1-1-2016

Novel Regression Models For High-Dimensional Survival Analysis

Yan Li

Wayne State University,

NOVEL REGRESSION MODELS FOR HIGH-DIMENSIONAL SURVIVAL ANALYSIS

by

Yan Li

DISSERTATION

Submitted to the Graduate School

of Wayne State University,

Detroit, Michigan

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

2016

MAJOR: COMPUTER SCIENCE

Approved By:

_____

Advisor Date

_____

_____

_____

DEDICATION

To my dear Wife and Parents.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my advisor, Dr. Chandan Reddy. In the past five years, I have learned a lot from him. With his help, I learned how to survey a field of interest, discover interesting research topics, write research papers and do presentation clearly. From him, I also learned that as a researcher, one needs to be always positive and active, and work harder and harder. What I learned from him would be great wealth in my future study and career.

I am grateful to Dr. Dongxiao Zhu, Dr. Kevin S. Xu, and Dr. Zichun Zhong for serving as committee members on my thesis defense. Their valuable comments are of great help in my future research.

Last but not least, I would like to give my deepest gratitude to my parents Chunhua Li and Jing Wen, who always encourage and support me when I feel depressed. Most of all, I would also like to give my great thanks to my dear wife, Lu Wang. Her kind help and support make everything I have possible.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1 INTRODUCTION

Survival analysis [41] [54] studies the time to event data wherein the observation starts from a particular starting time and will continue until the occurrence of a certain event or a predefined time point. In the healthcare domain, for example, the starting point of the observation is normally a medical intervention such as a hospitalization, the beginning of taking a certain medication or a diagnosis of a given disease. The event might be death, discharge from the hospitalization or any interesting event that can happen during the observation period. The missing trace of observation is a key characteristic of survival data; for example, during the hospitalization some patients may change to an other hospital. Survival analysis is useful whenever someone is interested not only in the frequency of a particular type of event, but also in the time process underlying such an occurrence. In the healthcare field, survival prediction models mainly aim at estimating the failure time distribution and point out the prognostic evaluation of different variables, jointly or singularly considered [56], such as biochemical, histological and clinical characteristics [52].

The prominent prediction methods in survival analysis can be categorized into three types: Cox-based, parametric censored regression, and linear models. The Cox proportional hazards model [18] 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. To deal with high-dimensional data, some regularization methods are proposed. These methods include LASSO-COX [70] which introduces the L_1 norm penalty in the partial log-likelihood loss function, Elastic-Net Cox (EN-COX) [65] which uses the elastic net penalty term and the kernel elastic net penalized Cox regression [75] which modifies the elastic net penalty using a kernel matrix.

However, 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 most of the real-world applications. Also, the Cox model does not predict the survival time directly but rather aims at modeling 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 [24]) or bad scalability (Discrete method [67]).

Apart from the Cox model, linear regression is another important branch of survival analysis. Strictly speaking, linear regression is a specific parametric censored regression; we group linear censored regression models separately because linear regression is the fundamental method in data analysis. Because of censoring, the least-squares estimator cannot be directly used in survival analysis. The Tobit model [71] is the earliest attempt to extend the linear regression for data analysis with censored observations. Later, Buckley-James (BJ) estimator [12] is proposed to solve survival prediction with the combination of the Kaplan-Meier (KM) estimator [39] (which is a non-parametric model). Recently, Wang et al. [78] applied the elastic net penalty to the BJ regression (EN-BJ) for efficiently handling the high-dimensional survival analysis problems. However, these approximation methods will induce bias into the final model since the actual survival times of censored instances cannot be observed. It induces bias because the KM estimation cannot accurately estimate the survival time of censored instances, and this estimated inaccurate survival time will be used to train the model. This makes the prediction problem more complex because the survival time of censored instances are calculated using the integral of the KM estimator.

We propose the **R**egularized **W**eighted **R**esidual **S**um-of-**S**quares "**RWRSS**" algorithm to handle the survival prediction with censored instances in high-dimensional data. In contrast to the Tobit regression model, we solve the prediction problem by optimizing the desired objective function directly rather than doing a maximum likelihood estimation. The loss function that is optimized is regularized using the elastic net penalty which can induce the required sparsity and efficiently handle the high-dimensionality. Comparing with the BJ and EN-BJ methods, our model does not need to compute the KM estimator to approximate the survival time of censored instances during the training process.

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 [43], and that there exists a linear relationship between the parameters of the selected distributions and the features. Thus, these regression models can be viewed as generalized linear models. The latter case, namely Case 2, is also termed as Accelerated failure time (AFT) models [81] 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 [43] are the commonly used distributions in parametric censored regression and the last two are considered to be the AFT models. The prediction performance of parametric censored regression is highly dependent on the choice of distribution. However, if the distribution can be inferred the parametric methods will be very efficient. To enable parametric models handel high-dimensional survival analysis problem we proposed a **U**nified model for **R**egularized **P**arametric **C**ensored **R**egression "**URPCR**" which utilize elestic net as the regularization term and unifies the learning process of regularized parametric censored regression with different probability distribution.

The two models we proposed partially overcome some weaknesses of Cox-based model. However, these two models and all parametric regression based models are highly dependent on the choice of distribution. 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 all the above types of methods, we propose the "**MTLSA**" model, which is 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 in 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.

Collecting labeling information of time-to-event data is very time consuming, i.e., one has to wait for the occurrence of the event of interest from sufficient number of training instances to build robust models. Moreover, in many practical applications, appropriate feature collection can also be extremely expensive and tedious. To over-

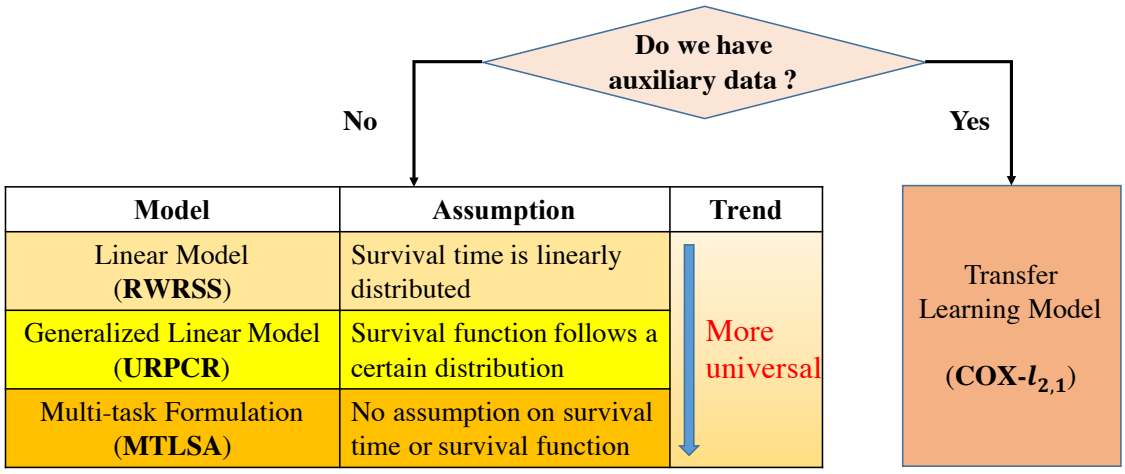


Figure 1.1: The relationship of the models proposed in this thesis.

