A Multi-Task Learning Formulation for Survival Analysis

Yan Li[1], Jie Wang[2], Jieping Ye[2,3] and Chandan K. Reddy[1]

[1]Dept. of Computer Science, Wayne State University, Detroit, MI - 48202.
rock_liyan@wayne.edu, reddy@cs.wayne.edu

[2]Dept. of Computational Medicine and Bioinformatics, University of Michigan, MI -48109.

[3]Dept. of Electrical Engineering and Computer Science, University of Michigan, MI -48109.
{jwangumi, jpye}@umich.edu

ABSTRACT

Predicting the occurrence of a particular event of interest at future time points is the primary goal of survival analysis. The presence of incomplete observations due to time limitations or loss of data traces is known as *censoring* which brings unique challenges in this domain and differentiates survival analysis from other standard regression methods. The popularly used survival analysis methods such as Cox proportional hazard model and parametric survival regression suffer from some strict assumptions and hypotheses that are not realistic in most of the real-world applications. To overcome the weaknesses of these two types of methods, in this paper, we reformulate the survival analysis problem as a multi-task learning problem and propose a new multi-task learning based formulation to predict the survival time by estimating the survival status at each time interval during the study duration. We propose an indicator matrix to enable the multi-task learning algorithm to handle censored instances and incorporate some of the important characteristics of survival problems such as *non-negative non-increasing list* structure into our model through max-heap projection. We employ the $l_{2,1}$-norm penalty which enables the model to learn a shared representation across related tasks and hence select important features and alleviate over-fitting in high-dimensional feature spaces; thus, reducing the prediction error of each task. To efficiently handle the two non-smooth constraints, in this paper, we propose an optimization method which employs Alternating Direction Method of Multipliers (ADMM) algorithm to solve the proposed multi-task learning problem. We demonstrate the performance of the proposed method using real-world microarray gene expression high-dimensional benchmark datasets and show that our method outperforms state-of-the-art methods.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database applications-Data Mining; I.2.6 [**Artificial Intelligence**]: Learning; G.3 [**Mathematics of Computing**]: Survival analysis.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

KDD '16, August 13-17, 2016, San Francisco, CA, USA
© 2016 ACM. ISBN 978-1-4503-4232-2/16/08...$15.00
DOI: http://dx.doi.org/10.1145/2939672.2939857

Keywords

Survival analysis, Multi-task learning, regularization, high-dimensional data.

1. INTRODUCTION

Survival analysis aims at modeling time-to-event data, which is typically collected in longitudinal studies that start from a particular time and last until a certain *event of interest* has occurred [11, 20]. However, the event of interest may not always be observed during the study period due to time limitations or losing data traces. This phenomenon is more commonly known as *censoring* and makes survival analysis different from (and more challenging than) the standard regression methods. For the cases where the event of interest has been observed, the time to the event of interest is known as the *survival time*; while for the other instances, the last observed time is known as the *censored time* (we call these instances as censored instances). The most common form of censoring that occurs in real-world scenarios is *right censoring*[1], where the survival time of a censored instance is longer than or equal to the censored time, but its precise value is unknown.

In survival analysis, the Cox proportional hazards model and parametric censored regression models are important fundamental techniques for survival time prediction. These two methods (and their extensions) have been extensively studied in the fields of statistical learning and data mining. We will now describe the primary weaknesses of these two methods.

The Cox model and its extensions are built based on the *proportional hazards hypothesis*, i.e., it assumes that the hazard ratio between two instances is constant in time. This hypothesis indicates that the survival curves of all instances share a similar shape which is not realistic in many real-world applications. Also, the Cox model does not predict the survival time directly but rather models the hazard ratio. To predict the survival time, a *baseline hazard function* has to be estimated separately and this estimation will induce more prediction errors. When tied observations (survival times of multiple instances is exactly the same) occur during the study, Cox model has to use some approximation methods which suffer from either inducing bias (Breslow's approximation and Efron's approximation [6]) or bad scalability (Discrete method [24]).

[1]In this paper, we refer to the right censored data as censored data.

The parametric censored regression models suffer from even more critical weaknesses. The prediction performance of parametric censored regression is highly dependent on the choice of the distribution [12]. However, in real-world applications there are too many complex interactions and scenarios that can affect the event of interest in various ways; thus, in practice, choosing an appropriate theoretical distribution to approximate survival data is very difficult, if not impossible.

To overcome these weaknesses of both types of methods, in this paper, we propose the "**MTLSA**" model, which stands for "**M**ulti-**T**ask **L**earning model for **S**urvival **A**nalysis". We formulate the original survival time prediction problem into a multi-task learning problem. The primary motivation of using multi-task learning is because of its ability to learn a shared representation across related tasks and reduce the prediction error of each task. Thus, the model can provide a more accurate estimation of whether an event occurs or not at the beginning of each time interval which will thus provide an accurate estimation of the survival time for each instance. Another advantage of using multi-task learning for survival time estimation is because it translates the regression problem into a series of related binary classification problems, and at each time interval the corresponding classifier only focuses on modeling the local problem and hence provides a more accurate estimation than the regression models which aim at modeling the entire problem at once. Our model is built without any additional hypothesis except linear hypothesis, i.e., the feature and target exhibit a linear relationship, unlike the Cox proportional hazards model and parametric censored regression models.

As the survival status of a censored instance is unknown after the corresponding censored time, the target labeling matrix is not complete; therefore, the standard multi-task learning methods fail to handle the censored instances. To overcome this problem, we propose to use an additional indicator matrix which allows the model to simultaneously learn from both uncensored and censored instances (details can be found in Section 3.1) and hence the proposed model can simultaneously take advantage of both uncensored and censored instances. We notice that in survival analysis with non-recurring events, the survival status of instances naturally follows the *non-negative non-increasing list* structure, i.e., once the event occurs it will not occur again. Since the $l_{2,1}$-norm encourages multiple predictors to share similar sparsity patterns, it will not only select important features and alleviate over-fitting in high-dimensional feature spaces but will also learn a shared representation across all tasks at different time intervals. In MTLSA, we incorporate the non-negative non-increasing constraint and the $l_{2,1}$-norm with the loss function and propose an Alternating Direction Method of Multipliers (ADMM) algorithm [3] for coefficient estimation. In the proposed ADMM method, the *non-negative non-increasing list* constraint optimization is transformed into an Euclidean projection problem that can be learned efficiently.

In our empirical evaluation using various real-world gene expression cancer survival benchmark datasets, our model attains very competitive prediction performance and outperforms state-of-the-art methods in survival analysis. Additionally, we also demonstrate that our model outperforms most of the competing methods for the task of classifying whether or not a subject is alive at the beginning of each time interval in the observed study period.

The rest of the paper is organized as follows. In Section 2, related data mining approaches for survival analysis are discussed. In section 3, our proposed approach including the details of optimization procedure is explained. Section 4 demonstrates our experimental results on several real-world high-dimensional datasets while Section 5 concludes our work.

2. RELATED WORK

The prominent prediction methods in survival analysis can be categorized into three types: Cox-based, parametric censored regression, and linear models. In this section, we will briefly describe the most important works under each category and highlight the differences and relationships between our proposed model and existing works.

The Cox proportional hazards model [5] is one of the earliest and most widely used survival analysis methods which has garnered significant interest from researchers in both statistics and data mining communities. It is a semi-parametric model which does not make any assumption about the distribution of the survival outcomes and is usually learned by optimizing a partial likelihood function. To deal with high-dimensional data, some regularization methods have been proposed in the literature. These methods include LASSO-COX [25] which introduces the L_1 norm penalty in the log-partial likelihood loss function, Elastic-Net Cox (EN-COX) [22] which uses the elastic net penalty term and the kernel elastic net penalized Cox regression [30, 29] which modifies the elastic net penalty using a kernel matrix.

