8.490 ± 0.204	8.454 ± 0.208	**8.140 ± 0.258**
	0.8	11.611 ± 0.223	9.792 ± 0.222	8.352 ± 0.117	8.485 ± 0.108	8.426 ± 0.135	**7.953 ± 0.103**
RAVLT	0.6	4.781 ± 0.042	3.226 ± 0.017	2.807 ± 0.016	2.744 ± 0.034	2.773 ± 0.020	**2.727 ± 0.031**
	0.8	4.580 ± 0.180	3.176 ± 0.074	2.767 ± 0.073	2.706 ± 0.055	2.737 ± 0.058	**2.659 ± 0.040**

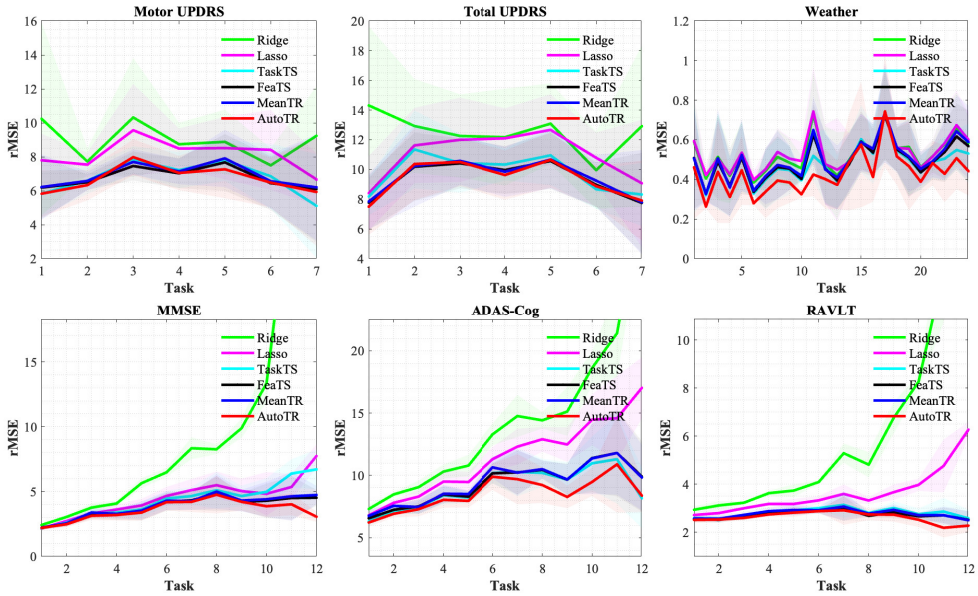

Figure 2: The average rMSE over 5 repetitions of every single task is displayed. The training ratio is 0.8.

5.1 Dataset

We have preprocessed the following six public real-life datasets.

Motor UPDRS and Total UPDRS Datasets [21]: The two datasets are composed of a range of biomedical voice measurements from 42 people with early-stage Parkinson's disease. The goal is to predict the motor and total Unified Parkinson's Disease Rating Scale scores (motor UPDRS and total UPDRS) to estimate the state of Parkinson's disease patients. In these two datasets, there are 18 features, including 16 biomedical features. Every thirty days as a period, we calculate the average UPDRS score for each period. We have seven time points in total.

Weather Dataset [35] [1]: This dataset contains local climatological data for nearly 1,600 locations in the United States from 2010 to 2013, with data points collected every 1 hour. Each data point contains the "Wet Bulb Celsius" target value (the wet-bulb temperature, which is given in tenths of a degree Celsius) and 11 climate features. We want to forecast how the wet bulb will change throughout the day. Each hour counts as one task, for a total of 24 tasks. Each task contains 1461 samples.

MMSE, ADAS-Cog, RAVLT Datasets [10, 24]: These three datasets come from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database [2] has been to ascertain whether serial magnetic

[1] Original Weather dataset: https://www.ncei.noaa.gov/data/local-climatological-data/.
[2] Original ADNI database: https://adni.loni.usc.edu/

resonance imaging (MRI), positron emission tomography (PET), and neuropsychological tests can be used in conjunction to track the development of early AD. After preprocessing , we have data for a total of 12 time points and every simple has 313 MRI features. Following the strategy used in previous works [15, 18, 40], we use the three most common cognitive scores, which are the Mini-Mental State Examination (MMSE), the Alzheimer's Disease Assessment Scale-Cognitive Subscale (ADAS-Cog), and the Rey Auditory Verbal Learning Test (RAVLT), as the response of the model.