come the challenge of insufficient uncensored instances, we propose a transfer learning based Cox method, called **Cox-$l_{2,1}$**, which uses auxiliary data to augment learning when there are insufficient number of training examples. The proposed method aims to extract "useful" knowledge from the source domain and transfer it to the target domain, thus potentially improving the prediction performance in such time-to-event data. The proposed method uses the $l_{2,1}$-norm penalty to encourage multiple predictors to share similar sparsity patterns, thus learns a shared representation across source and target domains, potentially improving the model performance on the target task. With the help of a risk set updating method [65], the proposed **Cox-$l_{2,1}$** algorithm achieves a linear time complexity with respect to both training sample size and feature dimensionality. In addition, to speedup the computation, we apply the screening approach and extend the strong rule to sparse survival analysis models in multiple high-dimensional censored datasets.

Figure 1.1 summarize the relationship of the four models proposed in this thesis. There are three standard survival prediction models and one transfer learning model. If the auxiliary data is available then the transfer learninig model, **Cox-$l_{2,1}$**, can be used to improve the prediction performance of the target task. However, if there

is no avliable auxiliary data then these three proposed standard survival prediction models can be used to predict the survival time. These three models are ordered as they become more generalized; the **RWRSS** is built based on a strict assumption, the survival time is linearly distributed; the **URPCR** model is built based on a little more general assumption that the distribution of survival function follows a certain theoretical statistical distribution; and the **MTLSA** is built without any specific assumption on either survival time or survival function and hence it works well in more number of time-to-event datasets than the **RWRSS** and **URPCR** model. However, if the dataset happens to meet the assumptions in **RWRSS** or **URPCR** model then they will perform better than the **MTLSA** model.

1.1 Contributions

The main contributions of this thesis are summarized as follows:

- Introduce some basic concepts in survival analysis and provide a comprehensive review of different categories of survival prediction methods.
- Propose a regularized weighted linear regression model for high-dimensional survival analysis.
- Propose a regularized parametric censored regression model for high-dimensional survival analysis.
- Formulate the survival analysis as a multi-task learning problem and propos an novel method for solving it.
- Propose a novel transfer learning method **Cox-$l_{2,1}$** for survival analysis which can select a subset of joint features to transfer the knowledge from the source domain to the target domain in the presence of censored data.
- Demonstrate the performance of the proposed methods using high-dimensional gene expression datasets from various cancer patients, and comprehensively compare our proposed models with the existing state-of-the-art methods avail-

able in the survival analysis literature.

1.2 Organization of this Thesis

The rest of this thesis is organized as follows. Chapter 2 explains the basic concept of survival analysis with some standard prediction methods in survival analysis. In Chapter 3, we propose a regularized weighted linear regression for high-dimensional survival analysis. In Chapter 4, we propose a regularized parametric censored regression for high-dimensional survival analysis. In Chapter 5, we propose a multi-task learning formulation for survival analysis. In Chapter 6, we propose a transfer learning methods for survival analysis. Chapter 7 concludes the previous work and discusses some future research directions.

CHAPTER 2 EXISTING METHODS IN SURVIVAL ANALYSIS

In this chapter, we will first introduce some basic concepts and functions and then briefly go through some popular methods used in survival analysis.

Survival analysis is an important branch of statistics which aims at predicting the time to the event of interest, and it can simultaneously model event data and censored data. In survival data the event of interest may not always be observed during the study; this scenario happens because of time limits or missing traces caused by other uninteresting events. This concept is known as censoring [41]. Let us consider a small number of N cancer patients to predict the time to patient death, which is the event of interest in this problem, after diagnosed date, and suppose the observation starts at 01/01/2015 and ends at 12/31/2015. Thus, the time to the patient death is known precisely only for those subjects who will have the event during our observation window. For the remaining patients, it is only known that the event of interest will happens after 12/31/2015 but not exact time. Also during this observation time, we lose track of some patients because of moving out of the area or other reasons. Both of these scenarios are considered as censoring in this particular example. Figure 2.1 provides a more intuitive way to describe the idea of censoring.

For each data instance, we observe either a survival time (O_i) or a censored time (C_i), but not both. The dataset is said to be *Right Censored* if and only if $T_i = \min(O_i, C_i)$ can be observed during the study [62]; otherwise, if $T_i = \max(O_i, C_i)$, it is called as *Left Censoring*. Practically, in many real-world domains, the majority of the survival data is right censored [52]. An instance in the survival data is usually represented by a triplet (X_i, T_i, δ_i), where X_i is a $1 \times p$ 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