Parametric censored regression provides an important alternative to the Cox-based models. Parametric censored regression methods assume that the survival times (Case 1) or the logarithm of the survival times (Case 2) of all instances in the data follow a particular distribution [11]. The latter case, namely Case 2, is also termed as Accelerated failure time (AFT) models [33] because they assume that the covariate will "accelerate" or "decelerate" the time to the event of interest. Weibull distribution, logistic distribution, log-normal distribution, and log-logistic distribution [11] are the most commonly used distributions in parametric censored regression and the last two are considered to be the AFT models. In [14], the elastic net penalty has been employed to enable the parametric censored regression models to handle high-dimensional censored data.

Apart from the above two types of survival prediction methods, linear regression is another important branch of survival analysis. Strictly speaking, linear regression is a specific parametric censored regression; we consider linear censored regression models separately because linear regression is a fundamental method in data analysis. Because of censoring, the least-squares estimator cannot be directly used in survival analysis. The Tobit model [26] is the earliest attempt to extend the linear regression for data analysis with censored observations. Later, Buckley-James (BJ) estimator [4] was proposed to solve survival prediction with the combination of the Kaplan-Meier (K-M) estimator [10] (which is a non-parametric model). Recently, the elastic net penalty has been used within the BJ regression (EN-BJ) [31] and a weighted linear model [13] for efficiently handling the high-dimensional survival analysis problems.

In this paper, different from all the above mentioned methods, we propose a new approach which transforms the original survival analysis problem into a series of related binary classification problems, and then develop a multi-task learning method which explicitly models two important structural constraints of this problem, namely the *non-negative and non-increasing list*. In addition, the $l_{2,1}$-norm penalty is employed to enable the model learn a shared representation across related tasks and hence select important features in high-dimensional feature spaces. In [15], the authors proposed a multi-task logistic regression to predict the survival times of each instance by using a likelihood function that combines multiple local logistic regression models and incorporate the dependency between these models. This procedure is very hard to learn and fails to handle high-dimensional cases.

3. THE PROPOSED METHOD

In this section, we will first transform the original survival analysis problem into a multi-task learning problem by decomposing the regression component into related classification tasks. Then we will propose a new objective function that can solve the transformed problem and develop an ADMM based algorithm to optimize the objective function. We also provide a detailed algorithmic analysis in terms of convergence and complexity and then discuss a variant of the algorithm by relaxing certain constraints.

3.1 Transform to multi-task learning problem

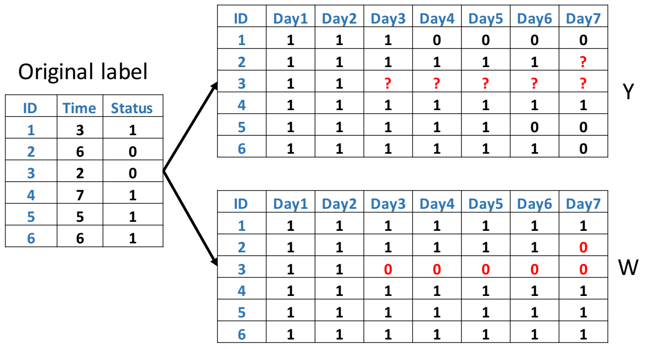
In survival analysis, for each data instance, we observe either a survival time (O_i) or a censored time (C_i), but not both. The dataset is right-censored if and only if $T_i = \min(O_i, C_i)$ can be observed during the study. An instance in survival data is usually represented by a triplet (X_i, T_i, δ_i), where X_i is a $1 \times q$ feature vector; δ_i is the censoring indicator, i.e., $\delta_i = 1$ for an uncensored instance, and $\delta_i = 0$ for a censored instance; and T_i denotes the *observed time* and is equal to the survival time O_i for uncensored instances and C_i otherwise, i.e.,

$$T_i = \begin{cases} O_i & \text{if} \quad \delta_i = 1 \\ C_i & \text{if} \quad \delta_i = 0 \end{cases} \tag{1}$$

For censored instances, O_i is a latent value, and the goal of survival analysis is to model the relationship between X_i and O_i by using the triplets (X_i, T_i, δ_i) for censored and uncensored instances.

In practice, time is considered as countable time intervals rather than a real number (a number with a fraction). We translate the original label into a k-column target matrix Y, where $k = \max(T_i), \forall i = 1, 2, \cdots, n$, is the maximum followup time of all the instances. Each element in the target matrix indicates whether the event occurred ("0") or not ("1"), and *the original survival prediction problem can thus be transformed into a multi-task learning problem. The primary motivation of transforming the survival analysis into a multi-task learning problem is that the dependency between the outcomes at various timepoints are accurately captured through a shared representation across related tasks in this multi-task transformation which will reduce the prediction error on each task.* Note that for censored instances we know that the event did not occur until the corresponding observed times, but we do not know whether the event occurs or not afterwards.

For now, we represent those unknown cells using "question marks" and later we will discuss the details of converting them into a more viable form. Figure 1 shows an example of generating a target matrix Y from the original labels. For the four uncensored instances (ID $\in 1, 4, 5, 6$), the cells of the corresponding rows in the target matrix Y are labeled as "1" until the observed time ($T_1 = 3, T_4 = 7, T_5 = 5, T_6 = 6$) and as "0" for the remaining cells; for the censored instances (ID $\in 2, 3$), the cells of the corresponding rows in the target matrix Y are labeled as "1" until the censored time ($T_2 = 6, T_3 = 2$) and as "?" for the remaining cells.



Original label

ID	Time	Status
1	3	1
2	6	0
3	2	0
4	7	1
5	5	1
6	6	1

Y

ID	Day1	Day2	Day3	Day4	Day5	Day6	Day7
1	1	1	1	0	0	0	0
2	1	1	1	1	1	1	?
3	1	1	?	?	?	?	?
4	1	1	1	1	1	1	1
5	1	1	1	1	1	0	0
6	1	1	1	1	1	1	0

W

ID	Day1	Day2	Day3	Day4	Day5	Day6	Day7
1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	0
3	1	1	0	0	0	0	0
4	1	1	1	1	1	1	1
5	1	1	1	1	1	1	1
6	1	1	1	1	1	1	1

Figure 1: Illustration of generating Y and W from the original label in a simple survival dataset.

In this paper, we only consider the non-recurring event scenario which means once the event occurs then it will not occur again. Hence, for a given row in the target matrix Y, once the label becomes "0" it will not change back to "1". Thus, we can see that each row of Y will have a *non-negative non-increasing list* structure:

$$P = \{Y \geq 0, Y_{ij} \geq Y_{il} | j \leq l, \forall j = 1, \cdots, k, \forall l = 1, \cdots, k\} \tag{2}$$

where $i = 1, 2, \cdots, n$.