5.2 Effectiveness of the Warm Start Strategy

We demonstrate the effectiveness of the proposed warm start strategy, refer to AutoTR initialized with zero as AutoTR-0, and initialized with our proposed warm start strategy as AutoTR-w. Note that for numerical accuracy consideration, we follow the way used in previous works [6, 37] and solve the AutoTR formulation with its objective function multiplied by $\sum_{i=1}^{m} n_i$, where m and n_i correspond to the task number and the sample sizes for task i, respectively.

To thoroughly test the effectiveness of the warm start strategy, we randomly select 5 times of hyperparameters and run them on each of the six datasets. $\lambda_1, \lambda_2 \in \{10^0, 10^1, 10^2, 10^3, 10^4\}$, the pseudo hyperparameter s is 10^9. The feature matrix $X_i, \forall i \in \mathbb{N}_m$, is normalized. When the relative change of objective function value in two successive iterations is not greater than the stopping criterion $\tau \in \{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$, the optimization algorithm is terminated. The maximum iteration is 1000. We record the average number of iterations and the time of algorithm computation on six datasets, respectively.

As shown in Table 1, we discover that the proposed warm start strategy can significantly reduce the number of iterations on all datasets and stopping criteria. When $\tau = 10^{-4}$, the number of iterations on the Total UPDRS dataset is reduced by 96%. When $\tau = [10^{-2}, 10^{-3}]$, the warm start strategy results in a slight increase in the number of iterations, but it has no significant impact because the highest iteration number is only 11.2.

According to Table 1, the warm start strategy can obviously reduce the computation time required for model convergence basically on all datasets and all stopping criteria. In the best case, on the Motor UPDRS dataset, when $\tau = 10^{-4}$, the computation time can be reduced by 94%, i.e., the efficiency is increased by 17.6 times, as it drops from 2.82s to 0.16s. On other datasets and stopping criteria, the warm start strategy continues to improve efficiency in terms of iterations and computation time. These experimental results demonstrate that using a decay strategy based on a Gaussian kernel variant to initialize the temporal relation matrix R is effective.

5.3 Empirical Evaluation

We use the Root Mean Square Error (rMSE) for empirical evaluation. Specially, we not only compute the rMSE of the whole multi-task learning model, but also calculate the rMSE of every single task. The baseline methods include single-task learning with ridge penalty (Ridge), single-task learning with Lasso penalty (Lasso), multi-task learning with task-level temporal smoothness (TaskTS) [17, 19, 38], multi-task learning with feature-level temporal smoothness (FeaTS) [34, 38, 40] , and multi-task learning with mean temporal relation (MeanTR) [3, 27, 32]. The grid range of hyperparameters of all

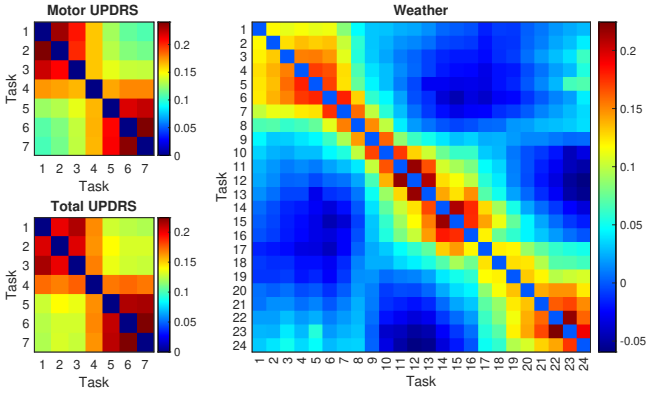

Figure 3: The temporal relation matrix of AutoTR on Motor UPDRS, Total UPDRS, and Weather datasets.

