Tensor-based Temporal Multi-Task Survival Analysis

Ping Wang, Tian Shi and Chandan K. Reddy, *Senior Member, IEEE*

Abstract—Survival analysis aims at predicting the time to event of interest along with its probability on longitudinal data. It is commonly used to make predictions for a single specific event of interest at a given time point. However, predicting the occurrence of multiple events of interest simultaneously and dynamically is needed in many real-world applications. An intuitive way to solve this problem is to simply apply the standard survival analysis method independently to each prediction task at each time point. However, it often leads to a sub-optimal solution since the underlying dependencies between these tasks are ignored. This motivates us to analyze these prediction tasks jointly in order to select the common features shared across all the tasks. In this paper, we formulate a temporal (Multiple Time points) Multi-Task learning framework (MTMT) for survival analysis problems using tensor representation. More specifically, given a survival dataset and a sequence of time points, which are considered as the monitored time points for the events of interest, we reformulate the survival analysis problem to jointly handle each task at each time point and optimize them simultaneously. We demonstrate the performance of the proposed MTMT model on important real-world datasets, including employee attrition and medical records. We show the superior performance of the MTMT model compared to several state-of-the-art models using standard metrics. We also provide the list of important features selected by our MTMT model thus demonstrating the interpretability of the proposed model.

Index Terms—Multi-task learning; survival analysis; temporal models; regularization; regression analysis.

1 Introduction

The primary goal of survival analysis is to predict time to the event of interest on longitudinal data obtained during a particular observation period [1]. Survival analysis methods have been successfully applied in many real-world applications, such as healthcare [2], social networks [3], reliability [4] and customer lifetime value [5]. One unique phenomenon, known as *censoring*, in the longitudinal data is the occurrence of the event may not always be observed for all instances during the observation due to non-occurrence of the event by the end of the observation or losing follow-up during the observation. Then in the longitudinal data, the event occurrence time of the instances which experienced the event of interest and the censoring time of the censored instances will be recorded as the observation time, respectively. This implies that people need to wait for a long time to collect the training data with sufficient event occurrences in longitudinal studies.

Most of these existing survival analysis methods are developed to solve the survival problem with a single specific event of interest at a given time point. In addition, the effect of the number of events in the dataset on the model performance is not considered in the standard survival analysis methods, even if the percentage of censoring is very high [6]. However, in the real-world applications, it is known that the data about several related longitudinal studies is available and can be modeled simultaneously. For example, in the human resource management domain, the information of employees before leaving, such as years stayed in the company and years worked with the manager, can be easily extracted. In other words, predicting the occurrence of multiple events of interest simultaneously is needed in the real-world applications. In addition, it is also important if we can dynamically follow up the survival status of the instances in each longitudinal study.

A naive way to solve this problem is to simply apply the standard survival analysis method independently to each task at each specific time point. However, it often leads to a sub-optimal solution since the underlying dependencies between these tasks and the correlations of each single task over time are ignored. On one hand, the dependencies between tasks (called *inter-task correlations*) is a good way to handle the insufficiency of event occurrences in each single task to perform the time to event prediction. On the other hand, the survival status of instances at the adjacent time points should be similar. Thus, for each single task, the problems at different time points should also have some correlations (called *intra-task temporal smoothness*). Thus, both types of correlations will need to be incorporated into a model that can jointly analyze multiple tasks by leveraging the limited number of events available for each single task.

In this paper, we will formalize a temporal (**M**ultiple **T**ime points) **M**ulti-**T**ask learning (MTMT) framework using tensor representation for survival analysis to overcome the weakness of the standard survival analysis methods and incorporate the two types of correlations (mentioned above) in the training process. Multi-task learning method is used in the literature to learn common feature subsets across all tasks [1], [7]. More specifically, in our paper, multi-task learning will be dynamically applied to several related survival tasks and the common feature subset will be the unified feature representation across all the tasks and all

P. Wang, T. Shi and C.K. Reddy are with the Department of Computer Science, Virginia Tech, Arlington, VA 22203.
E-mail: ping@vt.edu, tshi@vt.edu, reddy@cs.vt.edu.
Manuscript received June 22, 2018; revised September 12, 2019.

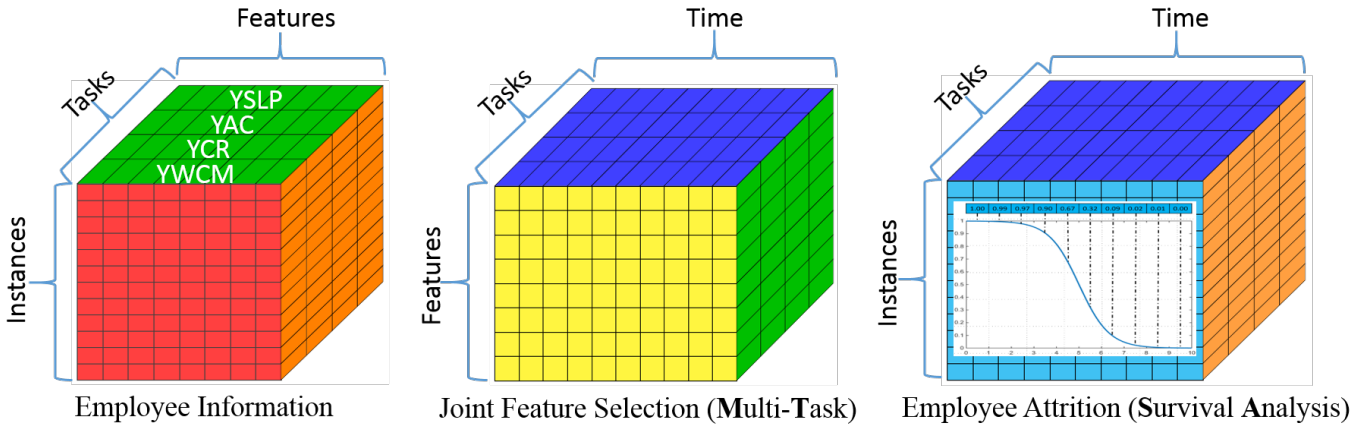


Fig. 1: An illustration of various components in the proposed MTMT framework using Employee Attrition dataset.

time points. Since there are multiple tasks and multiple time points in our problem, a 3-way tensor will also be introduced in this paper for effective representation. Therefore, we extend the standard $\ell_{2,1}$ norm which is usually used on a matrix to the $\ell_{F,1}$ norm (which is appropriate for the tensor representation) to incorporate the inter-task correlations in the proposed MTMT model. Another contribution of the MTMT algorithm is that it performs the multi-task learning dynamically. In this case, the survival status of the instances about multiple events of interest can be monitored over time and then the survival function for the problem in each task can also be obtained. If the standard multi-task learning formulation is used, it will not be able to handle the censored data in survival problems. For solving the survival analysis problems, we will formulate a multi-way indicator tensor which can incorporate the censored instances depending on their survival status over time for each task. In this case, at each time point, one given instance will be involved in the training process as long as it is not censored in at least one task. Figure 1 illustrates the main components of the proposed MTMT method on the Employee Attrition dataset in more detail.

In MTMT, we incorporate both the censoring constraint and two types of correlations by applying regularizations in the loss function of multi-task learning formulation and the Alternating Direction Methods of Multipliers (ADMM) algorithm [8]. In addition, the proximal gradient decent algorithm [9] with backtracking linear search is extended to optimize the $\ell_{F,1}$ regularized problems. The proposed MTMT method will be evaluated on both the employee attrition dataset provided by IBM Watson Analytics and the Medical Information Mart for Intensive Care III (MIMIC III) dataset [10], [11] and compared with the state-of-the-art survival analysis and multi-task learning methods.

The main contributions of this work are summarized as follows:

- Propose a multiple time points multi-task learning (MTMT) model for survival analysis which has the ability to analyze several survival problems jointly and dynamically predict the survival probability at each time point.
- Develop an ADMM based algorithm to optimize the proposed MTMT problem.
- Evaluate the performance of the proposed temporal multi-task learning method using multiple real-world datasets and compare with several state-of-the-art methods.