An intuitive approach for solve this multi-task learning based formulation is as follows:

$$\underset{XB \in P}{\text{minimize}} \quad \frac{1}{2} \parallel Y - XB \parallel_F^2 + R(B) \tag{3}$$

where $B \in \mathbb{R}^{q \times k}$ is the estimated coefficient matrix, $\| \cdot \|_F$ denotes Frobenius norm, and $R(B)$ denotes the regularization term that prevents over-fitting and incorporates the additional constraints imposed by this problem. Note that Y is not a complete $n \times k$ target matrix (from the previous discussion). Now, we propose an indicator matrix W to handle the question marks in Y. W is an $n \times k$ binary matrix; we set $W_{ij} = 1$ if the exact labeling information of Y_{ij} is known, and $W_{ij} = 0$ if $Y_{ij} =$ "?". In Figure 1, we also show the corresponding W for the example survival data considered. The optimization problem in Eq.(3) is re-defined as:

$$\underset{XB \in P}{\text{minimize}} \quad \frac{1}{2} \parallel \Pi_W(Y - XB) \parallel_F^2 + R(B) \tag{4}$$

where

$$(\Pi_W(U))_{ij} = \begin{cases} U_{ij} & \text{if} \quad W_{ij} = 1 \\ 0 & \text{if} \quad W_{ij} = 0 \end{cases}$$

3.2 Objective function

Let us now briefly discuss two unique characteristics of the proposed model and then design suitable regularization terms to incorporate other additional constraints. The proposed model in Eq.(4) has two unique properties, *Non-negative non-increasing* and *Temporal smoothness*, which are desired to match the nature of the non-recurring events survival analysis.

- *Non-negative and non-increasing:* As discussed above, each row of the target matrix follows the non-negative non-increasing list structure. To preserve this characteristic, a corresponding structure constraint is added in Eq.(4) to ensure that the estimated output XB also follows the non-negative non-increasing list structure.
- *Temporal smoothness:* Since many works in the survival data deal with non-recurring events, i.e., for a certain row in the target matrix Y, once the label becomes "0" it cannot change back to "1" (or any other value), and this label change will occur at most once. Hence, in most cases, the adjacent labels of each instance are the same; thus, for all the N instances, the label vectors of adjacent tasks are similar. This is the temporal smoothness characteristic which will be modeled in the proposed multi-task learning formulation.

In this paper, the non-negative max-heap projection [17] is employed to ensure that XB follows the non-negative non-increasing list structure. This projection approximates each element of every selected set of target values by their corresponding mean values (refer to the Appendix for more details), and hence all elements in each selected set (a sublist in our case) share a same estimated target value; therefore, the non-negative max-heap projection also induces the temporal smoothness of XB. Apart from the two characteristics discussed above, our model should also alleviate over-fitting and induce sparsity in the estimated coefficients.

- *Sparsity:* The goal here is to learn a shared representation across all the tasks; thus, the model can select important common hidden features and reduce the prediction error of each task. The $l_{2,1}$-norm is chosen to be an additional regularization term for our model because it encourages multiple predictors to share similar sparsity patterns. Thus, the $l_{2,1}$-norm regularized regression model is able to select some common features across all the tasks [1]. In addition, such a sparsity inducing penalty will be able to help our model effectively handle high-dimensional datasets.
- *Overfitting:* A Frobenius norm regularization on the coefficient matrix B is introduced to alleviate the over-fitting problem for high-dimensional data.

Incorporating all of the above additional constraints in the form of regularizers into the proposed multi-task learning model, **MTLSA**, the following minimization problem is formulated:

$$\underset{XB\in P}{\text{minimize}} \quad \frac{1}{2} \parallel \Pi_W(Y-XB) \parallel_F^2 + \frac{\lambda_1}{2} \parallel B \parallel_F^2 + \lambda_2 \parallel B \parallel_{2,1} \tag{5}$$

where $\| \cdot \|_{2,1}$ denotes the $l_{2,1}$-norm, and $\lambda_1 \geq 0$ and $\lambda_2 \geq 0$ are two regularization parameters.

3.3 The proposed MTLSA algorithm

The solution of the optimization problem proposed in Eq.(5) is not trivial since it contains non-negative and non-increasing constraints along with the fact that the $l_{2,1}$-norm is a non-smooth penalty. We propose an ADMM based algorithm to solve the optimization problem proposed in Eq.(5). By introducing a new matrix $M = XB$, Eq.(5) can be rewritten in ADMM form as

$$\begin{aligned} &\underset{M\in P}{\text{minimize}} \quad \frac{1}{2} \parallel \Pi_W(Y-M) \parallel_F^2 + \frac{\lambda_1}{2} \parallel B \parallel_F^2 + \lambda_2 \parallel B \parallel_{2,1} \\ &\text{subject to} \quad M = XB \end{aligned} \tag{6}$$

Using the scaled dual variable μ and penalty parameter $\rho > 0$, the resulting augmented Largrangian of Eq.(6) is

$$\begin{aligned} L_\rho(M,B,\mu) = \frac{1}{2} \parallel \Pi_W(Y-M) \parallel_F^2 + \frac{\lambda_1}{2} \parallel B \parallel_F^2 \\ + \lambda_2 \parallel B \parallel_{2,1} + \frac{\rho}{2} \parallel M - XB + \mu \parallel_F^2 \end{aligned} \tag{7}$$

Thus, the scaled form of ADMM algorithm can be written as:

$$\begin{aligned} M^{t+1} := \underset{M\in P}{\arg\min} \Bigg(\frac{1}{2} \parallel \Pi_W(Y-M) \parallel_F^2 \\ + \frac{\rho}{2} \parallel M - XB^t + \mu^t \parallel_F^2 \Bigg) \end{aligned} \tag{8}$$

$$\begin{aligned} B^{t+1} := \underset{B\in \mathbb{R}^{q\times k}}{\arg\min} \Bigg(\frac{\lambda_1}{2} \parallel B \parallel_F^2 + \lambda_2 \parallel B \parallel_{2,1} \\ + \frac{\rho}{2} \parallel M^{t+1} - XB + \mu^t \parallel_F^2 \Bigg) \end{aligned} \tag{9}$$

$$\mu^{t+1} := \mu^t + M^{t+1} - XB^{t+1} \tag{10}$$

Now the main task of our model is to solve the optimization problems proposed in Eq.(8) and Eq.(9). Next we will present the algorithm for solving Eq.(8) and Eq.(9) in detail.

Step 1: Update M^{t+1} given B^t and μ^t (solve Eq.(8))

The updating of M^{t+1} is a constrained smooth convex optimization problem which can be expressed as a generalized form:

$$\min_{M\in P} g(M) \tag{11}$$

where $g(M) = \frac{1}{2} \parallel \Pi_W(Y-M) \parallel_F^2 + \frac{\rho}{2} \parallel M - XB^t + \mu^t \parallel_F^2$ is a smooth function and P is the *non-negative non-increasing list* structure defined in Eq.(2). Let $S = \mu^t - XB^t$. The objective function can be reformulated as:

$$\begin{aligned} &g(M) \\ =&\frac{1}{2} \parallel \Pi_W(Y-M) \parallel_F^2 + \frac{\rho}{2} \parallel M + S \parallel_F^2 \\ =&\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{k} W_{ij}(Y_{ij} - M_{ij})^2 + \frac{\rho}{2} \sum_{i=1}^{n} \sum_{j=1}^{k} (M_{ij} + S_{ij})^2 \\ =&\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{k} \left[(W_{ij}+\rho)M_{ij}^2 + 2(\rho S_{ij} - Y_{ij}W_{ij})M_{ij} + W_{ij}Y_{ij}^2 + \rho S_{ij}^2 \right] \\ =&\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{k} (W_{ij}+\rho) \Bigg[M_{ij}^2 + \frac{2(\rho S_{ij} - Y_{ij}W_{ij})M_{ij}}{W_{ij}+\rho} \\ &+ \left(\frac{\rho S_{ij} - Y_{ij}W_{ij}}{W_{ij}+\rho} \right)^2 - \left(\frac{\rho S_{ij} - Y_{ij}W_{ij}}{W_{ij}+\rho} \right)^2 + \frac{W_{ij}Y_{ij}^2 + \rho S_{ij}^2}{W_{ij}+\rho} \Bigg] \\ =&\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{k} (W_{ij}+\rho) \left[\left(M_{ij} - \frac{Y_{ij}W_{ij} - \rho S_{ij}}{W_{ij}+\rho} \right)^2 + Q_{ij} \right] \end{aligned}$$