methods is $[10^0, \cdots, 10^4]$, the pseudo hyperparameter s in AutoTR is 10^9. We randomly select β of the dataset as the training set, where the training ratio $\beta \in \{0.6, 0.8\}$ and the rest is used to test. We repeat 5 trials. In each trial, a 2-fold cross validation is applied to select the regularization hyperparameters. The feature matrix $X_i, \forall i \in \mathbb{N}_m$, is normalized. When the relative change of objective function value in two successive iterations is not greater than the stopping criterion $\tau = 10^{-4}$, the optimization algorithm is terminated. The maximum iteration is 1000. We record the average number of rMSE of all methods on six datasets to measure the performance of the whole model and every single task.

First of all, we evaluate the overall performance of the model. According to Table 3, all of the multi-task learning methods TaskTS, FeaTS, MeanTR, and our AutoTR, perform much better than the two single-task learning methods Ridge and Lasso. The method of Ridge in particular consistently has the worst performance. These results demonstrate the effectiveness of using multi-task learning to jointly analyze multiple time points. Note that baseline method FeaTS performs best on the MMSE dataset with rMSE = 3.347 when $\beta = 0.6$, but only slightly better than AutoTR with rMSE = 3.349. The possible reason is that FeaTS introduces the sparsity of first-order difference of the feature weight at two adjacent time points which helps improve the model performance. Furthermore, compared to TaskTS, FeaTS has better performance under most circumstances, except when training ratio $\beta = 0.6$ on Motor UPDRS dataset, TaskTS with rMSE = 0.763 has a little better performance than FeaTS with rMSE = 7.814. TaskTS and FeaTS mainly differ in that FeaTS introduces sparse first-order difference of feature weight, which TaskTS does not. Experimental results show the importance of that kind of sparsity.

It is worth noting that our proposed AutoTR achieves the best performance on most datasets, except the MMSE dataset with training ratio $\beta = 0.6$. On the Weather dataset, when the training ratio $\beta = 0.6$, the rMSE of the best baseline method TaskTS is 0.523, our AutoTR significantly reduces the rMSE to 0.466, almost %11 of rMSE. When the training ratio is 0.8, the best baseline method is also TaskTS with rMSE = 0.505, and AutoTR has greatly reduced rMSE to 0.463. Given that the Weather dataset has the most time points, i.e., 24 tasks, the experimental result proves that our method

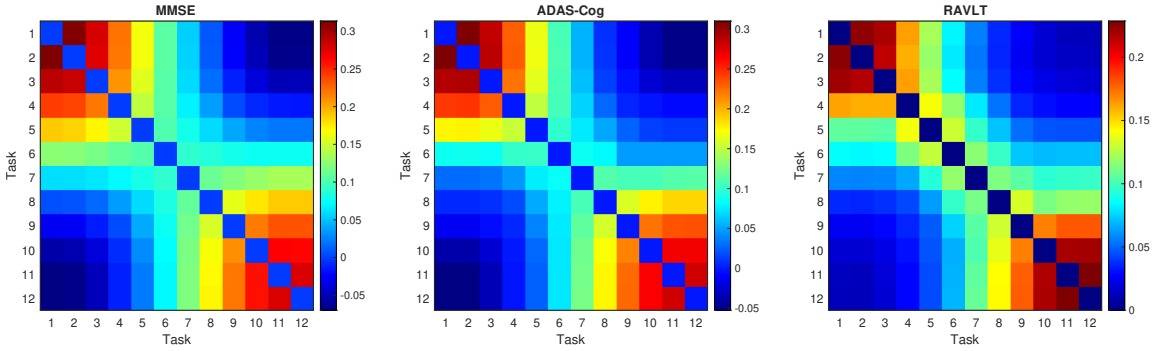

Figure 4: The temporal relation matrix of AutoTR on MMSE, ADAS-Cog and RAVLT datasets.

seems to have better performance with more tasks. In other words, it also shows that when the number of tasks is large, simply utilizing kinds of predefined temporal relations to capture the complex temporal relation between tasks is far from sufficient.