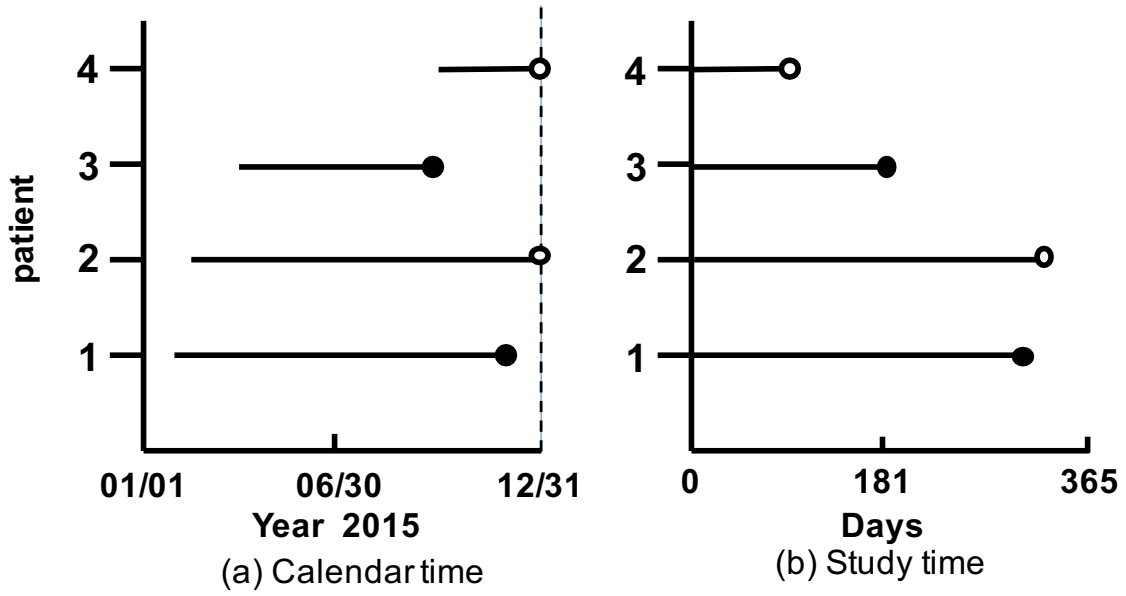


Figure 2.1: Example of censoring: Figure 2.1 (a) shows the observation of four patients during the calendar year and the Figure 2.1 (b) shows the corresponding four patients under the study time, where the diagnosed date is the starting time. Patient 1 and 3 have died during the observation window, so they are uncensored instances. Patient 2 and 4 are still alive at the end of observation window, so they are censored instances.

uncensored instances and C_i otherwise, i.e.,

$$T_i = \begin{cases} O_i & \text{if} \quad \delta_i = 1 \\ C_i & \text{if} \quad \delta_i = 0 \end{cases} \tag{2.1}$$

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.

2.1 Basic Functions of Survival Analysis

In survival analysis, a subject in survival data is usually represented by a triple of variables (X_i, T_i, δ_i), where X_i is the feature vector, and δ_i is an indicator. $\delta_i = 1$ if T_i is the time to the event of interest and $\delta_i = 0$ if T_i is the censored time. The object of primary interest of survival analysis is the *survival function* $S(t) = Pr(O \geq t)$,

which is the probability that the time to the event of interest is no earlier than some specified time t [41]. In contrast, the *cumulative death distribution function* $F(t)$ is defined as $F(t) = 1 - S(t)$, which represents the probability of time to the event of interest is less than t, and *death density function* $f(t)$ is defined as $f(t) = \frac{d}{dt}F(t)$ for continuous scenarios, and $f(t) = \frac{F(t+\Delta t)-F(t)}{\Delta t}$, where Δt is a short time interval, for discrete scenarios. One other function commonly used in survival analysis is the *hazard function* ($\lambda(t)$), which is also known as the *force of mortality*, the *conditional failure rate*, or the *instantaneous death rate* [23]. The hazard function is not the chance or probability of the event of interest, but instead it is the event rate at time t conditional on survival until time t. Mathematically, the hazard function is defined as:

$$\begin{aligned} \lambda(t) &= \lim_{\Delta t \to 0} \frac{Pr(t \leq O < t + \Delta t \mid O \geq t)}{\Delta t} \\ &= \lim_{\Delta t \to 0} \frac{F(t + \Delta t) - F(t)}{\Delta t \cdot S(t)} \\ &= \frac{f(t)}{S(t)} \end{aligned} \tag{2.2}$$

Consider the definition of $f(t)$, which can also be expressed as $f(t) = -\frac{d}{dt}S(t)$, so the hazard function can be represented as:

$$\lambda(t) = \frac{f(t)}{S(t)} = -\frac{d}{dt}S(t) \cdot \frac{1}{S(t)} = -\frac{d}{dt}[lnS(t)]. \tag{2.3}$$

Thus, the survival function can be rewritten as $S(t) = exp(-\Lambda(t))$, where $\Lambda(t) = \int_0^t \lambda(u)du$ is the *cumulative hazard function* (CHF) [43]. The relationship among these functions can be clearly described in Figure 2.2.

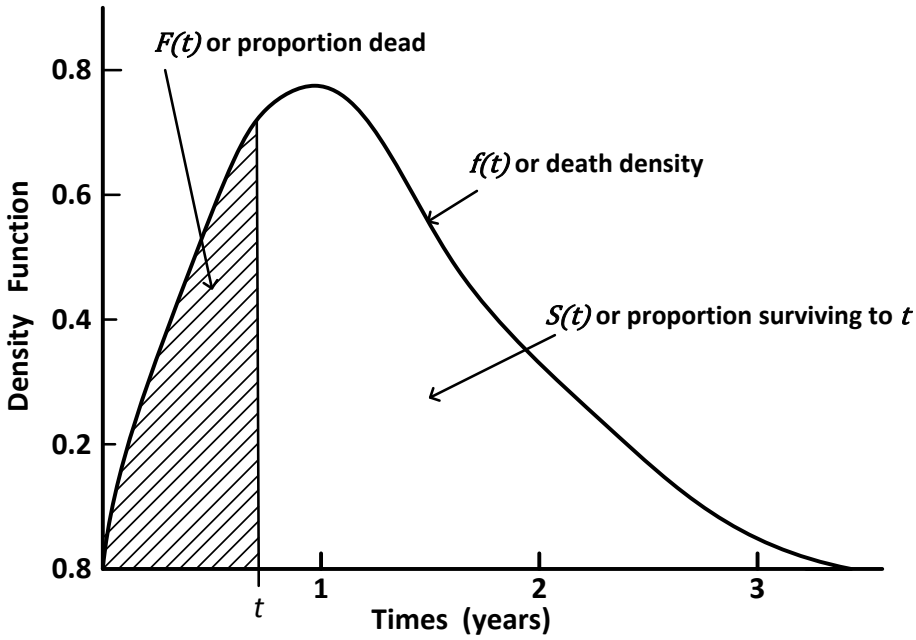


Figure 2.2: Relationship between $f(t)$, $F(t)$, and $S(t)$.

2.2 Non-Parametric Methods for Survival Estimation

Non-parametric or distribution-free methods are quite easy to understand and apply. They are less efficient than parametric methods when survival times follow a theoretical distribution and more efficient when no suitable theoretical distributions are known. In this section, we will introduce non-parametric methods for estimating the survival probabilities for censored data. Among all functions, the survival function or its graphical presentation, the *survival curve*, is the most widely used one. In 1958, Kaplan and Meier [39] developed the *product-limit estimator* or the *Kaplan-Meier Curve* to estimate the survival function based on the actual length of observed time. Let $O_1 < O_2 < ... < O_K$, $K \leq N$, is a set of distinct survival times observed in N individuals; in a certain time O_j $(j = 1, 2, ..., K)$, the $d_j \geq 1$ number of deaths are observed, and the r_j number of subjects, whose either death or censored time is greater than or equal to O_j, are considered to be "at risk". The obvious conditional

probability of surviving beyond time O_j can be defined as: $p(O_j) = \frac{r_j - d_j}{r_j}$, and the survival function at t is estimated by the following product

$$\hat{S}(t) = \prod_{j:O_j<t} p(O_j) = \prod_{j:O_j<t} (1 - \frac{d_j}{r_j}) \tag{2.4}$$

and its variance is defined as:

$$Var(\hat{S}(t)) = \hat{S}(t)^2 \sum_{j:O_j<t} \frac{d_j}{r_j(r_j - d_j)} \tag{2.5}$$

It is worth noting that because of the censoring, r_j is not simply equal to the difference between r_{j-1} and d_{j-1}; the correct way to calculate r_j is $r_j = r_{j-1} - d_{j-1} - c_{j-1}$, where c_{j-1} is the number of censored cases between O_{j-1} and O_j.