where $Q_{ij} = \frac{W_{ij}Y_{ij}^2 + \rho S_{ij}^2}{W_{ij}+\rho} - \left(\frac{\rho S_{ij} - Y_{ij}W_{ij}}{W_{ij}+\rho} \right)^2$ does not depend on M_{ij}. Therefor, the optimization problem in Eq.(11)

equals to an Euclidean projection

$$M^{t+1} = \min_{M \in P} \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{k} \left(M_{ij} - \tilde{M}_{ij} \right)^2 \tag{12}$$

which projects $\tilde{M}_{ij} = \frac{Y_{ij}W_{ij} - \rho S_{ij}}{W_{ij} + \rho}$ onto the set P and thus ensures $M_{i1} \geq M_{i2} \geq \cdots \geq M_{ik} \geq 0$. The Euclidean projection in Eq.(12) is a special case of the non-negative max-heap projection which can be efficiently solved [17], and some details of the non-negative max-heap projection can be found in Appendix.

Step 2: Update B^{t+1} given M^{t+1} and μ^t (solve Eq.(9))

The updating of B^{t+1} in Eq.(9) can be considered as a standard $l_{2,1}$-norm regularization problem:

$$\operatorname*{arg\,min}_{B \in \mathbb{R}^{q \times k}} \left(\frac{1}{2} \parallel \mathscr{L} - XB \parallel_F^2 + \rho_{L2} \parallel B \parallel_F^2 + \rho_{L1} \parallel B \parallel_{2,1} \right) \tag{13}$$

where $\mathscr{L}$ is the coresponding label matrix. To solve Eq.(9), we just need to set $\mathscr{L} = M^{t+1} + \mu^t$, $\rho_{L1} = \frac{\lambda_2}{\rho}$, and $\rho_{L2} = \frac{\lambda_1}{2\rho}$. Then Eq.(9) can be solved via alternating minimization algorithm proposed in [1] or Nesterov's method with efficient Euclidean projection [16]. In this paper, we choose the method proposed in [16] as the $l_{2,1}$ solver because it only requires $O(\frac{1}{\sqrt{\epsilon}})$ iterations to achieve an accuracy of ϵ.

Combining all of the above components, the proposed method can be summarized as shown in Algorithm 1. We initalize the M^0 to be the target matrix, and then updating M^t and B^t based on the two steps we discussed above, accordingly.

Algorithm 1: Proposed MTLSA Algorithm

```
    Input: Feature matrix $X$, Target matrix $Y$,
           Weight matrix $W$, $\rho$, $\lambda_1$, $\lambda_2$
    Output: $\hat{B}$
1   Initialize: $t = 0, M^t = Y, \mu^t = \mathbf{0}, B^t = \mathbf{0}$;
2   repeat
3       Compute $M^{t+1}$ by solving Eq.(12);
4       Let $\mathscr{L} = M^{t+1} + \mu^t$, $\rho_{L1} = \frac{\lambda_2}{\rho}$, and $\rho_{L2} = \frac{\lambda_1}{2\rho}$;
5       Compute $B^{t+1}$ via solving Eq.(13) by standard
        $l_{2,1}$ solvers;
6       Compute $\mu^{t+1} = \mu^t + M^{t+1} - XB^{t+1}$ ;
7       $t = t + 1$;
8   until Convergence;
9   $\hat{B} = B^t$;
```

3.4 Algorithm analysis

In this section, we will provide the details about the convergence and the time complexity of the proposed **MTLSA** algorithm.

3.4.1 Convergence analysis

The problem proposed in Eq.(6) follows the standard ADMM form:

$$\begin{aligned} \underset{M \in P,\, B \in \mathbb{R}^{q \times k}}{\text{minimize}} \quad & f_1(M) + f_2(B) \\ \text{subject to} \quad & M = XB \end{aligned} \tag{14}$$

where $f_1(M) = \frac{1}{2} \parallel \Pi_W(Y - M) \parallel_F^2$ and $f_2(B) = \frac{\lambda_1}{2} \parallel B \parallel_F^2 + \lambda_2 \parallel B \parallel_{2,1}$ are convex functions, P and $\mathbb{R}^{q \times k}$ are closed convex sets. Due to space constraints, we do not provide the proof of convergence and the readers are referred to [3, 8] which provide the details of the convergence of the standard ADMM form given in Eq.(14). Based on the analysis in [8], the Algorithm 1 requires $O(\frac{1}{\epsilon})$ iterations to achieve an accuracy of ϵ.

3.4.2 Complexity analysis

We first analyze the time complexity of Step 1. In Eq.(12) $\tilde{M}_{ij}$ can be calculated with a time complexity of $O(q)$ because $S_{ij} = \mu_{ij}^t - X_i B_{(,j)}^t$ where X_i (the i^{th} row of X) and $B_{(,j)}^t$ (the j^{th} column of B^t) all have q elements. For one instance (row), the Euclidean projection in Eq.(12) can be efficiently calculated with a worst-case complexity of $O(k)$ [17], so for all n instances the Euclidean projection can be calculated in $O(nk)$. Therefore, the updating of M^{t+1} needs in total $O(nqk)$ calculations, where n, q, and k denote the training sample size, feature dimensionality, and the number of tasks, respectively.

From [16], we know that in Step 2, the standard $l_{2,1}$-norm regularized multi-task least squares problem in Eq.(13) can be solved with a time complexity of $O(\frac{1}{\sqrt{\epsilon}}(nqk + qk)) = O(\frac{1}{\sqrt{\epsilon}}nqk)$ to achieve an accuracy of ϵ. Moreover, based on the convergence rate of Algorithm 1, we can conclude that the total time complexity of the proposed method is $O(\frac{1}{\epsilon\sqrt{\epsilon}}nqk)$ for achieving an ϵ-level optimal solution.

3.5 Adaptive variant of MTLSA model

We also develop a variant of the **MTLSA** model in order to be able to effectively handle large number of tasks. The **MTLSA** model proposed earlier strictly enforces that the estimated output follows the non-negative non-increasing structure which is the inherent nature of the target matrix Y. However, when the problem has a large number of tasks, this constraint may become too strict that may lead to model overfitting. Thus, the new variant model, **MTLSA.V2**, will not enforce the *non-negative non-increasing list* structure constraint, in the training phase. Mathematically, the proposed **MTLSA.V2** model is defined as:

$$\operatorname*{arg\,min}_{B \in \mathbb{R}^{q \times k}} \frac{1}{2} \parallel \Pi_W(Y - XB) \parallel_F^2 + \frac{\lambda_1}{2} \parallel B \parallel_F^2 + \lambda_2 \parallel B \parallel_{2,1} \tag{15}$$

This problem can be efficiently solved using the fast iterative shrinkage thresholding algorithm (FISTA) algorithm with $l_{2,1}$ projection [16] by incorporating the $\Pi_W(\cdot)$ when calculating the objective function and its gradient. Note that in the testing phase of both **MTLSA** and **MTLSA.V2**, the *non-negative non-increasing list* structure regularization will be used on $X\hat{B}$ to enforce the estimated output of test instances follow the property of non-recurring event survival analysis.