We also demonstrate the superiority of our AutoTR in terms of the performance of every single task. We show the experimental results with training ratio $\beta = 0.8$ in Figure 2. We first notice that Ridge has the highest single task error in most cases, especially in the MMSE, ADAS-Cog, and RAVLT datasets related to Alzheimer's disease, while Lasso has much better performance. The possible reason is all three datasets have 313 features, but the sample size is only several hundred, so feature selection can improve the performance of the model. AutoTR performs significantly better on a single task on two datasets, Weather and ADAS-Cog, than other baseline methods. Note that on the Total UPDRS dataset, the individual task rMSE curves of FeaTS and AutoTR almost overlap, and this result is consistent with the results found in Table 3, as the overall performance of the two models does not differ significantly when $\beta = 0.8$. We conclude that AutoTR has the best single-task performance in most cases compared to baseline methods. This shows that our approach can not only achieve better overall performance but also better individual task performance. It also clearly illustrates the necessity to fully explore and exploit the complex temporal relation between tasks.

In addition, to prove the effectiveness of our method even further, we repeat our experiment 5 times with $\beta = 0.8$. We conduct an independent two-sample t-test and almost all p-values are less than 0.05, except that on the Motor UPDRS dataset, the p-value between FeaTS and out AutoTR is marginally larger, i.e., 0.057.

5.4 Automatically Learned Temporal Relation

To demonstrate other advantages of our approach, we average the learned temporal relation matrix R when conducting a p-value test and visualize it. According to Figure 3 and 4, first and most notable, we clearly find that the temporal relation between tasks can be, although very slightly, negative. It reveals that the temporal relation between tasks is not necessarily similar, and may even be slightly repulsive. Specifically, on the Weather dataset, the pattern of temporal relation is clear. There is a slight negative correlation between tasks near 6 o'clock and 15 o'clock. The possible reason is that the humidity difference between 6 o'clock and 15 o'clock is relatively large. A similar situation also occurs in the tasks around 12 o'clock and 24 o'clock. In addition, we also find a slight negative correlation on the two datasets of MMSE and ADAS-Cog, roughly around the first time point and the 12-th time point. The possible reason is that the tasks are farther apart. It is noted that all the temporal relation matrices learned from each dataset imply that in most cases, the more adjacent tasks are, the stronger the temporal relation is. This also reveals why our proposed warm start strategy based on the decaying mode of the Gaussian kernel can effectively improve the efficiency of the algorithm. Because compared to initializing the temporal relation matrix R with zero, the start point of the warm start strategy is much closer to the optimal temporal relation among tasks. Finally, but perhaps more importantly, we note that none of the temporal relation matrices learned from the datasets is strictly symmetric. This makes sense, temporal relation is not supposed to be symmetric. For example, given two time points $i < j$, the temporal relation from time point i to j likes predicting the state of one patient in the future, but the relation from time point j to i likes analyzing the historic data of the patient. They should be different. However, to the best of our knowledge, the asymmetry of temporal relation is not considered in all existing baseline works.

6 CONCLUSION

In this work, under the setting of multi-task learning, we proposed a novel automatic temporal relation mechanism to fully capture and exploit the complex temporal relation between tasks. In order to improve the efficiency of the optimization algorithm, we designed a warm start strategy based on the Gaussian kernel function. To comprehensively analyze the performance of our method, we preprocessed six public real-life datasets. Extensive experimental results proved that, compared to baseline methods, our algorithm not only has the best overall performance of the whole model but also has the best performance on every single task. The designed warm start strategy, which is simple but effective, can significantly reduce the number of iterations required by algorithm convergence (up to 96%) and the computation time required by the algorithm (up to 94%, 17.6 times efficiency). We also visualized the six temporal relation matrices learned from six datasets, verifying the asymmetry of temporal relation that, however, none of the baseline methods considered. In the future, we will try to explore the effectiveness of our proposed approach AutoTR in other real-life areas, e.g., stock price movement prediction.

ACKNOWLEDGMENTS

This work was supported by UK Research and Innovation (award number: 107462) and Innovate UK (award number: 10050919).