The rest of this paper is organized as follows: In Section 2, the background and related work about survival analysis and multi-task learning is briefly reviewed. Section 3 introduces the details about the proposed temporal multi-task learning algorithm along with the optimization method used to solve the model. The prediction performance of the proposed MTMT algorithm is evaluated using the real-world datasets in Section 4. Finally, Section 5 concludes the discussion of our work.

2 Related Work

In this section, we discuss the related works on survival analysis, multi-task learning and temporal modeling. We will also distinguish our contribution in this paper from other existing algorithms in the literature.

2.1 Survival Analysis

Survival analysis is a branch of statistics which aims to handle censored data and then predict the time to the event of interest [12]. Typically, survival analysis methods can be grouped into three categories: non-parametric methods, Cox-based methods and parametric methods.

Kaplan-Meier (KM) method [13] is the most commonly used non-parametric survival analysis method due to its reasonable ability to estimate the survival function, which is one of the most important functions in survival problems. Generally, the KM method estimates the survival probability at a given time point during the observation period as the product of the survival rate at this time point and the same estimate at the previous time.

The Cox proportional hazards model [14] is well studied in both statistics and data mining fields due to its flexibility in choosing the baseline hazard function and its ability to provide a more consistent estimation. Cox model is built on a proportional hazard assumption and it estimates the parameters using the partial likelihood function. In order to adapt Cox model to the survival problem with high-dimensional features, the regularized Cox models [15], [16], [17] are proposed to perform the feature selection and identify the most significant features to the outcome variable.

Different from the non-parametric and Cox-based methods, parametric methods assume that the survival time (or

its logarithm) of all instances follow a certain theoretical distribution [18]. The commonly used distributions include exponential, weibull, logistic, log-logistic and log-normal distribution. The accelerated failure time (AFT) model [19] is one of the widely used parametric methods since it assumes that the logarithm of the survival times follows one theoretical distribution. In addition, several linear models, including the Buckley-James (BJ) [20] and Tobit regression [21] are proposed based on the standard linear regression to handle the survival analysis problems.

In order to further tackle the non-linear relationships between covariates and survival time, other researchers have proposed deep survival analysis methods [22], [23], [24], [25]. DeepSurv is proposed as a treatment recommender system that combines the power of both Cox proportional hazards model and deep neural networks and modelling the interactions between covariates and the effect of treatment [22]. The hierarchical generative deep survival analysis approach in [23] is proposed to handle the sparsity and heterogeneity in EHR data using a sequence of multilayer probability models. It is worth to note that all the survival analysis methods mentioned above are suitable for the problems with only one event of interest or at a single time point. Moreover, the number of event occurrences will affect the performance of the standard survival analysis methods, especially when the number of events is insufficient at the early stage of the observation period. An alternative way to compensate for this insufficiency is to make effective use of the event information that is available in other survival problems about a related event of interest. The goal of this paper is to solve the survival analysis problem with multiple events of interest jointly and improve the prediction performance of each single problem. For the survival problem with more than one event of interest, an intuitive way is to simply apply the standard survival algorithm independently on each of the events of interest. However, the correlations between these multiple tasks will be ignored. In this paper, we dynamically consider the survival problem with multiple correlated events of interest by using the multi-task learning method.

2.2 Multi-task Learning

Multi-task learning (MTL) method aims at improving the prediction performance of each task by analyzing multiple tasks jointly and learning shared features across all tasks. Compared to the methods which learn each task independently, it has been shown both theoretically [26], [27] and empirically [1], [7], [28], [29] that multi-task learning simultaneously improves the performance of each task. Multi-task learning has been widely used in many real-world applications, including biomedical informatics [1], [7] and computer vision [28], [29]. However, there is not much work on multi-task learning in the field of survival analysis. The authors in [1] developed a linear regression based multi-task learning formulation (MTLSA) to predict the survival status at each time point. In their method, the survival problem at each time point is considered as one single task and the $\ell_{2,1}$ norm penalty is applied in order to learn a shared feature subset across different tasks. However, only one event of interest is considered in MTLSA and each of the multiple tasks is formulated at each time point, which cannot solve the problems with multiple events of interest as described in our paper. In addition, deep multi-task Gaussian processes are applied in a non-parametric Bayesian model to solve survival problems with competing risks [24]. A neural multi-task logistic regression model is proposed in [25] to offset the linearity problem in traditional survival models. These multi-task learning based methods are able to analyze different survival analysis problems jointly, however, they do not consider the correlations between the problems at different time points.

In our work, we develop a new multi-task learning method to jointly analyze several related survival analysis problems and dynamically perform such multi-task learning to monitor the survival status of instances in each individual task. In other words, our proposed method provides the ability to generate the survival function for each event of interest by predicting the survival probability for each instance over time. To optimize the proposed MTMT method, several types of constraints, including censoring, inter-task correlations and intra-task temporal smoothness, are considered in the objective function to incorporate the unique properties of our survival problem. To the best of our knowledge, this paper is the first work about dynamically applying the multi-task learning in the field of survival analysis and also the first work to solve the survival analysis problem using tensor representation.

3 Proposed Model

In this section, we will first describe the problem formulation in the form of a joint multi-task learning and survival analysis framework by considering multi-task problems dynamically. Then, an ADMM based optimization algorithm will be proposed to solve the joint optimization problem. The convergence and complexity analysis of the proposed algorithm will also be discussed at the end of this section.

3.1 The MTMT Framework

Survival analysis methods are well known for their ability to handle the censored instances in the data in order to predict the time to event of interest. Generally, the standard survival analysis algorithm is proposed to estimate the time to a single event of interest. More specifically, it solves the single task (single event of interest) problem (eg. task k), in which the instance i in survival data is recorded by a triplet $(X_i^k; T_i^k; Y_i^k)$ [1], where $X_i^k \in \mathbb{R}^{1 \times P}$ is a vector which represents the feature vector; Y_i^k is a binary variable which indicates the event status, i.e., $Y_i^k = 1$ if an event is observed on instance i and the corresponding time (O_i^k) for the event is recorded as T_i^k, and $Y_i^k = 0$ if instance i is censored during the observation time period and then its censored time (C_i^k) will be recorded as T_i^k [1]. In other words, for each instance in survival data, only the survival time or the censored time can be observed during the observation time period, i.e.,

$$T_i^k = \begin{cases} O_i^k & \text{if } Y_i^k = 1 \\ C_i^k & \text{if } Y_i^k = 0 \end{cases} \quad (1)$$

It should be noted that the event time O_i^k for the censored instance is a latent value since there is no information about the event status after the censored time C_i^k. One of the

main goals of survival analysis is to measure the survival probability at each time point and estimate the time to the event of interest for each instance.

However, there are certain disadvantages for the standard survival analysis methods. A sufficient number of event occurrences in the training data must be collected by waiting for a long period of time, otherwise having a fewer event occurrences may affect the prediction performance of the survival analysis algorithm, especially in the early stage of the observation [30]. It will be challenging but beneficial for the prediction of each single problem if we can make the best use of the event information in multiple correlated regular survival analysis problems and analyze these individual survival analysis problems in a joint manner. More specifically, the event information in other correlated tasks can be used to compensate for the insufficiency of the event occurrence in a single task and improve the prediction performance. On the other hand, due to the new occurrence of censoring or events on some instances, the values of the event label for these instances will be updated at the corresponding event time or censored time. It will be vital to monitor the survival status of each instance dynamically, which cannot be solved by the existing survival analysis algorithms. These disadvantages motivate the need for developing an algorithm which can analyze multi-task survival analysis problems dynamically.