4. EXPERIMENTAL RESULTS

In this section, we will first describe the datasets used in our evaluation and then provide the performance results along with the implementation details. We also demonstrate the scalability of the proposed method.

4.1 Dataset description

For our evaluation, we used several publicly available high-dimensional gene expression cancer survival benchmark datasets [2]. The datasets used in our experiments are as follows:

- The Norway/Stanford breast cancer data (NSBCD) [23] contains gene expression measurements of 115 women with breast cancer. These women are observed for 188 months to monitor the death time.
- Van de Vijver's Microarray Breast Cancer data (VDV) [28] contains gene expression profile information which can be used for predicting the clinical outcome of breast cancer. It contains 4,707 gene expression values on 78 patients with survival information for 13 years.
- Adult myeloid leukemia (AML) data contains gene expression profiles of 116 AML patients with a maximum follow-up time of 1,625 days. In our experiments, we transform the observation time from daily basis to monthly basis and hence the observation lasts for 54 months.
- Gene-expression profiles of lung adenocarcinoma (Lung) [2] is a dataset containing observations of 86 early-stage lung adenocarcinoma patients for a period of 110 months.
- Mantle Cell Lymphoma (MCL) [3] [21] is the data collected from 92 MCL patients with survival information for 14 years.
- The Dutch Breast Cancer Data (DBCD) from van Houwelingen et al. [27] contains information on 4,919 gene expression levels for 295 women with breast cancer. The maximum follow-up time of these patients was 18 years.
- Diffuse Large B-Cell Lymphoma (DLBCL) is a dataset that contains Lymphochip DNA microarrays from 240 biopsy samples of DLBCL tumors for studying the survival status of the corresponding patients and the observation lasts 21 years.

Table 1 provides the details of the datasets that are used in our experiments. In this table, the column titled "# Censored" corresponds to the number of censored instances in each dataset. In cancer survival prediction, the event of interest is patient death; therefore, an uncensored instance corresponds to the death of the patient during the study, while a censored instance corresponds to the patient being still alive at the last observed time (censored time). Based on the study duration of each dataset, we translate the survival prediction problem to a corresponding multi-task problem as described in Section 3.1. The number of tasks for each data is given in the column titled "# Tasks" in the table. For model evaluation, we used 5-fold cross validation when the number of instances is greater than 150 and 3-fold cross validation otherwise.

Table 1: Details of the datasets used in this paper.

Dataset	**# Instances**	**# Features**	**# Censored**	**# Tasks**
NSBCD	115	549	77	188
VDV	78	4705	44	13
AML	116	6283	49	54
Lung	86	7129	62	110
MCL	92	8810	28	14
DBCD	295	4919	216	18
DLBCL	240	7399	102	21

[2]http://user.it.uu.se/~liuya610/download.html
[3]http://llmpp.nih.gov/MCL/

4.2 Comparison methods

We comprehensively compare our proposed methods with several popular state-of-the-art related methods. We now summarize the comparison methods into five categories, and we will briefly describe the basic idea and also provide the implementation details.

- **Cox based models**: The Cox proportional hazards model [5] is the most commonly used semi-parametric model in survival analysis. The *hazard function* has the form $\lambda(t, X_i) = \lambda_0(t)exp(X_i\beta)$, where the $\lambda_0(t)$ is the common *baseline hazard function* for all instances and β is the coefficient vector which can be estimated by minimizing the negative *log-partial likelihood* function. The Cox model can be trained by using the *coxph* function in the *survival* package [24]. The l_1-norm penalized Cox model "LASSO-COX" and elastic net penalized Cox model "EN-COX" can be learned using the *cocktail* function in the *fastcox* package [34].
- **Parametric censored regression models**: In parametric models, the joint probability of the uncensored instances can be formulated as a product of *death density functions* and the joint probability of the censored instances can be formulated as a product of *survival functions*. Thus, a standard likelihood function can be built by combining these two components and the corresponding model parameters are estimated using the maximum-likelihood estimation (MLE) procedure [11]. In our experiments, the parametric censored regression methods are trained using the *survreg* function in the *survival* package with Weibull, Logistic, Loglogistic, and Loggaussian distributions.
- **Linear models**: Tobit model [26] is an extension of the linear regression $y_j = X_j\beta + \varepsilon_j, \varepsilon_j \sim N(0, \sigma^2)$, but the parameters are estimated by the maximum likelihood method rather than using the least squares error. It can be trained using the *survreg* function with Gaussian distributions. The elastic net penalized Buckley-James regression "EN-BJ" is implemented using the *bujar* package [32]. We also compared our proposed methods with the ordinary least squares (OLS) linear regression since the loss function in our model has a similar form to the OLS. Note that, the OLS is not a censored regression method and hence it is learned using only the uncensored instances rather than the entire set of training instances.
- **Pairwise ranking based models**: Boosting concordance index (BoostCI) [18] is an approach where the concordance index metric is modified into an equivalent smoothed criterion using the sigmoid function and the resulting optimization problem is solved using a gradient boosting algorithm. The implementation of BoostCI (using R code) can be found in the supporting file of [18] [4].
- **Multi-task learning models**: We compared our proposed methods with the standard multi-task learning models, multi-task Lasso (Multi-LASSO) and $l_{2,1}$-norm based multi-task feature learning method (Multi-$l_{2,1}$). In MALSAR package, these two models are learned via "Lest_L21" and "Lest_Lasso" functions, respectively [35]. Note that, the these two methods cannot handle censored instances, so the model is learned using only the uncensored instances rather than the entire set of training instances,

[4]files.figshare.com/1339232/Text_S1.pdf

Table 2: Performance comparison of the proposed methods and other existing related methods using C-index values (along with their standard deviations).