REFERENCES

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

[2] Michele Bernardini, Luca Romeo, Emanuele Frontoni, and Massih-Reza Amini. 2021. A semi-supervised multi-task learning approach for predicting short-term kidney disease evolution. *IEEE Journal of Biomedical and Health Informatics* 25, 10 (2021), 3983–3994.

[3] Zsolt Bitvai and Trevor Cohn. 2015. Day trading profit maximization with multi-task learning and technical analysis. *Machine Learning* 101, 1 (2015), 187–209.

[4] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. 2004. *Convex optimization*. Cambridge university press.

[5] Stephen Boyd, Neal Parikh, and Eric Chu. 2011. *Distributed optimization and statistical learning via the alternating direction method of multipliers*. Now Publishers Inc.

[6] 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*. 42–50.

[7] Saba Emrani, Anya McGuirk, and Wei Xiao. 2017. Prognosis and diagnosis of Parkinson's disease using multi-task learning. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*. 1457–1466.

[8] Theodoros Evgeniou and Massimiliano Pontil. 2004. Regularized multi–task learning. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. 109–117.

[9] Pinghua Gong, Jieping Ye, and Changshui Zhang. 2012. Robust multi-task feature learning. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. 895–903.

[10] Clifford R Jack Jr, Matt A Bernstein, Nick C Fox, Paul Thompson, Gene Alexander, Danielle Harvey, Bret Borowski, Paula J Britson, Jennifer L. Whitwell, Chadwick Ward, et al. 2008. The Alzheimer's disease neuroimaging initiative (ADNI): MRI methods. *Journal of Magnetic Resonance Imaging: An Official Journal of the International Society for Magnetic Resonance in Medicine* 27, 4 (2008), 685–691.

[11] Jun-Yong Jeong and Chi-Hyuck Jun. 2018. Variable selection and task grouping for multi-task learning. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1589–1598.

[12] Biao Jie, Mingxia Liu, Jun Liu, Daoqiang Zhang, and Dinggang Shen. 2016. Temporally constrained group sparse learning for longitudinal data analysis in Alzheimer's disease. *IEEE Transactions on Biomedical Engineering* 64, 1 (2016), 238–249.

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

[14] Huan Li, Cong Fang, and Zhouchen Lin. 2020. Accelerated first-order optimization algorithms for machine learning. *Proc. IEEE* 108, 11 (2020), 2067–2082.

[15] Xiaoli Liu, Peng Cao, André R Gonçalves, Dazhe Zhao, and Arindam Banerjee. 2018. Modeling alzheimer's disease progression with fused laplacian sparse group lasso. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 12, 6 (2018), 1–35.

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

[17] Liqiang Nie, Luming Zhang, Lei Meng, Xuemeng Song, Xiaojun Chang, and Xuelong Li. 2016. Modeling disease progression via multisource multitask learners: A case study with Alzheimer's disease. *IEEE transactions on neural networks and learning systems* 28, 7 (2016), 1508–1519.

[18] Saullo HG Oliveira, André R Gonçalves, and Fernando J Von Zuben. 2022. Asymmetric Multi-Task Learning with Local Transference. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 16, 5 (2022), 1–30.

[19] Luca Romeo, Giuseppe Armentano, Antonio Nicolucci, Marco Vespasiani, Giacomo Vespasiani, and Emanuele Frontoni. 2020. A Novel Spatio-Temporal Multi-Task Approach for the Prediction of Diabetes-Related Complication: a Cardiopathy Case of Study.. In *IJCAI*. 4299–4305.

[20] Robert Tibshirani, Michael Saunders, Saharon Rosset, Ji Zhu, and Keith Knight. 2005. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67, 1 (2005), 91–108.

[21] Athanasios Tsanas, Max Little, Patrick McSharry, and Lorraine Ramig. 2009. Accurate telemonitoring of Parkinson's disease progression by non-invasive speech tests. *Nature Precedings* (2009), 1–1.

[22] Simon Vandenhende, Stamatios Georgoulis, Wouter Van Gansbeke, Marc Proesmans, Dengxin Dai, and Luc Van Gool. 2021. Multi-task learning for dense prediction tasks: A survey. *IEEE transactions on pattern analysis and machine intelligence* (2021).