In this paper, the temporal multi-task learning framework is proposed to estimate the survival probabilities of the instances in K tasks over J time points. Specifically, each event of interest is considered as a single task, and we simultaneously analyze all the tasks by considering both the inter-task correlations and the intra-task time smoothness. Figure 1 shows various components used in the proposed MTMT framework. In the input data, the data record for instance i ($i = 1; \quad ; N_k$) in task k ($k = 1; \quad ; K$) at each time point t_j ($j = 1; \quad ; J$) can be obtained as a triplet $(\mathbf{X}_{i:k}; \mathbf{T}_{ijk}; \mathbf{Y}_{ijk})$ using the following three procedures:

1) $\mathbf{X}_{i:k} \; 2 \; \mathrm{R}^{1 \quad P}$ represents the feature vector for instance i in task k. Due to the difficulty and limitation to collect the feature values over time, the static feature values are commonly used in the survival analysis literature.
2) $\mathbf{Y}_{ijk}$ is the binary variable indicating the survival status of instance i at time point t_j in task k. It is obtained as follows:

$$\mathbf{Y}_{ijk} = \begin{cases} 0 & \text{if } Y_i^k = 1 \text{ and } t_j \quad O_i^k \\ 1 & \text{Otherwise} \end{cases} \tag{2}$$

where $\mathbf{Y}_{ijk} = 1$ indicates that there is no event observed for instance i at t_j in task k and $\mathbf{Y}_{ijk} = 0$ indicates that instance i did not survive after t_j in task k. Eq. (2) indicates that, for an instance having an event during the observation time in task k, it will survive until the event time O_i^k, which is the event time of this instance in the task. After the event time O_i^k, it will be labeled as 0 in the data collected at each time point which is the unique property of the non-recurrent event survival analysis problem. The instances without an event occurrence during the observation time will be labeled as 1 over all time points. It should be noted that these instances have definitely survived at each time point until the censored time, however, the survival status after the censoring time is unknown even if we label it as 1. These unknown records will be carefully handled by an indicator formulation (to be introduced in Subsection 3.2.1).
3) $\mathbf{T}_{ijk}$ is obtained according to the value of Y_i as follows:

$$\mathbf{T}_{ijk} = \begin{cases} O_i^k & \text{if } \mathbf{Y}_{ijk} = 0 \\ \min(t_j; C_i^k) & \text{if } \mathbf{Y}_{ijk} = 1 \text{ and } Y_i^k = 0 \\ \min(t_j; O_i^k) & \text{if } \mathbf{Y}_{ijk} = 1 \text{ and } Y_i^k = 1 \end{cases} \tag{3}$$

It should be noted that each instance is not required to be observed for all tasks. If instance i is not observed in task k, then all values in the triplet $(\mathbf{X}_{i:k}; \mathbf{T}_{ijk}; \mathbf{Y}_{ijk})$ at each time point t_j will be set to 0. This representation leads to equal number of instances in each of the tasks. Based on the data transformation procedures, the entire data for task k at all the time points can be represented as $(\mathbf{X}_{::k}; \mathbf{T}_{::k}; \mathbf{Y}_{::k})$, where $\mathbf{X}_{::k} \; 2 \; \mathrm{R}^{N_k \quad P}$, $\mathbf{T}_{::k} \; 2 \; \mathrm{R}^{N_k \quad J}$ and $\mathbf{Y}_{::k} \; 2 \; \mathrm{R}^{N_k \quad J}$ represent matrices of the features, survival status and event time, respectively. Following the three procedures, the survival problem in each task can be transformed into a sequence of similar problems ordered by time. Our goal is to analyze the $J \quad K$ problems jointly in order to completely utilize the event information available in each task, the inter-task correlations and the intra-task temporal smoothness. It can be observed that this proposed temporal (multi-time points) multi-task problem (MTMT) can be considered as a generalized framework for solving the survival analysis problems. Other existing problems in the literature can be considered as the special cases of MTMT as described below:

1) The standard survival analysis problem is a special case of MTMT with single task at single time point (STST), in which $J = 1$ and $K = 1$.
2) The MTLSA method proposed in [1] can be considered as a special case of MTMT with single task at multi-time points (MTST), in which $K = 1$.
3) Besides STST and MTST problems, MTMT can also incorporate the method in [31] as a multiple tasks at a single time point (STMT), in which $J = 1$.

All the three problems can be derived from our generalized MTMT framework. Therefore, it is important to find a way to solve the MTMT algorithm efficiently. A commonly used approach to solve the formulated temporal multi-task learning problems is to optimize $J \quad K$ linear regression problems as follows:

$$\min_{\mathbf{B}} \sum_{j=1}^{J} \sum_{k=1}^{K} \frac{1}{2} jj\mathbf{Y}_{:jk} \quad \mathbf{X}_{::k}\mathbf{B}_{:jk}jj_2^2 + R(\mathbf{B}) \tag{4}$$

where $\mathbf{X}_{::k} \; 2 \; \mathrm{R}^{N_k \quad P}$ is the feature matrix of the k^{th} task, and $\mathbf{Y}_{:jk} \; 2 \; \mathrm{R}^{N_k \quad 1}$ and $\mathbf{B}_{:jk} \; 2 \; \mathrm{R}^{P \quad 1}$ represent the response vector and the estimated coefficient vector in the k^{th} task at time point t_j, respectively. It should be noted that the estimated coefficients across all the tasks and all the time points can also be represented as a 3-way tensor $\mathbf{B} \; 2 \; \mathrm{R}^{P \quad J \quad K}$ since we assume that the features in different tasks are homogeneous in this work. The term $R(\mathbf{B})$ in Eq. (4) denotes the regularization which is used to avoid the over-fitting of the model and to incorporate other constraints for the parameters.

3.2 Constraints and Regularization Function

There are mainly two challenges in the proposed MTMT algorithm. (i) As defined in Eq. (2), we only know that the censored instances survive until the censoring time, while the survival status after the censoring time is not available. Either treating them as survival or non-survival may introduce some bias into the model. (ii) *The goal of the MTMT algorithm is to incorporate the inter-task correlations and intra-task temporal smoothness into the algorithm in order to estimate the survival probabilities over time* and increase the prediction performance of each single task. It is critically important to find a reasonable way to incorporate both properties into the algorithm. In this subsection, we will discuss and determine suitable regularization terms to incorporate these characteristics into the proposed model.

3.2.1 Censoring

Censoring is one of the main characteristics of survival data, i.e., the components for the censored instances in the survival status vector $\mathbf{Y}_{:jk}$ for task k at time point t_j is latent since we do not know the exact time until which the instances survive beyond the censoring time [2]. The censoring problem in the temporal multi-task learning algorithm is more complex due to the dynamically changing behavior of the survival status in each task. More specifically, the censoring time or event time, for one instance, in different tasks may be different, which also leads to the difference in survival status at different time points in different tasks. Thus it is important to handle this censored information in order to jointly utilize the data in each task and improve the prediction in each single task. We formulate a 3-way indicator tensor to handle these censored instances in the training process.

For the task k at time point t_j, each component indicator $\mathbf{S}_{ijk}$ defined in Eq. (5) is used to indicate the extent of contribution of the survival information about instance i in the model.

$$\mathbf{S}_{ijk} = \begin{cases} 0 & \text{if } \mathbf{Y}_{ijk} = 1 \text{ and } t_j > C_i^k \\ 1 & \text{Otherwise} \end{cases} \tag{5}$$

Then the objective function in Eq. (4) can be updated to handle the censoring information as:

$$\min_{\mathbf{B}} \sum_{j=1}^{J} \sum_{k=1}^{K} \frac{1}{2} \| \mathbf{S}_{:jk} \circledast (\mathbf{Y}_{:jk} - \mathbf{X}_{::k} \mathbf{B}_{:jk}) \|_2^2 + R(\mathbf{B}) \tag{6}$$

where the symbol $\circledast$ which represents the component-wise multiplication of two vectors is defined as $a \circledast b = \sum_i a_i b_i$. By applying the indicator tensor $\mathbf{S}$, one instance will contribute to the training process only if it is not censored in at least one of the tasks, no matter if it is survived or not in these tasks.

3.2.2 Overfitting

In order to avoid the overfitting of the problem, a ℓ_F regularization [32] on the coefficient tensor $\mathbf{B}$ defined in Eq. (7) will be used.

$$\| \mathbf{B} \|_F^2 = \sum_{k=1}^{K} \| \mathbf{B}_{::k} \|_F^2 \tag{7}$$

3.2.3 Inter-task correlations

One of the goals of multi-task learning is to compensate for the insufficiency of the event occurrences in each task and to learn shared features among all the tasks by jointly learning the models. In other words, multi-task learning aims at identifying the most important and common features that contribute to the survival of the instances across all the tasks. The $\ell_{2,1}$ norm on a matrix is commonly used to learn the shared features since it tends to result in similar sparsity patterns for the parameters in all tasks [1]. Here, we extend the $\ell_{2,1}$ norm on the matrix to the $\ell_{F,1}$ norm, defined in Eq. (8), on the tensor $\mathbf{B}$, which will lead to shared features across all tasks at all time points.

$$\| \mathbf{B} \|_{F,1} = \sum_{p=1}^{K} \sqrt{\| \mathbf{B}_{p::} \|_F^2} \tag{8}$$

where $\| \mathbf{B}_{p::} \|_F^2 = \sum_{k=1}^{K} \sum_{j=1}^{J} \mathbf{B}_{pjk}^2$.