		NSBCD	**VDV**	**AML**	**Lung**	**MCL**	**DBCD**	**DLBCL**
COX based	COX	0.4411 (0.0589)	0.5973 (0.1097)	0.5515 (0.0683)	0.5158 (0.1333)	0.5773 (0.0591)	0.5539 (0.1233)	0.4553 (0.0718)
	LASSO-COX	0.5910 (0.1086)	0.6484 (0.0276)	0.5995 (0.0307)	0.6698 (0.0910)	0.6824 (0.0701)	0.6880 (0.0429)	0.6344 (0.0421)
	EN-COX	0.6046 (0.1000)	0.6422 (0.0681)	0.5715 (0.0596)	0.6652 (0.0702)	0.6734 (0.0733)	0.7214 (0.0306)	0.6488 (0.0394)
Parametric models	Logistic	0.3787 (0.0195)	0.5276 (0.1404)	0.4544 (0.0772)	0.5714 (0.0942)	0.4827 (0.0682)	0.4908 (0.0872)	0.4840 (0.0496)
	Weibull	0.3045 (0.1528)	0.3159 (0.1321)	0.5286 (0.0546)	0.4287 (0.1023)	0.4735 (0.0747)	0.4555 (0.1046)	0.2507 (0.0627)
	Log-gaussian	0.4435 (0.0539)	0.5210 (0.1653)	0.4048 (0.0651)	0.4122 (0.0754)	0.2564 (0.0715)	0.4875 (0.0553)	0.3167 (0.0914)
	Log-logistic	0.2378 (0.0500)	0.5267 (0.1071)	0.4677 (0.0800)	0.5924 (0.0655)	0.4802 (0.0724)	0.5257 (0.0232)	0.4246 (0.1243)
Linear models	OLS	0.6333 (0.1108)	0.5206 (0.0163)	0.4555 (0.0595)	0.5743 (0.0658)	0.5007 (0.1059)	0.5690 (0.0744)	0.5024 (0.1023)
	Tobit	0.3733 (0.0214)	0.5192 (0.1581)	0.4726 (0.0759)	0.4689 (0.1358)	0.4591 (0.0322)	0.4869 (0.0762)	0.4969 (0.0527)
	BJ-EN	0.6215 (0.0924)	0.6081 (0.0646)	0.6500 (0.0585)	0.6646 (0.1324)	0.7234 (0.1099)	0.7094 (0.0391)	0.6285 (0.0726)
Ranking based	Boost-CI	0.6263 (0.0831)	0.6650 (0.0594)	0.5817 (0.0501)	0.5713 (0.0926)	0.7049 (0.0956)	0.7103 (0.0426)	0.6082 (0.0296)
Multi-task based	Multi-LASSO	0.6117 (0.1493)	0.5293 (0.1083)	0.5088 (0.0952)	0.4410 (0.1655)	0.6539 (0.0140)	0.6256 (0.0749)	0.6104 (0.0510)
	Multi-$l_{2,1}$	0.6100 (0.1700)	0.5973 (0.1440)	0.5246 (0.0285)	0.5248 (0.1130)	0.6912 (0.0602)	0.6899 (0.0720)	0.6115 (0.0512)
	MTLSA.V2	**0.6858** (**0.0834**)	0.6727 (0.0429)	0.6592 (0.0554)	**0.6769** (**0.0271**)	0.7079 (0.0963)	0.7515 (0.0625)	**0.6545** (**0.0600**)
	MTLSA	0.6820 (0.0446)	**0.7008** (**0.0330**)	**0.7145** (**0.0493**)	0.6327 (0.0753)	**0.7274** (**0.1257**)	**0.7581** (**0.0304**)	0.6527 (0.0713)

and the labeling matrix is generated based on the scheme presented in Section 3.1.

4.3 Performance comparison

Due to the presence of censoring in the data, the standard evaluation metrics for regression such as root of mean square error and R^2 are not suitable for measuring the performance in survival analysis [9]. Instead, the concordance index (C-index), or the *concordance probability*, is used to measure the performance of prediction models in survival analysis [7]. Let us consider a pair of bivariate observations (y_1, $\hat{y}_1$) and (y_2, $\hat{y}_2$), where y_i is the actual observation, and $\hat{y}_i$ is the predicted one. The concordance probability is defined as:

$$c = Pr(\hat{y}_1 > \hat{y}_2 | y_1 \geq y_2) \tag{16}$$

By definition, the C-index has the same scale as the classical area under the ROC (AUC) in binary classification, and if y_i is binary, then the C-index is same as the AUC. In the Cox based models, the instances with a low hazard rate should survive longer, and the C-index will be calculated as follows:

$$c = \frac{1}{num} \sum_{i \in \{1 \cdots N\} \delta_i = 1} \sum_{y_j > y_i} I(X_i \hat{\beta} > X_j \hat{\beta}) \tag{17}$$

where num denotes the number of comparable pairs and $I[\cdot]$ is the indicator function. The C-index in other methods which aim at directly learning the survival time should be calculated as:

$$c = \frac{1}{num} \sum_{i \in \{1 \cdots N\} \delta_i = 1} \sum_{y_j > y_i} I[S(\hat{y}_j | X_j) > S(\hat{y}_i | X_i)] \tag{18}$$

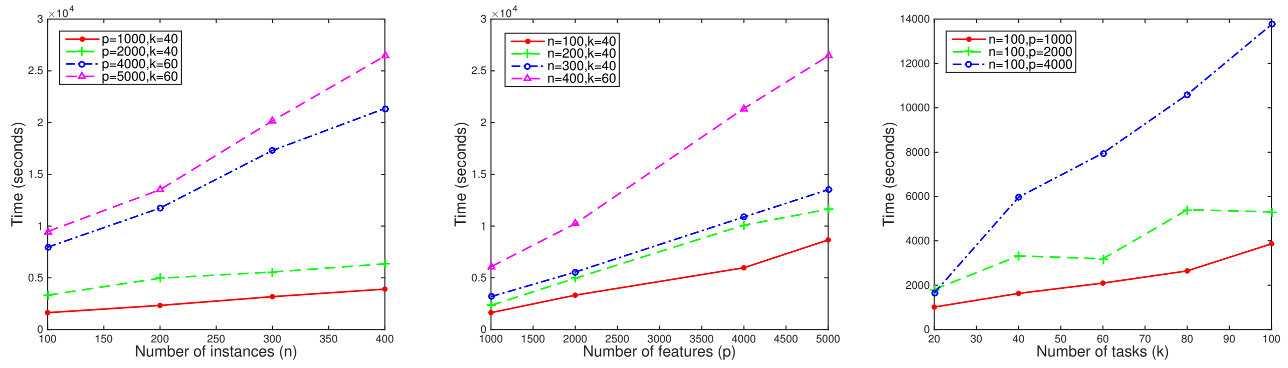
where $S(\hat{y}_i|X_i)$ is the predicted target value. Multi-task learning models cannot directly predict the survival time but they can determine whether an instance is alive or not at each time interval (or task); thus, based on this information, we can predict the survival time. In Table 2, we provide the performance results of C-index values of different algorithms on various real-world high-dimensional micro-array cancer survival datasets. The best results are highlighted in bold. The results show that our proposed models outperform the other state-of-the-art models[5].

The C-index measures the model performance in regression problems. In addition to it, we also evaluate the model performance in classification problems which corresponds to whether a patient can survive at each time interval or not. Since censoring occurs, the number of patients, who have a known survival status label ("1" or "0" in target matrix Y), will reduce during the observation period, (as shown in

[5]The method described in [15] was not able run on our high-dimensional datasets and hence we were not able to obtain any results for that method.

Table 3: Performance comparison of the proposed methods and other existing related methods using Weighted average of AUC (along with their standard deviations).