However, if the data is already grouped into intervals, or if the sample size is very large, it may be more convenient to perform a *Clinical Life Table* analysis [20]. Where the total number of N subjects are partitioned into J intervals based on the observed time, and the survival function is estimated based on a similar way as done in the Kaplan-Meier Curve.

2.3 Semi-Parametric Methods for Survival Estimation

The Cox proportional hazard model [18] is the most commonly used model in survival analysis. Unlike parametric methods, this model does not require knowledge of the underlying distribution, but the attributes are assumed based on an exponential influence on the output. The baseline hazard function in this model can be an arbitrary nonnegative function, but the baseline hazard functions of different individuals are assumed to be the same. The estimation and hypothesis testing of parameters in the model can be calculated by minimizing the negative partial likelihood function

rather than the ordinary likelihood function.

2.3.1 The Basic Cox Model

Let N be the number of subjects in the survival analysis, and each of the individual can be represented by a triplet of variables (X_i, T_i, δ_i). Consider an individual specific hazard function $\lambda(t, X_i)$ in the Cox model the proportional hazard assumption is

$$\lambda(t, X_i) = \lambda_0(t) exp(X_i \beta) \tag{2.6}$$

for $i = 1, 2, ..., N$, where the $\lambda_0(t)$ is the *baseline hazard function*, which can be an arbitrary non-negative function of time, $X_i = (x_{i1}, x_{i2}, ..., x_{ip})$ is the corresponding covariant vector for individual i, and $\beta^T = (\beta_1, \beta_2, ..., \beta_p)$ is the coefficient vector. The Cox model is a semi-parametric model since it does not specify the form of $\lambda_0(t)$. In fact, the hazard ratio does not depend on the baseline hazard function; for two individuals the hazard ratio is

$$\frac{\lambda(t, X_1)}{\lambda(t, X_2)} = \frac{\lambda_0(t) exp(X_1 \beta)}{\lambda_0(t) exp(X_2 \beta)} = exp[(X_1 - X_2)\beta] \tag{2.7}$$

Since the hazard ratio is a constant, and all the subjects share the same baseline hazard function, the Cox model is a proportional hazard model. Based on this assumption the survival function is given by

$$S(t) = exp(-\Lambda_0(t) exp(X\beta)) = S_0(t)^{exp(X\beta)} \tag{2.8}$$

where $\Lambda_0(t)$ is the *cumulative baseline hazard function*, and $S_0(t) = exp(-\Lambda_0(t))$ is the baseline survival function.

2.3.2 Estimation of the Regression Parameter

Since in the Cox proportional hazard model, the baseline hazard function $\lambda_0(t)$ is not specified, it is impossible to fit the model based on the standard likelihood function. To estimate the coefficient, Cox [18] proposed a partial likelihood which represents the data only depending on the β. Consider the definition of the hazard function, the probability that an individual with covariant X fails at time t conditional on survival until time t can be expressed as $\lambda(t, X)dt, dt \to 0$. Again, let N be the number of subjects who have a total number of $J \leq N$ events of interest occurring during the observation, and $O_1 < O_2 < ... < O_J$ is the distinct ordered times to the event of interest. Without considering the ties, let X_j be the corresponding covatiate vector for the individual who fails at time O_j, and $R(O_j)$ be the set of subjects at time O_j. Thus, conditional on the fact that one individual is observed to fail at O_j, the probability that its corresponding covariant is X_j is

$$\frac{\lambda(O_j, X_j)dt}{\sum_{i \in R(O_j)} \lambda(O_j, X_i)dt} \tag{2.9}$$

and the partial likelihood is the product of this probability; referring to the Cox assumption and the existence of censoring, the definition of the partial likelihood is given by

$$L(\beta) = \prod_{j=1}^{N} \left[\frac{exp(X_j\beta)}{\sum_{i \in R_j} exp(X_i\beta)} \right]^{\delta_j} \tag{2.10}$$

It should be noted that here $j = 1, 2, ..., N$; if $\delta_j = 1$, the j^{th} term in the product is the conditional probability; otherwise, when $\delta_j = 0$, the corresponding term is 1 and has no effect on the result. The estimated coefficient vector $\hat{\beta}$ can be calculated by maximizing this partial likelihood; to achieve a better time efficiency, it is usually

equivalently estimated by minimizing the negative *log-partial likelihood*

$$LL(\beta) = \sum_{j=1}^{N} \delta_j \{X_j\beta - log[\sum_{i \in R_j} exp(X_i\beta)]\} \tag{2.11}$$

2.3.3 Penalized Cox models

Currently, with the development of medical procedures and detection methods, electronic health records (EHR) tend to have more features than before. In some case, the number of features (P) is almost equivalent to or even larger than the number of subjects (N); it is unnecessary or even wrong to fit the prediction model with all the features because of the overfitting [34]. The primary motivation of using sparsity inducing norms is that in high dimensions, it is wise to proceed under the assumption that most of the attributes are not significant, and it can be used to identify the vital features in prediction [29]. In biomedical data analysis, the sparsity inducing norms are also widely used to penalize the loss function [83]. Consider the L_p norm penalty; the smaller the p that is chosen, the sparser the solution, but when $0 \leq P < 1$, the penalty is not convex, and the solution is difficult and often impossible. Commonly, the penalized methods have also been used to do feature selection in the scenarios when $N > P$. In the following paragraph, we will introduce three commonly used penalty functions and their applications in the Cox proportional hazard model.