[23] Ping Wang, Tian Shi, and Chandan K Reddy. 2020. Tensor-based Temporal Multi-Task Survival Analysis. *IEEE Transactions on Knowledge and Data Engineering* (2020).

[24] Michael W Weiner, Paul S Aisen, Clifford R Jack Jr, William J Jagust, John Q Trojanowski, Leslie Shaw, Andrew J Saykin, John C Morris, Nigel Cairns, Laurel A Beckett, et al. 2010. The Alzheimer's disease neuroimaging initiative: progress report and future plans. *Alzheimer's & Dementia* 6, 3 (2010), 202–211.

[25] John Wright and Yi Ma. 2022. *High-dimensional data analysis with low-dimensional models: Principles, computation, and applications*. Cambridge University Press.

[26] Dongbo Xi, Zhen Chen, Peng Yan, Yinger Zhang, Yongchun Zhu, Fuzhen Zhuang, and Yu Chen. 2021. Modeling the sequential dependence among audience multi-step conversions with multi-task learning in targeted display advertising. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 3745–3755.

[27] Jianpeng Xu, Pang-Ning Tan, Jiayu Zhou, and Lifeng Luo. 2017. Online multi-task learning framework for ensemble forecasting. *IEEE Transactions on Knowledge and Data Engineering* 29, 6 (2017), 1268–1280.

[28] Yaqiang Yao, Jie Cao, and Huanhuan Chen. 2019. Robust task grouping with representative tasks for clustered multi-task learning. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1408–1417.

[29] Lei Yuan, Jun Liu, and Jieping Ye. 2013. Efficient methods for overlapping group lasso. *IEEE transactions on pattern analysis and machine intelligence* 35, 9 (2013), 2104–2116.

[30] Yu Zhang, Vitaveska Lanfranchi, Xulong Wang, Menghui Zhou, and Po Yang. 2022. Modeling Alzheimer's Disease Progression via Amalgamated Magnitude-Direction Brain Structure Variation Quantification and Tensor Multi-task Learning. In *2022 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE, 2735–2742.

[31] Yu Zhang and Qiang Yang. 2021. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering* (2021).

[32] Liming Zhao, Xi Li, Jun Xiao, Fei Wu, and Yueting Zhuang. 2015. Metric learning driven multi-task structured output optimization for robust keypoint tracking. In *Twenty-ninth AAAI conference on artificial intelligence*.

[33] Xiangyu Zhao, Tong Xu, Yanjie Fu, Enhong Chen, and Hao Guo. 2017. Incorporating spatio-temporal smoothness for air quality inference. In *2017 IEEE International Conference on Data Mining (ICDM)*. IEEE, 1177–1182.

[34] Jiangchuan Zheng and Lionel M Ni. 2013. Time-dependent trajectory regression on road networks via multi-task learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 27. 1048–1055.

[35] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 35. 11106–11115.

[36] Jiayu Zhou, Jianhui Chen, and Jieping Ye. 2011. Clustered multi-task learning via alternating structure optimization. *Advances in neural information processing systems* 2011 (2011), 702.

[37] Jiayu Zhou, Jianhui Chen, and Jieping Ye. 2011. MALSAR: Multi-task learning via structural regularization. *Arizona State University* (2011).

[38] Jiayu Zhou, Jun Liu, Vaibhav A Narayan, and Jieping Ye. 2012. Modeling disease progression via fused sparse group lasso. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. 1095–1103.

[39] 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*. 814–822.

[40] Menghui Zhou, Yu Zhang, Tong Liu, Yun Yang, and Po Yang. 2022. Multi-task Learning with Adaptive Global Temporal Structure for Predicting Alzheimer's Disease Progression. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2743–2752.

A DETAILED OPTIMIZATION

The pseudocode of the overall alternating optimization algorithm for solving our AutoTR is in Algorithm 1. We summarize the procedure of APM in Algorithm 2.

Algorithm 1 Alternating Optimization for AutoTR.