3.2.4 Intra-task temporal smoothness

One of the main objectives of the MTMT algorithm is to dynamically perform the multi-task learning since the survival status of each instance in each task keeps changing over time. By dynamically learning the multi-task problem, we can monitor the survival status for each instance over time, incorporate the updated survival status in the training process and track the most significant features across all the tasks. According to the definition of the indicator tensor $\mathbf{S}$ in Eq. (5), there exits a high temporal dependency between the problems at different time points. In this paper, an intra-task temporal smoothness regularization term defined in Eq. (9) is introduced to reduce the parameter deviations estimated at adjacent time points [7].

$$\| \mathbf{B} \|_{\text{ts}} = \sum_{k=1}^{K} \sum_{j=1}^{J-1} \| \mathbf{B}_{:(j+1)k} - \mathbf{B}_{:jk} \|_2^2 \tag{9}$$

It should be noted that similar to fused lasso [33], Eq. (7) and Eq. (9) are designed to prevent overfitting and encourage intra-task temporal smoothness, respectively. However, fused lasso introduces sparsity to parameters and differences of parameters for adjacent time points, which is not required in our problem. Therefore, we only consider regularizers with Euclidean norm to decay weights (i.e., model parameters) and parameter differences, instead of introducing sparsity.

After incorporating all the constraints into Eq. (4), we can solve the proposed temporal multi-task learning framework by minimizing the following objective function:

$$\begin{aligned} \min_{\mathbf{B}} \sum_{k=1}^{K} \sum_{j=1}^{J} & \frac{1}{2} \| \mathbf{S}_{:jk} \circledast (\mathbf{Y}_{:jk} - \mathbf{X}_{::k} \mathbf{B}_{:jk}) \|_2^2 + \frac{1}{2} \| \mathbf{B} \|_F^2 \\ & + \| \mathbf{B} \|_{F,1} + \frac{1}{2} \| \mathbf{B} \|_{\text{ts}} \end{aligned} \tag{10}$$

where , and are the positive parameters controlling the importance of the overfitting term, inter-task correlation terms and intra-task time smoothness term, respectively.

3.3 Optimization

The optimization problem in Eq. (10) incorporates four unique constraints into our problem. In this section, we apply the alternating direction method of multipliers (ADMM) method [8] and the proximal methods to optimize it since the solution to this problem is not trivial. First, we introduce a sequence of new constraints $\mathbf{M}_{:jk} = \mathbf{X}_{::k}\mathbf{B}_{:jk}$ for $(k = 1; \quad ; K;\ j = 1; \quad ; J)$ and update the problem in Eq. (10) as follows:

$$\begin{aligned}\min_{\mathbf{B}} & \sum_{k=1}^{K}\sum_{j=1}^{J} \frac{1}{2}\|\mathbf{S}_{:jk} \quad (\mathbf{Y}_{:jk} \quad \mathbf{M}_{:jk})\|_2^2 \\ & + \frac{1}{2} \ \|\mathbf{B}\|_F^2 + \ \|\mathbf{B}\|_{F;1} + \frac{1}{2} \ \|\mathbf{B}\|_{\text{ts}} \\ \text{s.t. } & \mathbf{M}_{:jk} = \mathbf{X}_{::k}\mathbf{B}_{:jk}; (k = 1; \quad ; K;\ j = 1; \quad ; J)\end{aligned} \tag{11}$$

By combining the linear and quadratic terms in the augmented Lagrangian and scaling the dual variable, the objective function can be written as:

$$\begin{aligned}\min_{\mathbf{B}} & \sum_{k=1}^{K}\sum_{j=1}^{J} \frac{1}{2}\|\mathbf{S}_{:jk} \quad (\mathbf{Y}_{:jk} \quad \mathbf{M}_{:jk})\|_2^2 \ + \frac{1}{2} \ \|\mathbf{B}\|_F^2 \\ & + \sum_{k=1}^{K}\sum_{j=1}^{J} \frac{1}{2} \ \|\mathbf{M}_{:jk} \quad \mathbf{X}_{::k}\mathbf{B}_{:jk} + \mathbf{u}_{:jk}\|_2^2 \quad \frac{1}{2} \ \|\mathbf{u}_{:jk}\|_2^2 \\ & + \ \|\mathbf{B}\|_{F;1} + \frac{1}{2} \ \|\mathbf{B}\|_{\text{ts}}\end{aligned} \tag{12}$$

where $\mathbf{u}$ is the scaled variable tensor and $\quad > 0$ is the penalty parameter. Then we can have the scaled form of ADMM as follows:

$$\begin{aligned}\mathbf{M}^{n+1} := \arg\min_{\mathbf{M}} & \sum_{k=1}^{K}\sum_{j=1}^{J} \frac{1}{2}\|\mathbf{S}_{:jk} \quad (\mathbf{Y}_{:jk} \quad \mathbf{M}_{:jk})\|_2^2 \\ & + \frac{1}{2} \ \|\mathbf{M}_{:jk} \quad \mathbf{X}_{::k}\mathbf{B}^n_{:jk} + \mathbf{u}^n_{:jk}\|_2^2\end{aligned} \tag{13}$$

$$\begin{aligned}\mathbf{B}^{n+1} := \arg\min_{\mathbf{B}} & \sum_{k=1}^{K}\sum_{j=1}^{J} \frac{1}{2} \ \|\mathbf{M}^{n+1}_{:jk} \quad \mathbf{X}_{::k}\mathbf{B}_{:jk} + \mathbf{u}^n_{:jk}\|_2^2 \\ & + \frac{1}{2} \ \|\mathbf{B}_{:jk}\|_2^2 + \frac{1}{2} \ (1 \quad (j = J))\|j\mathbf{B}_{:(j+1)k} \quad \mathbf{B}_{:jk}\|_2^2 \\ & + \ \|\mathbf{B}\|_{F;1}\end{aligned} \tag{14}$$

$$\mathbf{u}^{n+1}_{::k} := \mathbf{u}^n_{::k} + \mathbf{M}^{n+1}_{::k} \quad \mathbf{X}_{::k}\mathbf{B}^{n+1}_{::k} \tag{15}$$

Thus, next we need to optimize the two problems given in Eq. (13) and Eq. (14) in order to solve the problem in Eq. (11). Here, the symbol n in the superscript represent the number of iterations.

Step 1: Update $\mathbf{M}^{n+1}$ given $\mathbf{B}^n$ and $\mathbf{u}^n$ (solve Eq. (13)).

The first order and second order derivative of the problem in Eq. (13) with respect to $\mathbf{M}_{ijk}$ calculated in Eq. (16) and Eq. (17) indicates that the objective function in Eq. (13) is strictly convex.

$$\frac{@L}{@\mathbf{M}_{ijk}} = \ \mathbf{S}_{ijk}(\mathbf{Y}_{ijk} \quad \mathbf{M}_{ijk}) + \ (\mathbf{M}_{ijk} \quad \mathbf{X}_{i:k}\mathbf{B}^n_{:jk} + \mathbf{u}^n_{ijk}) \tag{16}$$

$$\frac{@^2L}{(@\mathbf{M}_{ijk})^2} = \mathbf{S}_{ijk} + \quad > 0 \tag{17}$$

Algorithm 1: FISTA for $\grave{}_{F;1}$ Constrained Optimization.