Lasso [68] is a L_1 norm penalty which can select at most $K = min(N, P)$ features while estimating the regression coefficient. In [70], the Lasso penalty was used along with the log-partial likelihood to obtain the Cox-Lasso algorithm.

$$\hat{\beta}_{lasso} = \min_{\beta} -\frac{2}{N} \left[\sum_{j=1}^{N} \delta_j X_j \beta - \delta_j log \left(\sum_{i \in R_j} e^{X_i \beta} \right) \right] + \lambda \sum_{p=1}^{P} |\beta_p| \tag{2.12}$$

Elastic Net is a combination of the L_1 and squared L_2 norm penalties to obtain

both sparsity and handle correlated feature spaces [86]. For Cox-Elastic Net, Noah Simon et al. [65] implemented the Elastic Net to penalize the log-partial likelihood function

$$\begin{aligned}\hat{\beta}_{elastic\ net} &= \min_{\beta} -\frac{2}{N}\left[\sum_{j=1}^{N}\delta_j X_j\beta - \delta_j log\left(\sum_{i\in R_j} e^{X_i\beta}\right)\right] \\ &+ \lambda\left[\alpha\sum_{p=1}^{P}|\beta_p| + \frac{1}{2}(1-\alpha)\sum_{p=1}^{P}\beta_p^2\right] \end{aligned} \tag{2.13}$$

where $0 \leq \alpha \leq 1$. Different from Cox-Lasso, Cox-Elastic Net can select more than N features if $N \leq P$.

Ridge regression was proposed by Hoerl and Kennard [32] and introduced to Cox regression by verweij et al. [74]. It is a L_2 norm regularization that tends to select all the correlated variables, and shrink their values towards each other. The regression parameters of Cox-Ridge can be estimated by

$$\hat{\beta}_{ridge} = \min_{\beta} -\frac{2}{N}\left[\sum_{j=1}^{N}\delta_j X_j\beta - \delta_j log\left(\sum_{i\in R_j} e^{X_i\beta}\right)\right] + \frac{\lambda}{2}\sum_{p=1}^{P}\beta_p^2 \tag{2.14}$$

In all the three equations (2.12, 2.13, 2.14), $\lambda \geq 0$ is used to adjust the influence introduced by the penalty. The performance of these penalized estimator depends strongly on λ, and the optimal λ_{opt} can be chosen via cross-validation.

2.3.4 CoxBoost

CoxBoost was proposed in [7] to estimate parameter vector (β) in the Cox proportional hazards model. In each boosting step, the CoxBoost adaptively selects a flexible subset of covariates to update the corresponding parameters. In the k^{th} boosting step, the Newton-Raphson step will be separately used for g_k predetermined candidate sets of covariates and the corresponding elements of β will be updated

based on the candidate set which maximizes the improvement of the overall fit of the log-partial likelihood. Let us denote the chosen set using ϕ, the updated estimated coefficient $\hat{\beta}^{(k)}$ of k^{th} boosting step can be calculated as:

$$\hat{\beta}^{(k)} = \begin{cases} \hat{\beta}_j^{(k-1)} + \hat{\gamma}_j^{(k)} & j \in \Phi \\ \hat{\beta}_j^{(k-1)} & j \notin \Phi \end{cases} \quad \forall j = 1, \cdots, p \tag{2.15}$$

where $\hat{\gamma}_j^{(k)}$ is the element of the Newton-Raphson updating in k^{th} boosting step. In addition, the chosen set Φ will not be considered as candidate set in the next boosting step. Thus, in the $(k+1)^{st}$ boosting step, β will be updated based on the remaining $(g_k - 1)$ predetermined candidates sets of covariates.

2.4 Parametric Methods

Parametric methods for estimating the survival probability are efficient and accurate when survival times follow a particular distribution. Unlike the Cox proportional hazard model, in parametric methods, a complete likelihood function can be solved directly, and the parameters can be estimated using maximum-likelihood estimation (MLE) [43]. We now discuss the generic MLE procedure [19] used for survival data with censored observations.

Consider a set of N instances out of which there are c censored observations and $(N - c)$ uncensored observations. For convenience, we use the general notation $\mathbf{b} = (b_1, b_2, \cdots, b_p)$ to represent a set of parameters and assume that the survival times follow a probability distribution with survival function $S(t, \mathbf{b})$ and death density function $f(t, \mathbf{b})$. If the i^{th} instance is a censored observation, then it is not possible to obtain the actual survival time; however, it can be concluded that the event of interest did not occur until the censored time C_i, so $S(C_i, \mathbf{b})$ should be a probability value that is close to 1. On the contrary, if the i^{th} instance is an uncen-

sored observation with survival time O_i, then $f(O_i, \mathbf{b})$ should be a high probability value. Thus, we can use $\prod_{\delta_j=1} f(T_i, \mathbf{b})$ to represent the joint probability of the $(N-c)$ uncensored observations and $\prod_{\delta_j=0} S(T_i, \mathbf{b})$ to represent the joint probability of the c right-censored observations. Therefore, the likelihood function of all N instances is given by

$$L(\mathbf{b}) = \prod_{\delta_i=1} f(T_i, \mathbf{b}) \prod_{\delta_i=0} S(T_i, \mathbf{b}) \tag{2.16}$$

Note that $\mathbf{b}$ is not the feature coefficient vector but the parameters of the assumed distribution. These models assume that there is a linear relationship between the feature vector and the theoretical distribution parameters, in other words, $b_k = X\beta$ where b_k is a theoretical distribution parameter and β is the coefficient vector. We can see that the parametric censored regression methods can be viewed as an extension of the standard parametric regression methods; if there are no censored observations, the censored regression methods will automatically reduce to the corresponding standard regression methods.

Choosing an appropriate theoretical distribution to approximate the success curve is a critical component of the parametric methods [44]. However, since the distribution is not known apriori, one has to fit different models to the data in the best possible manner. Table 1 shows the density, survival, and hazard functions of some commonly used distributions for parametric survival regression. We will discuss their details in subsequent paragraphs.

Weibull Distribution: The Weibull model is the most widely used parametric survival model. It's hazard function is in the form of

$$h(t) = \lambda p t^{p-1} \tag{2.17}$$

where $\lambda > 0$. p is a shape parameter and determines the shape of the hazard function.

Table 2.1: Density, Survival and Hazard functions for commonly used distributions.

Distribution	PDF $f(t)$	Survival $S(t)$	Hazard $h(t)$
Weibull	$\lambda k(\lambda t)^{k-1}exp(-(\lambda t)^k)$	$exp((-\lambda t)^k)$	$\lambda k t^{k-1}$
Exponential	$\lambda exp(-\lambda t)$	$exp(-\lambda t)$	λ
Log-logistic	$\lambda p t^{p-1}(1+\lambda t^p)^{-2}$	$\frac{1}{1+\lambda t^p}$	$\frac{\lambda p t^{p-1}}{1+\lambda t^p}$
Log-normal	$\frac{1}{\sqrt{2\pi}\sigma t}exp(-\frac{(ln(t)-\mu)^2}{2\sigma^2})$	$exp(\mu+\frac{\sigma^2}{2})$	$exp(2\mu+\sigma^2)(exp(\sigma^2)-1)$
Gamma	$\frac{\lambda^k t^{k-1}e^{-\lambda t}}{\Gamma(k)}$	$1-\frac{\int_0^t \sigma^{k-1}e^{-\sigma}d\sigma}{\Gamma(k)}$	$\frac{\lambda^k t^{k-1}e^{-\lambda t}}{(1-\frac{\int_0^t \sigma^{k-1}e^{-\sigma}d\sigma}{\Gamma(k)})\Gamma(k)}$

If $p = 1$, the hazard is constant and the Weibull model will become the exponential model with $h(t) = \lambda$. If $p < 1$, the hazard decreases over time. The Weibull model is more flexible than the exponential model due to the shape parameter p.