		NSBCD	**VDV**	**AML**	**Lung**	**MCL**	**DBCD**	**DLBCL**
COX based	COX	0.4611 (0.1893)	0.6352 (0.1666)	0.5351 (0.0814)	0.5464 (0.1632)	0.4695 (0.1701)	0.5334 (0.1620)	0.4480 (0.1079)
	LASSO-COX	0.5986 (0.1589)	0.6857 (0.0456)	0.7277 (0.0346)	0.7499 (0.1780)	0.7401 (0.0166)	0.7068 (0.0292)	0.7104 (0.0533)
	EN-COX	0.6479 (0.0970)	0.6770 (0.0978)	0.6819 (0.0790)	0.7540 (0.1398)	0.7350 (0.0025)	0.7497 (0.0189)	0.7260 (0.0618)
Parametric models	Logistic	0.4597 (0.1742)	0.5917 (0.1433)	0.4918 (0.0417)	0.6301 (0.0924)	0.2986 (0.0501)	0.4840 (0.1086)	0.5011 (0.0489)
	Weibull	0.4575 (0.2622)	0.3177 (0.1369)	0.5227 (0.0393)	0.4379 (0.1018)	0.3240 (0.0484)	0.4707 (0.0809)	0.4320 (0.1080)
	Log-gaussian	0.4992 (0.2378)	0.5647 (0.2026)	0.4718 (0.0206)	0.4182 (0.0680)	0.4457 (0.0161)	0.4742 (0.0763)	0.4270 (0.0977)
	Log-logistic	0.3304 (0.1057)	0.5573 (0.0627)	0.4984 (0.0521)	0.5822 (0.1544)	0.2983 (0.0505)	0.5302 (0.0298)	0.4712 (0.0627)
Linear models	OLS	0.6599 (0.1042)	0.5268 (0.0887)	0.4457 (0.0339)	0.5677 (0.1120)	0.5594 (0.1191)	0.5998 (0.1096)	0.4934 (0.1952)
	Tobit	0.4567 (0.1812)	0.5680 (0.1778)	0.5042 (0.0412)	0.4708 (0.1422)	0.5074 (0.0283)	0.4668 (0.1021)	0.5243 (0.0691)
	BJ-EN	0.6376 (0.1262)	0.6664 (0.0953)	0.7633 (0.0393)	0.7494 (0.1544)	**0.8567** (**0.0306**)	0.7344 (0.0393)	0.6574 (0.0388)
Ranking based	Boost-CI	0.6483 (0.0972)	0.7151 (0.0788)	0.6664 (0.1293)	0.6497 (0.2193)	0.7660 (0.0417)	0.7380 (0.0493)	0.6626 (0.0553)
Multi-task based	Multi-LASSO	0.6495 (0.1226)	0.5166 (0.0502)	0.4802 (0.1090)	0.4410 (0.1655)	0.6079 (0.0696)	0.6402 (0.0572)	0.5876 (0.1047)
	Multi-$l_{2,1}$	0.6501 (0.1314)	0.6463 (0.1510)	0.5247 (0.0316)	0.5589 (0.1486)	0.6476 (0.0653)	0.7125 (0.0775)	0.6001 (0.0528)
	MTLSA.V2	0.6822 (0.0576)	0.7441 (0.0437)	0.7401 (0.0658)	**0.8076** (**0.0559**)	0.7639 (0.0651)	0.7569 (0.0645)	**0.7405** (**0.0719**)
	MTLSA	**0.7032** (**0.0427**)	**0.7659** (**0.0286**)	**0.8098** (**0.0077**)	0.7169 (0.0964)	0.8095 (0.0367)	**0.8003** (**0.0425**)	0.7385 (0.0638)



(a) Scalability w.r.t. n (b) Scalability w.r.t. q (c) Scalability w.r.t. k

Figure 2: Scalability results of the MTLSA model. The times denote total runtime for 100 λ values averaged over three trials.

Figure 1, all 6 instances are labeled in Day1 and Day2, and only 5 instances are labeled in Day3). Thus, in Table 3, we present the comparison of weighted average AUC values of different tasks (time intervals). The weighted average AUC is defined as

$$\mathrm{AUC}_{\mathrm{avg}} = \frac{\sum_{i=1}^{k} \mathrm{AUC}^{(i)} n_{\bar{c}}^{(i)}}{\sum_{i=1}^{k} n_{\bar{c}}^{(i)}} \tag{19}$$

where $\mathrm{AUC}^{(i)}$ is the AUC value of the i^{th} task, and $n_{\bar{c}}^{(i)}$ is the number of instances which have a known survival status label in the i^{th} time interval. The results in Table 3 show that the proposed models obtain higher $\mathrm{AUC}_{\mathrm{avg}}$ in most of the datasets. This demonstrates that our proposed methods have a better time-dependent prediction capability than other related methods.

From Tables 2 and 3, we note that our proposed methods significantly outperform the standard multi-task learning models. This reflects that the model, which is able to handle and utilize both censored and uncensored instances, can provide significantly better results compared to the model which will only use fully observed (uncensored) instances. Our proposed indicator matrix W can appropriately handle the censored instances. We can also observe that for some datasets, especially which have a large number of tasks like NSBCD and Lung, the **MTLSA.V2** performs better than **MTLSA**. This is because the non-negative non-increasing structure is a strict constraint and may affect the flexibility of the **MTLSA** model as mentioned in Section 3.5. We can also observe the standard deviation values of the results obtained from both the models are significantly lower across all of the datasets compared to the other methods. This shows the robustness of our method with respect to obtaining better results across different folds.

4.4 Scalability experiments

We empirically evaluate the scalability of the proposed **MTLSA** method with respect to the sample size (n), the number of features (q) and the number of tasks (k). The synthetic datasets are generated using the the function "simple.surv.sim" in *survsim* package [19] with different sample sizes, feature dimensionality, and maximum follow-up times (which corresponds to the tasks). All the features are generated based on a uniform distribution, and each of them have a different randomly set interval. The coefficient vector is also randomly generated and remains within $[-1, 1]$. The observed time is assumed to follow a Log-logistic distribution and the time to censorship follows a Weibull distribution (which is a standarded practice in survival analysis). Figure 2(a) shows the runtimes for fixed q-k combination and varying n, Figure 2(b) shows the runtimes for fixed n-k combination and varying q, and Figure 2(c) shows the runtimes for fixed n-q combination and varying k. These three plots clearly demonstrate that the runtime of **MTLSA** is close to being *linear with respect to n, q and k*.

5. CONCLUSION

In this paper, we formulated the survival analysis problem as a multi-task learning problem and proposed a new multi-task learning algorithm, **MTLSA**, which is able to handle censored instances in time-to-event data. The **MTLSA** algorithm explicitly models the critical properties of single event survival analysis by imposing the *non-negative and non-increasing list* structural constraint. In addition, the $l_{2,1}$-norm penalty is used to enable the model learn a shared representation across related tasks and hence select important features thus alleviating over-fitting in the high-dimensional feature space. We also develop an adaptive variant, **MTLSA.V2**, which relaxes the structural constraints and produces better results when the number of tasks is large. We extensively compared the performance of the proposed algorithm with state-of-the-art survival analysis methods using several publicly available high-dimensional microarray gene expression datasets using both regression and classification based standard evaluation metrics. We also demonstrated the linear scalability of the proposed model with respect to the sample size, feature dimensionality, and the number of tasks (number of time intervals).

Acknowledgments

This work was supported in part by the US National Science Foundation grants IIS-1231742, IIS-1527827, III-1539991, and III-1539722.

6. REFERENCES

[1] A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

[2] D. G. Beer, S. L. Kardia, C.-C. Huang, T. J. Giordano, A. M. Levin, D. E. Misek, L. Lin, G. Chen, T. G. Gharib, D. G. Thomas, et al. Gene-expression profiles predict survival of patients with lung adenocarcinoma. *Nature medicine*, 8(8):816–824, 2002.

[3] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

[4] J. Buckley and I. James. Linear regression with censored data. *Biometrika*, 66(3):429–436, 1979.

[5] D. R. Cox. Regression models and life-tables. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 187–220, 1972.

[6] B. Efron. The efficiency of cox's likelihood function for censored data. *Journal of the American Statistical Association*, 72(359):557–565, 1977.

[7] F. E. Harrell, R. M. Califf, D. B. Pryor, K. L. Lee, and R. A. Rosati. Evaluating the yield of medical tests. *JAMA*, 247(18):2543–2546, 1982.

[8] B. He and X. Yuan. On the o(1/n) convergence rate of the douglas-rachford alternating direction method. *SIAM Journal on Numerical Analysis*, 50(2):700–709, 2012.

[9] P. J. Heagerty and Y. Zheng. Survival model predictive accuracy and roc curves. *Biometrics*, 61(1):92–105, 2005.

[10] E. L. Kaplan and P. Meier. Nonparametric estimation from incomplete observations. *Journal of the American Statistical Association*, 53(282):457–481, 1958.

[11] E. T. Lee and J. Wang. *Statistical methods for survival data analysis*, volume 476. Wiley. com, 2003.