Input: Predictor tensor ($\mathbf{X}$);
Proximal survival status tensor ($\mathbf{M}$);
Scaled variable tensor ($\mathbf{u}$);
Output: Regression coefficient tensor ($\mathbf{B}$)

Initialize: $\mathbf{B}^0$ random real numbers, $\mathbf{B}^{\ 1} = \mathbf{B}^0$;
  $m = 1, \ ^0 = 1, \ ^1 = 0, \ ^0 = 1$;
repeat
  $\mathbf{A}^m = \mathbf{B}^{m\ 1} + \frac{m-2}{m-1}1(\mathbf{B}^{m\ 1} \quad \mathbf{B}^{m\ 2})$;
  repeat
    Calculate $\mathbf{B}^m = \ _Z(\mathbf{A}^m \quad \frac{1}{m}g'(\mathbf{A}^m))$;
    Calculate $\mathbf{B}^m = \mathbf{B}^m \quad \mathbf{A}^m$;
    Calculate $h(\mathbf{A}^m;\mathbf{B}^m) = g(\mathbf{A}^m) + \sum_{k;j}^{P} \frac{@g(\mathbf{B})}{@\mathbf{B}_{:jk}}^T \mathbf{B}^m_{:jk} + \frac{m}{2}\|\mathbf{B}^m\|_F^2$;
    if $g(B^m) \quad h(A^m;B^m)$ **then**
      break ;
    end
    Set $^m = 2^{\ m\ 1}$;
  until *Convergence*;
   $^m = \frac{1+\sqrt{1+4\ ^{2}_{m-1}}}{2}$;
  $m = m + 1$;
until *Convergence*;
$\mathbf{B} = \mathbf{B}^m$.

Thus, $\mathbf{M}_{::k}$ can be updated by Eq. (18) in each iteration for each task k.

$$\mathbf{M}^{n+1}_{::k} \quad \frac{\mathbf{S}_{::k} \quad \mathbf{Y}_{::k} + \ (\mathbf{X}_{::k}\mathbf{B}^n_{::k} \quad \mathbf{u}^n_{::k})}{\mathbf{S}_{::k} +} \tag{18}$$

Step 2: Update $\mathbf{B}^{n+1}$ given $\mathbf{M}^{n+1}$ and $\mathbf{u}^n$ (solve Eq. (14)).

The objective function in Eq. (14) can be separated into two terms, a smooth term and a non-smooth term, as shown in Eq. (19).

$$L(\mathbf{B}) = g(\mathbf{B}) + \ \|\mathbf{B}\|_{F;1} \tag{19}$$

where $g(\mathbf{B})$ is a smooth convex function defined as:

$$\begin{aligned}g(\mathbf{B}) := & \sum_{k=1}^{K}\sum_{j=1}^{J} \frac{1}{2} \ \|\mathbf{M}_{:jk} \quad \mathbf{X}_{::k}\mathbf{B}_{:jk} + \mathbf{u}_{:jk}\|_2^2 \\ & + \frac{1}{2} \ \|\mathbf{B}_{:jk}\|_2^2 + \frac{1}{2} \ (1 \quad (j = J))\|j\mathbf{B}_{:(j+1)k} \quad \mathbf{B}_{:jk}\|_2^2\end{aligned} \tag{20}$$

and the first order derivative of $g(\mathbf{B})$ can be obtained as:

$$\begin{aligned}\frac{@g(\mathbf{B})}{@\mathbf{B}_{:jk}} := & \ \mathbf{X}^T_{::k}(\mathbf{M}_{:jk} \quad \mathbf{X}_{::k}\mathbf{B}_{:jk} + \mathbf{u}_{:jk}) \\ & + \ \mathbf{B}_{:jk} + \ (1 \quad (j = J))(\mathbf{B}_{:jk} \quad \mathbf{B}_{:(j+1)k}) \\ & + \ (1 \quad (j = 1))(\mathbf{B}_{:jk} \quad \mathbf{B}_{:(j \ 1)k})\end{aligned} \tag{21}$$

Then, the optimization problem for Eq. (14) is equivalent to the following $\grave{}_{F;1}$-ball constrained smooth convex optimization problem.

$$\min_{\mathbf{B} 2 Z} \quad g(\mathbf{B}) \tag{22}$$

where $Z = f\mathbf{B} \ 2 \ \mathbb{R}^{P \ J \ K} \ jz \quad \|\mathbf{B}\|_{F;1} g$ and $z \ 2 \ Z_+$ is the radius of the $\grave{}_{F;1}$-ball. The minimization problem in Eq. (22)

can be solved by a proximal gradient decent algorithm with backtracking linear search [9], which is based on the fast iterative shrinkage-thresholding algorithm (FISTA) framework provided in Algorithm 1. Within the iteration m, we set the initial searching point as $\mathbf{A}^m = \mathbf{B}^{m-1} + {}^m(\mathbf{B}^{m-1} - \mathbf{B}^{m-2})$, where ^{m} is an non-negative parameter. Then, the approximated solution $\mathbf{B}^m$ can be obtained as:

$$\mathbf{B}^m = {}_Z(f(\mathbf{A}^m)) \tag{23}$$

where $f(\mathbf{A}^m) = \mathbf{A}^m - {}^1g'(\mathbf{A}^m)$ with parameter , and g' denotes the derivative (See line 6 of Algorithm 1). The Euclidean projection operator [34] ${}_Z$ is defined as:

$$\begin{aligned} {}_Z(f(\mathbf{A})) = &\arg\min_{\mathbf{B}} \|\mathbf{B} - f(\mathbf{A})\|_F^2 \\ &\text{s.t. } \|\mathbf{B}\|_{F;1} \quad Z \end{aligned} \tag{24}$$

To solve this $\ell_{F;1}$ constrained optimization problem, we adopt the method introduced in [34], which is originally proposed for $\ell_{2;1}$ constrained optimization problem. In this paper, we extend this method to the higher dimension where $\mathbf{B}$ is a tensor.

Theorem 1. *For a given , the primal optimal point $\mathbf{B}$ is given by*

$$\mathbf{B}_{p::} = \begin{cases} \frac{\|f(\mathbf{A}_{p::})\|_F - }{\|f(\mathbf{A}_{p::})\|_F} f(\mathbf{A}_{p::}) & > 0; \|f(\mathbf{A}_{p::})\|_F > \\ 0 & > 0; \|f(\mathbf{A}_{p::})\|_F \\ f(\mathbf{A}_{p::}) & = 0 \end{cases} \tag{25}$$

Proof. To prove this theorem, we first convert $\mathbf{B}$ to a giant matrix $\underline{B} \in R_+^{P \times Q}$, where $Q = JK$. Each element of $\underline{B}$ is given by $\underline{B}_{pq} = \mathbf{B}_{pjk}$, where $q = k \quad J + j$. In the same way, we can create the giant matrices $\underline{M}$; $\underline{u}$ and $\underline{A}$ for $\mathbf{M}$; $\mathbf{u}$ and $\mathbf{A}$, respectively. It is easy to prove that the value of $g(\mathbf{B})$ will remain the same by using the giant matrix representation, and there is a giant matrix $\frac{\partial g}{\partial \underline{B}_{:q}}$ corresponding to $\frac{\partial g(\mathbf{B})}{\partial \mathbf{B}_{:jk}}$, since they are both vectors. Therefore, the problem can be solved by the following Euclidean projection

$$\begin{aligned} \underline{B} = &\arg\min_{\underline{B}} \|\underline{B} - (\underline{A} - \frac{1}{}\frac{\partial g}{\partial \underline{B}})\|_F^2 \\ &\text{s.t. } \|\underline{B}\|_{2;1} \quad Z \end{aligned} \tag{26}$$

where $\|\underline{B}\|_{2;1} = \sum_{p=1}^{P} \|\underline{B}\|_2$. It is obvious that $\|\underline{B}\|_{2;1} = \|\mathbf{B}\|_{F;1}$. Finally, comparing Eq. (24) with Eq. (26), we can conclude that Theorem 1 based on the giant matrix representation is equivalent to Theorem 5 in [34]. □

The details of the proposed model is shown in Algorithm 2. In the algorithm, $\mathbf{M}$ is initialized to be the survival status tensor $\mathbf{Y}$ and $\mathbf{B}$ is randomly initialized. Within each iteration, $\mathbf{M}$ and $\mathbf{u}$ are updated in line 3-5 and line 7-9, respectively. $\mathbf{B}$ is updated in line 6 using Algorithm 1. Algorithm 1 is based on the framework of FISTA algorithm, where the Euclidean projection is conducted in line 6.