There are two important properties for Weibull model. One is that if the AFT assumption holds then the PH assumption also holds. In addition, based on the survival function $S(t) = exp(-\lambda t^p)$ for Weibull model, we can have

$$ln[-lnS(t)] = ln(\lambda) + pln(t) \tag{2.18}$$

where t denotes time. It means that for Weibull model $ln[-lnS(t)]$ is a linear function of $ln(t)$ with slope p and intercept $ln(\lambda)$. By this key property, we can also evaluate the Weibull model by plotting this linear relationship.

Exponential Distribution: Exponential model is the simplest parametric survival model in that the hazard is constant over time, which means $h(t) = \lambda$. However, the assumption that the hazard is constant for each pattern of covariates is a much stronger assumption than the PH assumption. If the hazards are constant, then the ratio of the hazards will remain constant as well. However, the hazard ratio being constant does not necessarily mean that each hazard is constant. In other words, the baseline hazard function in Cox model is not specified, but it is a PH model.

The regression coefficients are estimated using maximum likelihood estimation (MLE), and are asymptotically normally distributed.

It should be noted that, parametric survival models need not to be PH models. Many parametric models are acceleration failure (AFT) models rather than PH models. The exponential and Weibull distributions can accommodate both the PH and AFT assumptions. The interpretation of the estimated results, it also differs for AFT models and PH models. The goal to compare the hazards for PH model, while AFT is applicable for a comparison of survival times.

Log-logistic Distribution: Log-logistic distribution parametric censored regression model accommodates an AFT model but not a PH model. In contrast to Weibull model, its hazard function allows for some non-monotonic behavior in the hazard function, which is in the form of

$$h(t) = \frac{\lambda p t^{p-1}}{1 + \lambda t^p} \tag{2.19}$$

where $p > 0$ is the shape parameter. If $p \leq 1$, the hazard decreases over time. If $p > 1$, however, the hazard increases to a maximum point and then decreases over time, which means that the hazard function is unimodal if $p > 1$.

The Log-logistic AFT model is a proportional odds (PO) model instead of a PH model. For a PO survival model, the odds ratio is assumed to remain constant over time.

Here, we introduce two new definitions for Log-logistic model. The survival odds (SO) is defined as the odds of surviving beyond time t in the form of

$$\frac{S(t)}{1 - S(t)} = \frac{P(T > t)}{P(T \leq t)} \tag{2.20}$$

and the failure odds (FO), which is the reciprocal of survival odds, means the odds

of getting the event by time t in the form of

$$\frac{1 - S(t)}{S(t)} = \frac{P(T \leq t)}{P(T > t)} \tag{2.21}$$

According to the Log-logistic survival function in Table 1, the failure odds equals to λt^p, which indicates a linear relationship between the log odds of the failure and the log of time. This is also helpful for the evaluation by plotting $log(FO)$ against $ln(t)$. The plots will be a line with slope p if the survival time in the data follows a Log-logistic distribution.

$$log(FO) = ln(\lambda) + pln(t) \tag{2.22}$$

Log-normal Distribution: The Log-normal model has a relatively complicated hazard and survival function that can only be expressed in terms of integrals. The shape of the Log-normal distribution is very similar to the Log-logistic distribution and yields similar results.

Generalized Gamma Distribution: The hazard and survival function of generalized Gamma model is also complicated. It has three parameters allowing more flexibility in its shape. Actually, the Weibull and Log-normal distributions are two special cases of the generalized gamma distribution.

2.5 BoostCI

The concordance index (C-index), or *concordance probability*, is the most commonly used evaluation metric in survival analysis [28]. Considering 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

$$c = Pr(\hat{y}_1 > \hat{y}_2 | y_1 \geq y_2). \tag{2.23}$$

Thus, we can see that concordance index is a pairwise ranking based evaluation metric.

Boosting concordance index (BoostCI) [53] is an approach which aims at directly optimizing the C-index. In BoostCI, the sigmoid function is used to approximate the indicator function and hence make the C-index smooth, and a gradient boosting based algorithm is proposed to solve the optimization problem.

The goal of BoostCI is to estimate the optimal prediction function y^* that maximizes the C-index by minimizing the empirical risk function using gradient boosting algorithms. The C-index used in this algorithm is defined by Uno et al. [72], which is formulated as:

$$C = \frac{\sum_{i,k}(\hat{G}_n^L(\tilde{T}_i))^{-2} I(\tilde{T}_i < \tilde{T}_k) I(\hat{\eta}_i > \hat{\eta}_k)\delta_i}{\sum_{i,k}(\hat{G}_n^L(\tilde{T}_i))^{-2} I(\tilde{T}_i < \tilde{T}_k)\delta_i} \tag{2.24}$$

where $\hat{G}_n^L(t)$ is the Kaplan-Meier estimator, $\hat{\eta}_i$ and $\hat{\eta}_k$ are the predicted risk marker values for instance i and k, respectively. By introducing the sigmoid function, $K(u) = 1/(1 + exp(-u/\sigma))$, the smoothed C-index function will be in the form of

$$C_{smooth} = \sum_{i,k} w_{ik} \frac{1}{1 + exp(\frac{\hat{\eta}_k - \hat{\eta}_i}{\sigma})} \tag{2.25}$$

where σ is a tuning parameter that control the smoothness of the approximation, and

$$w_{ik} = \frac{\delta_i(\hat{G}_n^L(\tilde{T}_i))^{-2} I(\tilde{T}_i < \tilde{T}_k)}{\sum_{i,k} \delta_i(\hat{G}_n^L(\tilde{T}_i))^{-2} I(\tilde{T}_i < \tilde{T}_k)} \tag{2.26}$$

The negative of the Eq.(2.25) can be minimized via a standard component-wise gradient boosting algorithm.

2.6 Survival Trees

Survival Trees are one form of classification and regression trees which are tailored to handle censored data. The basic intuition behind the tree models is to recursively

partition the data based on a particular splitting criterion, and the objects which belong to the same node are similar to each other based on the event of interest. The earliest attempt at using tree structure analysis for survival data was made in [15]. The primary difference between a survival tree and the standard decision tree is the choice of splitting criterion, where the splitting methods should be able to measure the difference between the survival distributions. Commonly, the survival distribution difference can be measured by either non-parametric methods, semi-parametric methods, or parametric methods. And this is the reason why we cannot classify the survival trees into any of the three categories above.