[12] Y. Li, V. Rakesh, and C. K. Reddy. Project success prediction in crowdfunding environments. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, WSDM '16, pages 247–256, 2016.

[13] Y. Li, B. Vinzamuri, and C. K. Reddy. Regularized weighted linear regression for high-dimensional censored data. In *Proceedings of SIAM International Conference on Data Mining*, pages 45–53, 2016.

[14] Y. Li, K. S. Xu, and C. K. Reddy. Regularized parametric regression for high-dimensional survival analysis. In *Proceedings of SIAM International Conference on Data Mining*, pages 765–773, 2016.

[15] H.-c. Lin, V. Baracos, R. Greiner, and J. Y. Chun-nam. Learning patient-specific cancer survival distributions as a sequence of dependent regressors. In *Advances in Neural Information Processing Systems*, pages 1845–1853, 2011.

[16] J. Liu, S. Ji, and J. Ye. Multi-task feature learning via efficient l 2, 1-norm minimization. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*, pages 339–348. AUAI Press, 2009.

[17] J. Liu, L. Sun, and J. Ye. Projection onto a nonnegative max-heap. In *Advances in Neural Information Processing Systems*, pages 487–495, 2011.

[18] A. Mayr and M. Schmid. Boosting the concordance index for survival data–a unified framework to derive and evaluate biomarker combinations. *PloS one*, 9(1):e84483, 2014.

[19] D. Morina and A. Navarro. The R package survsim for the simulation of simple and complex survival data. *Journal of Statistical Software*, 59(2):1–20, 2014.

[20] C. K. Reddy and Y. Li. A review of clinical prediction models. In C. K. Reddy and C. C. Aggarwal, editors, *Healthcare Data Analytics*. Chapman and Hall/CRC Press, 2015.

[21] A. Rosenwald, G. Wright, A. Wiestner, W. C. Chan, J. M. Connors, E. Campo, R. D. Gascoyne, T. M. Grogan, H. K. Muller-Hermelink, E. B. Smeland, et al. The proliferation gene expression signature is a quantitative integrator of oncogenic events that predicts survival in mantle cell lymphoma. *Cancer cell*, 3(2):185–197, 2003.

[22] N. Simon, J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for cox's proportional hazards model via coordinate descent. *Journal of statistical software*, 39(5):1–13, 2011.

[23] T. Sørlie, R. Tibshirani, J. Parker, T. Hastie, J. Marron, A. Nobel, S. Deng, H. Johnsen, R. Pesich, S. Geisler, et al. Repeated observation of breast tumor subtypes in independent gene expression data sets. *Proceedings of the National Academy of Sciences*, 100(14):8418–8423, 2003.

[24] T. Therneau. A package for survival analysis in s. r package version 2.37-4. *URL http://CRAN. R-project. org/package= survival. Box*, 980032:23298–0032, 2013.

[25] R. Tibshirani. The lasso method for variable selection in the cox model. *Statistics in medicine*, 16(4):385–395, 1997.

[26] J. Tobin. Estimation of relationships for limited dependent variables. *Econometrica: journal of the Econometric Society*, pages 24–36, 1958.

[27] H. C. van Houwelingen, T. Bruinsma, A. A. Hart, L. J. van't Veer, and L. F. Wessels. Cross-validated cox regression on microarray gene expression data. *Statistics in medicine*, 25(18):3201–3216, 2006.

[28] L. J. van't Veer, H. Dai, M. J. Van De Vijver, Y. D. He, A. A. Hart, M. Mao, H. L. Peterse, K. van der Kooy, M. J. Marton, A. T. Witteveen, et al. Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, 415(6871):530–536, 2002.

[29] B. Vinzamuri, Y. Li, and C. K. Reddy. Active learning based survival regression for censored data. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 241–250. ACM, 2014.

[30] B. Vinzamuri and C. K. Reddy. Cox regression with correlation based regularization for electronic health records. *IEEE 13th International Conference on Data Mining (ICDM)*, pages 757–766, 2013.

[31] S. Wang, B. Nan, J. Zhu, and D. G. Beer. Doubly penalized buckley–james method for survival data with high-dimensional covariates. *Biometrics*, 64(1):132–140, 2008.

[32] Z. Wang, C. Wang, et al. Buckley-james boosting for survival analysis with high-dimensional biomarker data. *Statistical Applications in Genetics and Molecular Biology*, 9(1):1–33, 2010.

[33] L. Wei. The accelerated failure time model: a useful alternative to the cox regression model in survival analysis. *Statistics in medicine*, 11(14-15):1871–1879, 1992.

[34] Y. Yang and H. Zou. A cocktail algorithm for solving the elastic net penalized cox's regression in high dimensions. *Statistics and its Interface*, 6(2):167–173, 2012.

[35] J. Zhou, J. Chen, and J. Ye. *MALSAR: Multi-tAsk Learning via StructurAl Regularization.* Arizona State University, 2011.

APPENDIX

A. PROJECTION ONTO A NON-NEGATIVE MAX-HEAP

A non-negative max-heap is an ordered tree where the values of the nodes are all non-negative and the value of any parent node is no less than the value(s) of its child node(s). It can be mathematically defined as:

$$P = \{X \geq 0, x_i \geq x_j | \forall (x_i, x_j) \in E^t\} \tag{20}$$

where $T^t = (V^t, E^t)$ is a target tree with $V^t = \{x_1, x_2, \cdots, x_p\}$ containing all the nodes and E^t denotes all the edges. The non-negative non-increasing list structure defined in Eq.(2) is a special case of non-negative max-heap where the T^t is a sequential list.

In [17], a maximal root-tree based algorithm (Atda) was proposed to solve the non-negative max-heap projection

$$\pi_P(V) = \min_{X \in P} \frac{1}{2} \parallel X - V \parallel^2 \tag{21}$$

Before describing the algorithm itself, we first introduce some definitions to help understand the maximal root-tree.

Definition 1. *For a non-empty tree $T = (V, E)$, its root-tree is any non-empty tree $\tilde{T} = (\tilde{V}, \tilde{E})$ that satisfies: (1) $\tilde{V} \subseteq V$, (2) $\tilde{E} \subseteq E$, and (3) $\tilde{T}$ shares the same root as T.*

Definition 2. *For a non-empty tree $T = (V, E)$, $R(T)$ is defined as the root-tree set which contains all root-trees of T.*

Definition 3. *For a non-empty tree $T = (V, E)$, we define $m(T) = \max\left(\frac{\sum_{v_i \in V} v_i}{|V|}, 0\right)$, which equals to the mean of all the nodes in T if such mean is non-negative, and 0 otherwise.*

Definition 4. *For a non-empty tree $T = (V, E)$, we define its maximal root-tree as:*

$$M(T) = \arg \max_{\tilde{T}=(\tilde{V},\tilde{E}), \tilde{T} \in R(T), m(\tilde{T}) = m_{max}(T)} |\tilde{V}| \tag{22}$$

where $m_{max}(T) = \max_{\tilde{T} \in R(T)} m(\tilde{T})$ is the maximal value of all the root-trees of T, and if some root-trees share the same maximal value then $M(T)$ is the one with the largest tree size.

The key idea of Atda is that, in the i^{th} call, we find $T_i = M(T)$, the maximal root-tree of T, set its corresponding nodes to $m(T_i)$, then remove T_i from the tree T, and apply Atda to the resulting trees one by one in a recursive manner. The detailed discussion to justify the working of Atda along with feasible solution of Eq.(21) is given in [17], and for the non-negative non-increasing list structure Atda has a (worst-case) linear time complexity.