3.4 Complexity Analysis

For the ADMM algorithm, the time complexity in each iteration is determined by updating $\mathbf{M}$ and $\mathbf{B}$. Firstly, it takes $O(NPJ)$ float point operations to calculate $\mathbf{M}$ using Eq. (18), where $N = max_k(N_k)$. When updating $\mathbf{B}$, within each iteration, the time complexity for evaluating the values of $g(\mathbf{B})$ and its derivative is $O(NPJK)$, and we need to perform $O(PJK)$ operations on the Euclidean projections, i.e., Eq. (25). Thus, the time complexity for updating $\mathbf{B}$ is $O(\frac{1}{}(NPJK + PJK)) = O(\frac{1}{}NPJK)$, where is the desired accuracy. Therefore, the overall time complexity is $O(\frac{1}{}NPJK)$. Since our proposed model trains all the tasks in a single run, it is more convincing to evaluate the average time complexity for each task, which is $O(\frac{1}{}NPJ)$.

Algorithm 2: MTMT Algorithm

Input: Predictor tensor ($\mathbf{X}$); Survival status tensor ($\mathbf{Y}$); Indicator tensor ($\mathbf{S}$); regularization parameters (; ; ;);

Output: Regression coefficient tensor ($\hat{\mathbf{B}}$);

1 Initialize $n = 0$; $\mathbf{M}^0 = \mathbf{Y}$; $\mathbf{u}^0 = \mathbf{0}$, and randomly initialize $\mathbf{B}^0$;
2 **repeat**
3 **for** $k = 1; K$ **do**
4 Compute $\mathbf{M}_{::k}^{n+1}$ using Eq. (18);
5 **end**
6 Compute $\mathbf{B}$ using Algorithm 1;
7 **for** $k = 1; K$ **do**
8 Compute $\mathbf{u}_{::k}^{n+1} = \mathbf{u}_{::k}^{n} + \mathbf{M}_{::k}^{n+1} \quad \mathbf{X}_{::k}\mathbf{B}_{::k}^{n+1}$;
9 **end**
10 $n = n + 1$;
11 **until** *Convergence*;
12 $\hat{\mathbf{B}} = \mathbf{B}^n$.

4 Experimental Results

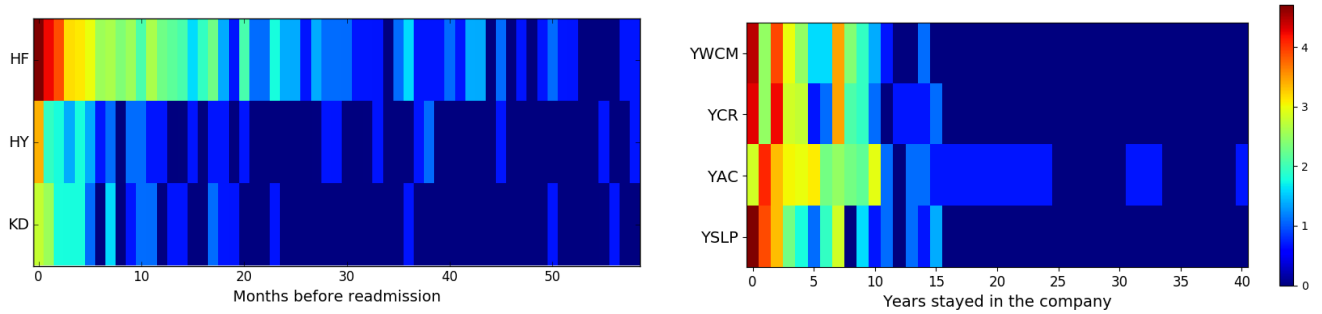
In this section, we first introduce the two real-world datasets, Medical Information Mart for Intensive Care III (MIMIC III) [10], [11] and Employee attrition dataset obtained from IBM Waston Analytics, used for the model evaluation and then conduct various experiments on the datasets. We will compare the performance of the proposed MTMT model with the state-of-the-art baseline methods. In addition, we also analyze the shared features selected by the proposed method and provide the parameter sensitivities.

4.1 Datasets Description

Two real-world datasets will be used in our paper.

MIMIC III Dataset: The Medical Information Mart for Intensive Care III (MIMIC III) dataset used in this paper is a public dataset provided by PhysioNet[1]. It includes de-identified health-related dataset with more than 40,000 ICU patients from the Beth Isreal Deaconess Medical Center between 2001 and 2012. The information available in this dataset includes: demographics, diagnosis, laboratory test results, procedures and medications, etc. In addition, the dataset shows a good temporal resolution, which is a helpful property for researchers to find the latent relationships of various diseases. In the experiments, we analyze the readmission problems for three diseases, including Heart Failure (HF), Hypertensive (HY) and Kidney Failure (KD). A correlated pattern between the three tasks

1. https://mimic.physionet.org/



(a) MIMIC III (b) Employee Attrition

Fig. 2: Distribution of events over time on both datasets.

can be observed in Figure 2(a) through the distribution of patient readmission (shown as the logarithm of the number of patients) over time provided.

Employee attrition dataset: The employee attrition dataset provided by IBM Waston Analytics[2] is also used to evaluate the performance of the proposed algorithm. The dataset spans 40 years and consists of 1470 employees attrition information. The features provided in the dataset can be categorized into four types: demographical, education related, job related and salary related information. Based on the attrition status and the time information provided, we formulate four related problems and generate four longitudinal datasets. (i) *Years At Company (YAC)*: How many years have the employee stayed at the company before leaving? (ii) *Years With Current Manager (YWCM)*: How many years have the employee worked with the current manager before leaving? (iii) *Years in Current Role (YCR)*: How many years have the employee worked in the current role? (iv) *Years Since Last Promotion (YSLP)*: How many years have the employee stayed in the company since the last promotion? The distribution of employee attrition (shown as the logarithm of the number of employees) over time provided in Figure 2(b) indicates a correlated pattern between the four tasks, especially in the first 15 years.

There are mainly three steps in data preparation for the two survival problems. (i) We extracted features from the original dataset after generating dummy variables for the categorical features and removing the features with zero-variance. (ii) The longitudinal dataset for each survival task was obtained as the triplet $(X; T; Y)$ by collecting the features, event status and the corresponding event time for all subjects. (iii) The entire dataset of each problem represented by a tensor is obtained by combining the data in individual tasks.

More details about the datasets are provided in Table 1. In the table, "# instances", "# features" and "# censoring" indicate the number of instances, features, and censored instances, respectively. For the Employee Attrition problem, the event of interest in the formulated survival problem is employee attrition. Therefore, the censored instances correspond to the employees who are still working in the company at the observation time, while the uncensoring indicates that the employee has left the company during the observation time period. For the MIMIC III dataset, the event of interest is the patient readmission and the censored instances are the patients who are not readmitted to the hospital during the observation period. We implemented the proposed MTMT algorithm in Python on the two datasets using 5-fold cross validation. Our codes are publicly available at https://github.com/wangpinggl/MTMT.

TABLE 1: Details of the dataset used in our experiments.

Dataset		# instances	# features	# censored
MIMIC III	HF	1811	118	1333
	HY	731	118	657
	KD	1209	118	1143
Employee Attrition	YWCM	1207	46	1055
	YCR	1226	46	1062
	YAC	1426	46	1205
	YSLP	889	46	762

4.2 Comparison Methods

To evaluate the performance of the proposed algorithm, we compare it with the following commonly used survival analysis methods.

Cox-based methods: The Cox-based models [14] assume that all the instances share the same baseline hazard function. The Lasso-Cox and the EN-Cox method are developed for the high-dimensional survival problems. DeepSurv [22] is a Cox proportional hazards deep neural network method for modelling the dependencies between covariates and events. In our experiments, Cox model and the regularized Cox models are trained using the *survival* [35] and *fastcox* [36] package in R, respectively. DeepSurv is implemented with Keras[3]. The batch size and epoch are set to be 100 and 30, respectively. The RMSprop optimizer with hyper-parameter = 0:9, = 10 ^{8} and learning rate of 10 ^{5} is adopted to train the model parameters. The dimension of the hidden states is set to be 32.

Parametric survival methods: The survival time or the logarithm of the survival time is assumed to follow a theoretical distribution in parametric methods. In the experiments, the parametric survival methods learned through *survival* package with Exponential and Weibull distributions are compared with the proposed method.