The splitting criteria used for survival trees can be grouped into two categories: minimizing within-node homogeneity or maximizing between-node heterogeneity. The first class of approaches minimizes a loss function based on within-node homogeneity criterion. Gordon and Olshen [27] measured the homogeneity using L_P, L_P Wasserstein metric, and Hellinger distances (non-parametric method) between estimated distribution functions. Davis and Anderson [22] employed an exponential log-likelihood loss function in recursive partitioning based on the sum of residuals from the Cox model (semi-parametric method). Leblanc and Crowley [42] measured the node deviance based on the first step of a full likelihood estimation procedure (parametric method); Cho and Hong [14] proposed an L_1 loss function to measure the within-node homogeneity. In the second category of splitting criteria, Ciampi et al. [17] employed log-rank test statistics for between-node heterogeneity measures. Later, Ciampi et al. [16] proposed a likelihood ratio statistic (LRS) to measure the dissimilarity between two nodes. Based on the Tarone-Ware class of two-sample statistics, Segal [64] introduced a procedure to measure the between-node dissimilarity.

To overcome the instability of a single tree, bagging [9] and random forests [10], proposed by Breiman, are commonly used to perform the ensemble based model build-

ing. Hothorn et al. [33] proposed a general bagging method which was implemented in the R package "ipred". In 2008, Ishwaran et al. introduced a general random forest method, called random survival forest (RSF) [36] and implemented it in the R package "randomSurvivalForest".

CHAPTER 3 REGULARIZED WEIGHTED LINEAR REGRESSION

3.1 Motivation and Overview

Due to the emergence of a wide range of data acquisition technologies, it has become a common practice in many domains to monitor subjects over a period of time in order to tell if there are any interesting events (such as device failure, disease occurrence, etc.) that occur. Such monitoring typically starts from a particular time point and lasts until a certain event of interest occurs [43]. Due to time limitations or loss of data traces, however, the event of interest may not always be observed during the study period. This phenomenon is known as censoring and makes this problem more challenging for any standard regression methods. For the instances where the event of interest is observed, the time to the event of interest is known as the failure time (or event time); while for the remaining (censored) instances, the last observed time is known as the censored time.

Figure 3.1 shows three timelines to demonstrate the relationship between the estimated survival time and the censored time in the right censored case. Based on the definition of right censoring, the timeline can be separated into two parts, the range of impossible survival time and the range of possible survival time using the censored time (Figure 3.1(a)). Hence, there are two cases that can arise. Case 1: when the estimated survival time is lesser than the censored time, then it falls within the range of impossible survival time and the real difference between the estimated survival time and the actual survival time is definitely greater than the difference between the estimated survival time and the observed censored time (Figure 3.1(b)). Thus, the model should give more emphasis to this case with assigning more weight in the loss function for this case in order to reduce such an occurrence. Case 2: If the estimated survival time is greater than the censored time, then it falls into the range

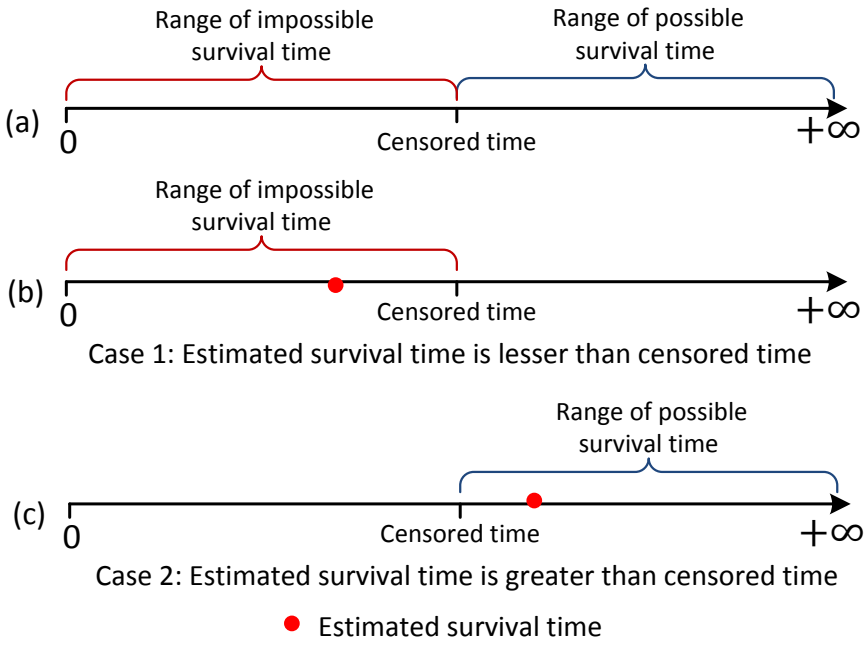


Figure 3.1: Relationship between estimated survival time and censored time for a right censored observation.

of possible survival time (Figure 3.1(c)), and hence, in the loss function, the model should assign less weight for this case.

Motivated by this observation, in this chapter, we propose a *Regularized Weighted Residual Sum-of-Squares (RWRSS) algorithm* which imposes more penalty to the first case and less penalty to the second case [46]. Thus, the proposed RWRSS is able to effectively handle the censored instances. Additionally, we also employ the elastic net as a penalty to sparsity the learned coefficients, so that the RWRSS is able to avoid overfitting and deal more efficiently with high-dimensional datasets. We propose a linear model because it is simple, effective and scalable. In survival analysis, linear regression for data analysis with censored observations is an alternative research direction that has attracted broad attention in the fields of data mining and biostatistics [12, 78, 80].

Some linear models such as Tobit regression [71] and Buckley-James (BJ) regres-

sion [12] were proposed to handle censored observations. These methods employ the Kaplan-Meier (KM) estimator [39] (in the case of BJ) and the Gaussian distribution (in the case of Tobit regression) to approximate the survival times of censored instances for satisfying the least-squares principle. However, these approximation methods will induce bias into the final model since the actual survival times of censored instances cannot be observed. It induces bias because the KM estimation cannot accurately estimate the survival time of censored instances, and this estimated inaccurate survival time will be used to train the model. This makes the prediction problem more complex because the survival time of censored instances are calculated using the integral of the KM estimator.