Input:
$X = [X_1, \cdots, X_m]$: feature dataset for m tasks.
$Y = [\mathbf{y_1}, \cdots, \mathbf{y_m}]$: response for m tasks.
λ_1, λ_2: hyperparameter.
s: the pseudo hyperparameter to constrain $r_{i,i} = 0$.
ϵ: the threshold for terminating the procedure.
Output:
W: the model coefficient matrix.
R: the temporal relation between tasks.

Initialize: $W = 0, R = 0$.
for $k = 1$ *to* $\cdots$ **do**
  Fix R, update W.
  Fix W, update R.
  if $\Delta F \leqslant \epsilon$ **then**
    break
  end if
end for

Algorithm 2 The Accelerated Proximal Gradient Algorithm.

Input:
$X = [X_1, \cdots, X_m]$: feature dataset for m tasks..
$Y = [\mathbf{y_1}, \cdots, \mathbf{y_m}]$: response for m tasks.
Output:
W: the model coefficient matrix.

Initialize: $\eta_0 = 1$, $t_0 = 0$, $t_1 = 1$, $W^1 = W^0$.
for $k = 1$ *to* $\cdots$ **do**
  $\alpha_k = \frac{t_{k-1}-1}{t_k}$, $S^k = W^k + \alpha_k(W^k - W^{k-1})$. ▷ search point
  for $m = 0$ *to* $\cdots$ **do**
    $\eta_k = 2^m \eta_{k-1}$
    Solving (5) for W^{k+1}.
    if (6) is satisfied **then** ▷ line search
      break
    end if
  end for
  $t_k = \frac{1}{2}\left(1 + \sqrt{1 + 4t_{k-1}^2}\right)$
  if convergence crition is satisfied **then**
    Output W^k, break
  end if
end for

B DETAILS OF ADNI DATASETS

Three ADNI Datasets [10, 24]: The Alzheimer's Disease Neuroimaging Initiative (ADNI) database's main objective has been to determine whether serial positron emission tomography (PET), magnetic resonance imaging (MRI), and neuropsychological tests can be used in conjunction to track the progression of early AD. The baseline refers to a patient's initial hospital screening. The follow-up time point is represented as the time point when the baseline started. For example, "M12" denotes a time point that is one year after the initial visit (baseline time point). The most recent ADNI offers follow-up information from up to 120 months for specific patients. However, many participants drop out of the study for a variety of reasons. As a result, the data on fever increases with increasing distance from the baseline time point. Finally, we have information for 12 different time points, ranging from M00 to M120. Please refer to Table 4 for detailed information.

The data preprocessing steps we take are the same as [40]. Finally, we have three datasets, MMSE, ADAS-Cog, and RAVLT for predicting AD evolution. Each dataset receives 314 features in total. Table 4 shows the specifics of the datasets.

Table 4: The sample number at each time point on the MMSE, ADAS-Cog, and RAVLT datasets.

Time point	MMSE	ADAS-Cog	RAVLT
M00	1092	1074	1091
M06	1078	1064	1074
M12	1027	1014	1021
M24	883	867	877
M36	579	556	576
M48	494	483	468
M60	305	299	267
M72	333	327	249
M84	262	259	192
M96	200	200	148
M108	118	118	97
M120	69	69	61

C P-VALUE TEST

We repeat the experiment procedure used in Table 3 5 times and report the p-value to further prove the effectiveness of our method. However, for the sake of paper space, we only show the experimental result with the training ratio $\beta = 0.8$. We conduct an independent two-sample t-test. If the p-value $\leqslant 10^{-3}$, we report 0; if p-value $\geqslant 0.05$, we highlight it.

Table 5: The p-value between different methods, and the training ration $\beta = 0.8$

Method	Ridge	Lasso	TaskTS	FeaTS	MeanTR
Motor UPDRS	0	0	0.032	**0.057**	0.011
Total UPDRS	0	0	0.006	0.043	0
Weather	0	0	0	0	0
MMSE	0	0	0	0.027	0
ADAS-Cog	0	0	0	0	0.006
RAVLT	0	0	0.013	0	0.009

Obviously, except for the p-value of the FeaTS method on the Motor UPDRS dataset which is slightly higher than 0.05, the other values are very small. This proves that the improvement of our method is statistically significant.