Linear methods: We also compare the proposed method with the standard linear regression optimized using ordinary least square (OLS) method since the multi-task

2. https://www.ibm.com/communities/analytics/watson-analytics-blog/hr-employee-attrition/

3. https://github.com/KITMILTU/DeepSurv-Keras

TABLE 2: Performance evaluation using time-dependent AUC (along with their standard deviations) on Employee Attrition dataset.

Methods		YWCM	YCR	YAC	YSLP	Average
Cox-based methods	Cox	0.8473 (0.0022)	0.8371 (0.0020)	0.8741 (0.0013)	0.7820 (0.0031)	0.8351 (0.0015)
	Lasso-Cox	0.8478 (0.0011)	0.8418 (0.0004)	0.8696 (0.0015)	0.7841 (0.0065)	0.8358 (0.0013)
	EN-Cox	0.8474 (0.0018)	0.8441 (0.0003)	0.8691 (0.0008)	0.7961 (0.0067)	0.8392 (0.0009)
	DeepSurv	0.5335(0.0597)	0.4964 (0.0546)	0.5591 (0.0522)	0.5508 (0.0513)	0.5350(0.0278)
Parametric methods	Exponential	0.8325 (0.0026)	0.8275 (0.0021)	0.8616 (0.0012)	0.7753 (0.0039)	0.8242 (0.0013)
	Weibull	0.8543 (0.0020)	0.8411 (0.0021)	0.8698 (0.0011)	0.7799 (0.0038)	0.8363 (0.0016)
Linear models	OLS	0.7329 (0.0007)	0.7584 (0.0016)	0.8202 (0.0010)	0.6623 (0.0006)	0.7435 (0.0043)
	Tobit	0.8463 (0.0022)	0.8410 (0.0022)	0.8727 (0.0010)	0.7722 (0.0034)	0.8331 (0.0018)
	BJ	0.8494 (0.0000)	0.8312 (0.0006)	0.8636 (0.0033)	0.7760 (0.0045)	0.8301 (0.0015)
Multi-task linear models	Multi-Lasso	0.7061 (0.0011)	0.7557 (0.0010)	0.7900 (0.0017)	0.6894 (0.0052)	0.7353 (0.0021)
	Multi-$\ell_{2,1}$	0.7941 (0.0025)	0.8128 (0.0003)	0.8326 (0.0012)	0.7751 (0.0107)	0.8037 (0.0006)
	MTLSA	0.8176 (0.0007)	0.8315 (0.0003)	0.8524 (0.0019)	0.7923 (0.0037)	0.8235 (0.0007)
	MTLSA.V2	0.8327 (0.0011)	0.8432 (0.0006)	0.8649 (0.0022)	0.7972 (0.0042)	0.8345 (0.0008)
	MTMT	**0.8628 (0.0045)**	**0.8455 (0.0009)**	**0.8750 (0.0009)**	**0.8151 (0.0010)**	**0.8496 (0.0005)**

framework in this paper is based on linear regression method. It should be noted that the standard linear regression method is trained only using the subjects who experienced events since it cannot handle the censored data in longitudinal studies. Tobit model and Buckley-James (BJ) regression method are two linear regression methods adapted to solve survival problems and trained in *survival* and *bujar* package in R, respectively.

Multi-task learning methods: We also compare with the standard multi-task learning method regularized with lasso and $\ell_{2,1}$ implemented in the MALSAR package [37] written in MATLAB. The multi-task learning model for survival analysis (MTLSA) in [1] and its variant MTLSA.V2 which formulate a sequence of standard survival analysis problems over time for the original problem with single event of interest are implemented using the MTLSA package[4].

4.3 Performance Evaluation

A common approach to evaluate survival analysis methods is to consider the relative risk between two comparable instances instead of the absolute survival times for each instance using the time-dependent AUC [38], which is also known as concordance index (C-index) for survival analysis problems. Two instances are considered as comparable if their survival times fall in one of the following: (i) both of them are uncensored; (ii) the observed event time of the uncensored instance is smaller than the censoring time of the censored instance [39]. Two comparable instances are considered to be concordant if the predicted probability of event for the instance which has event earlier is higher than that of the other instance.

Consider two instances i and j, we can use an indicator function to represent the comparability of this pair of instances as $C_{ij} = I(Y_i = 1 \, and \, T_i < T_j)$, which indicates that $C_{ij} = 1$, if the two instances are comparable and $C_{ij} = 0$, otherwise. Thus, the total number of comparable pairs can be obtained as

$$N_C = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} C_{ij} \tag{27}$$

where N represents the number of instances in the testing dataset. We can also represent the concordance of the two instances i and j similarly using the indicator function as $c_{ij} = I(s_i < s_j \, and \, C_{ij} = 1)$, where s_i and s_j represent the estimated survival probabilities for instances i and j, respectively. This indicates that $c_{ij} = 1$, if instances i and j are concordant and $c_{ij} = 0$, otherwise. Thus, the total number of concordant pairs can be calculated as

$$N_c = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} c_{ij} \tag{28}$$

Therefore, the time-dependent AUC can be calculated as:

$$AUC_t = N_c / N_C \tag{29}$$

Table 2 and Table 3 show the performance results of different algorithms using time-dependent AUC on the Employee Attrition and MIMIC III datasets, respectively. The results in bold represent the best performance. It can be observed that the proposed MTMT algorithm outperforms other state-of-the-art methods for all tasks on Employee Attrition dataset and two tasks on MIMIC III dataset. For other tasks, MTMT also achieves similar time-dependent AUC value compared to the best performance. The last column in Table 2 and Table 3 is the averaged time-dependent AUC across all the tasks for each algorithm. The results show that the MTMT algorithm outperforms other baseline methods on average for each task on the two problems. These promising results indicate that the proposed multi-task framework can increase the prediction performance of each task on average.

4. https://github.com/MLSurvival/MTLSA

TABLE 3: Performance evaluation using time-dependent AUC (along with their standard deviations) on MIMIC III dataset.

Methods		HF	HY	KD	Average
Cox-based methods	Cox	0.5499 (0.0010)	0.6809 (0.0127)	0.5112 (0.0052)	0.5807 (0.0079)
	Lasso-Cox	0.5534 (0.0004)	0.6764 (0.0110)	0.5395 (0.0081)	0.5898 (0.0057)
	EN-Cox	0.5552 (0.0007)	0.6783 (0.0127)	0.4897 (0.0048)	0.5744 (0.0092)
	DeepSurv	0.5102 (0.0240)	0.5209 (0.0311)	**0.6018 (0.1317)**	0.5464 (0.0537)
Parametric methods	Exponential	0.5567 (0.0011)	0.6166 (0.0191)	0.5688 (0.0079)	0.5807 (0.0010)
	Weibull	0.5479 (0.0008)	0.5703 (0.0177)	0.5419 (0.0351)	0.5534 (0.0002)
Linear models	OLS	0.5354 (0.0007)	0.4671 (0.0286)	0.3181 (0.0366)	0.4402 (0.0123)
	Tobit	0.5578 (0.0011)	0.5946 (0.0029)	0.5531 (0.0198)	0.5685 (0.0005)
	BJ	0.5571 (0.0006)	0.6862 (0.0115)	0.2452 (0.0237)	0.4962 (0.0514)
Multi-task models	Multi-Lasso	0.5220 (0.0015)	0.6803 (0.0126)	0.5268 (0.0186)	0.5764 (0.0081)
	Multi-$\ell_{2,1}$	0.5412 (0.0012)	0.6859 (0.0098)	0.5342 (0.0146)	0.5871 (0.0073)
	MTLSA	0.5348 (0.0010)	0.6528 (0.0096)	0.5587 (0.0097)	0.5821 (0.0039)
	MTLSA.V2	0.5404 (0.0010)	0.6630 (0.0116)	0.5633 (0.1228)	0.5889 (0.0042)
	MTMT	**0.5657 (0.0166)**	**0.6868 (0.0007)**	0.5674 (0.0178)	**0.6066 (0.0048)**

TABLE 4: The top-10 common features selected across four tasks on the Employee Attrition dataset.

Feature Name	Score
Total Working Years	2.2011
Job Level	0.8613
Performance Rating	0.7617
Job Involvement	0.4067
Number of Companies Worked	0.4012
Over Time	0.3537
Work Life Balance	0.3077
Job Role: Sales Executive	0.2938
Environment Satisfaction	0.2591
Job Role: Sales Representative	0.2508

TABLE 5: The top-10 common features selected across three tasks on MIMIC III dataset.

Feature Name	Score
Support Systems	11.1720
Non Invasive Blood Pressure Systolic	11.0395
Braden Mobility	11.0392
Urine Appearance	10.9794
Position	10.9227
Creatinine	10.8996
Dorsal PedPulse R	10.8814
Marital status	10.8604
Pain Present	10.8504
Blood Urea Nitrogen	10.8452

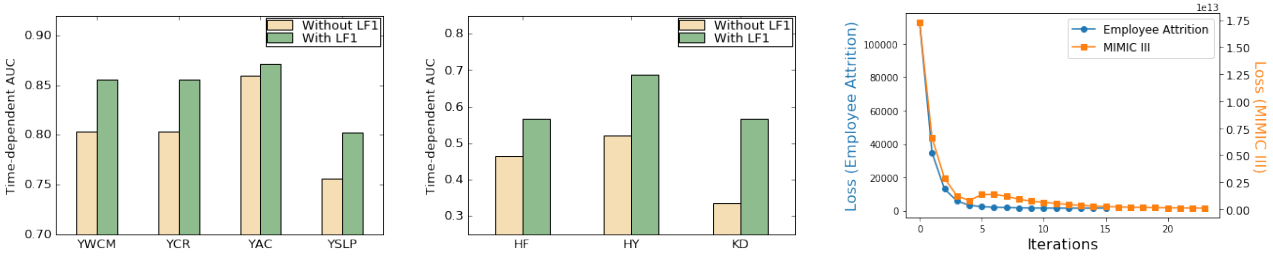
4.4 Feature Selection

One of the main goals of multi-task learning methods is to learn the shared features across all the tasks. In this section, we evaluate the effect of the $\ell_{F,1}$ norm on the tensor, which is mainly used to perform the common feature selection across all tasks over time. The bar plots in Figure 3(a) and Figure 3(b) show the effect of the $\ell_{F,1}$ norm on the prediction performance using time-dependent AUC. We can observe that the MTMT model regularized by $\ell_{F,1}$ norm (in green) outperforms the model without $\ell_{F,1}$ norm (in yellow) on all the tasks. This indicates that the new proposed $\ell_{F,1}$ norm can perform effectively for feature selection.

Table 4 and Table 5 provide the top-10 common features selected across different tasks on the Employee Attrition dataset and MIMIC III dataset, respectively, based on the averaged $\ell_{F,1}$ norm over 5-fold of the tensor slices $\mathbf{B}_{i::}$ (as given in Section 3) for each feature. The higher score indicates the greater impact on the final prediction. According to the results provided in Table 4, we can observe that the feature *Total Working Years* has the highest impact on the employee attrition. In addition, all these selected features are job related features, which means that these features mainly dominate the employee attrition. It is worth noting that two dummy features about "Job Role" among the selected features belong to the sales department. It indicates that the turnover rate in the sales department is relatively high compared to other departments. The selected common features among all the problems can provide an important guideline to reduce the attrition in a company. Based on the results in Table 5 on MIMIC III dataset, we can observe that the feature *Support Systems* affects the patient readmission most. It can also be observed that the variance of the scores for the top-10 features on MIMIC III dataset is relatively small, which indicates that these features affect the three tasks at the same level.

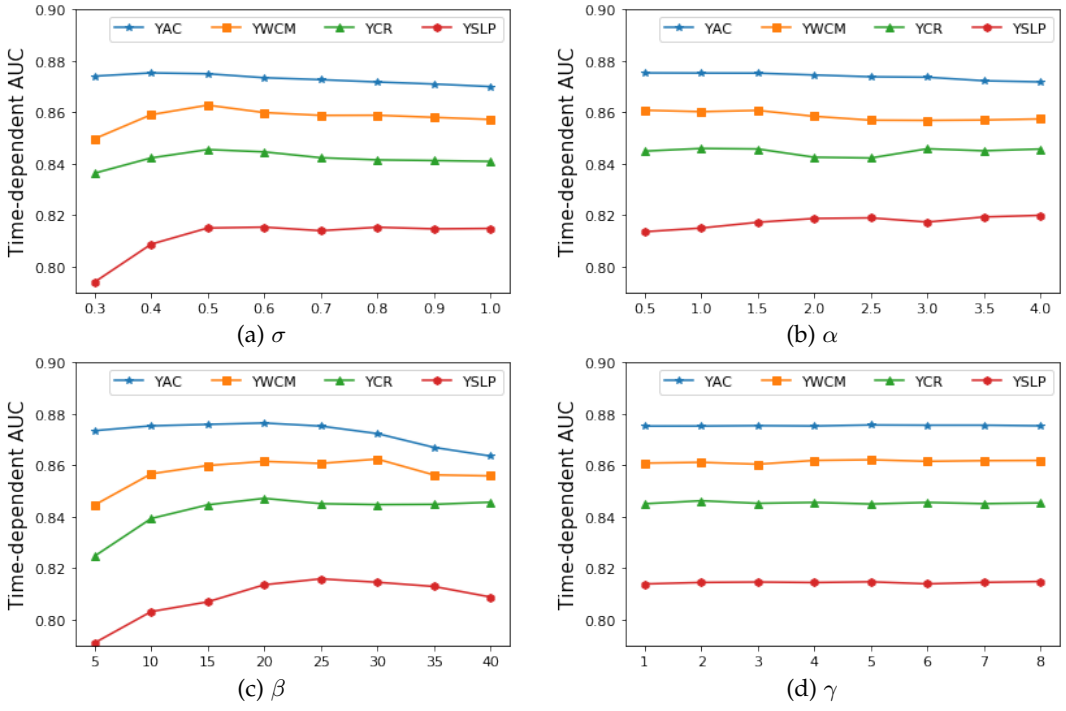
4.5 Parameter Sensitivity and Convergence Analysis

There are mainly four parameters in the proposed MTMT method, including (i) , which is used in the ADMM algorithm, (ii) the weight for the ℓ_F norm , (iii) the weight



(a) Employee Attrition (b) MIMIC III (c) Convergence

Fig. 3: Effect of `$_{F;1}$ norm on feature selection for both (a) employee attrition and (b) MIMIC III. (c) Convergence of MTMT model on both datasets.



(a) σ (b) α

(c) β (d) γ

Fig. 4: Parameter sensitivity of the four parameters used in the proposed method on Employee Attrition dataset.

of the inter-task correlation and (iv) the weight for the intra-task time smoothness . In this section, we provide the parameter sensitivity for each of these parameters in order to show the robustness of the proposed MTMT algorithm. Figure 4 shows the parameter sensitivity of the four parameters in their best performance range based on our experiments on the Employee Attrition dataset. It can be observed that each of the prediction performances are stable across all the four tasks when varying each parameter. The parameters used in our experiments are = 0:5, = 0:5, = 25 and = 2. Figure 3(c) shows the convergence of MTMT model on both datasets. The convergence threshold is set to 0.01. It took around 300 seconds for MTMT model to converge on both datasets.

5 Conclusions

Standard event prediction models are commonly used to make predictions for a single specific event at a given time point instead of predicting the occurrence of multiple events of interest simultaneously in a dynamic setting. To avoid sub-optimal solutions that are obtained by simply applying the standard survival analysis method independently to each task at specific time points, we formulated a temporal multi-task learning framework MTMT for survival analysis and optimized the problem using ADMM method. Our MTMT method demonstrates a superior performance on two real-world datasets compared to other state-of-the-art models. The qualitative results show that the common features selected by MTMT method can provide an important guideline for the real-world applications.

Acknowledgments

This work was supported in part by the US National Science Foundation grants IIS-1619028, IIS-1707498 and IIS-1838730.

References

[1] Y. Li, J. Wang, J. Ye, and C. K. Reddy, "A multi-task learning formulation for survival analysis," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016, pp. 1715–1724.

[2] C. K. Reddy and Y. Li, "A review of clinical prediction models," *Healthcare Data Analytics*, vol. 36, pp. 343–378, 2015.