In contrast to these existing methods, in our work, we do not approximate the survival times of censored instances. RWRSS aims at directly minimizing the difference between the estimated survival time and the actual survival time of uncensored instances and ensures that the estimated survival time of censored instances is longer than the censored time. Thus, compared to the existing linear censored regression models, our proposed model simplifies the prediction problem. In this chapter, we demonstrate that such a simplification improves the prediction performance of the proposed model. The concordance index (C-index) [28] is the most commonly used performance metric in survival analysis which measures the concordance between the orderings of the survival times and the predicted marker values. *Since the actual survival time of censored instances are unknown, it is infeasible to calculate the concordance between a pair of censored instances. Hence, an accurate estimation of the survival time of censored instances is not possible or needed.*

From the viewpoint of traditional data mining, survival analysis can also be viewed as a semi-supervised learning problem, where the uncensored instances can be viewed as labeled data and the censored instances can be viewed as unlabeled data. However,

different from most of the existing semi-supervised algorithms which are focused on classification, survival analysis deals with a regression problem. Motivated by self-training [61], which uses the confidence estimated labels of unlabeled data in the next training round, we develop a framework which uses the proposed RWRSS model to infer the survival time of censored instances.

However, different from the traditional self-training approaches, in the first training round RWRSS is trained from both labeled (uncensored) and unlabeled (censored) instances. In the right censoring scenario, for certain censored instances, the censored time should be equal to or less than the survival time. Thus, once the estimated survival time is greater than the censored time, we will use the estimation to approximate the survival time of censored instances and train a new model based on the updated training instances in the next training round. Experimental results over high-dimensional biomedical datasets indicate that our model outperforms other related competing methods and attains very competitive C-index values on high-dimensional datasets.

The main contributions of this proposed work can be summarized as follows:

- Propose a novel weighted linear regression method, RWRSS, for prediction problems with censored observations which avoids the use of approximate survival time of censored data during the training phase.
- Develop a self-training framework which involves both uncensored and censored instances in each training round and is able to improve the prediction performance of the proposed RWRSS model.
- Demonstrate the performance of the proposed censored regression method using real-world high-dimensional cancer gene expression survival benchmark datasets and compare it with several existing survival estimation methods.

3.2 Regularized Weighted Linear Regression for Survival Analysis

In this section, we will explain the details of the proposed regularized weighted linear regression model for predicting survival times. We will first discuss the proposed weighted loss function along with its main intuition. Later, the regularized weighted linear regression and the optimization procedure will be explained in detail. Finally, a self-training framework for handling survival data with right censored instances will be discussed. This self-training framework is used with our regularized weighted linear regression as the base learning algorithm.

3.2.1 Objective Function

For censored observations, the exact difference between the estimated outcome and the actual target value cannot be measured. The estimated survival time for a right censored instance should be either equal to or larger than its censored time. For the i^{th} instance, if $\delta_i = 1$, then the estimated survival time of the proposed model should be as close as possible to $y_i = T_i = O_i$, and hence the standard squared residual can be used as loss function for uncensored instances; however, if $\delta_i = 0$, the estimated survival time should be greater than $y_i = T_i = C_i$, and hence, in the loss function, we should give more weight to the censored instances whose estimated survival time is lesser than censored time and less weight to the censored instances whose estimated survival time is greater than censored time. Thus, we propose the following weighted residual sum-of-squares (WRSS) as the objective function to minimize.

$$WRSS = \sum_{i=1}^{N} (y_i - X_i\beta)^2 w_i \tag{3.1}$$

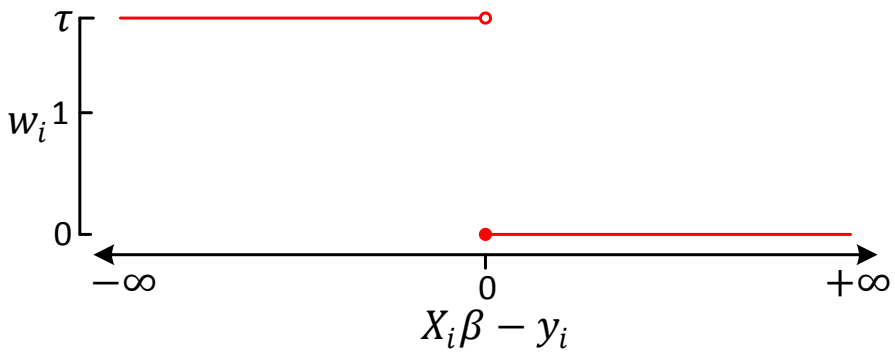


Figure 3.2: w_i is a step function for censored instances.

where weight w_i is defined as follows:

$$w_i = \begin{cases} 1 & \text{if} \quad \delta_i = 1 \\ \tau & \text{if} \quad \delta_i = 0 \text{ and } y_i \geq X_i\beta \\ 0 & \text{if} \quad \delta_i = 0 \text{ and } y_i < X_i\beta \end{cases} \tag{3.2}$$

From Eqs.(3.1) and (3.2), we can see that the WRSS calculates the standard residual value $(y_i - X_i\beta)$ for uncensored observations $(\delta_i = 1)$. However, for censored observations, it ignores the difference between the estimated output and the censored time when the estimated output is greater than the censored time. For the right censored observations, we know that the actual survival time is equal to or greater than the censored time; therefore, when the estimated output is lesser than the censored time, then the difference between the actual survival time and the estimated survival time is indeed greater than $(y_i - X_i\beta)$. Hence, τ is a constant which should be greater than 1 and is a parameter of the model that needs to be empirically determined, because it is infeasible to measure the true difference between the estimated output and the actual survival time of the corresponding censored instance.

In our proposed model, the elastic net [86] is used as the penalty term. The

corresponding optimization problem is formulated as follows:

$$arg\min_{\beta} \frac{1}{N}\sum_{i=1}^{N}(y_i - X_i\beta)^2 w_i + \lambda\left(\alpha\|\beta\|_1 + \frac{1-\alpha}{2}\|\beta\|_2^2\right) \tag{3.3}$$

where $\lambda \geq 0$ is the regularization parameter, and $0 \leq \alpha \leq 1$ is used to adjust the weights of the L_1 and L_2 norm penalties.

3.2.2 Optimization

From Eq. (3.2), we can see that w_i is a constant when $\delta_i = 1$ and it is a step function when $\delta_i = 0$ (see Figure 3.2). To handle this discontinuity in Eq. (3.3) due to the presence of the step function, we propose an iterative optimization method based on coordinate descent method. Coordinate descent minimizes a multi-variate function by minimizing it along only one direction at a time. Let us assume, except β_k, all other $\tilde{\beta}_l\,(l = 1, 2, 3, \cdots, p \text{ and } l \neq k)$ have already been estimated, and we would like to partially optimize with respect to β_k. The coordinate-wise updating [25] is done as follows:

$$\tilde{\beta}_k \leftarrow \frac{S(\frac{1}{N}\sum_{i=1}^{N} w_i x_{ik}(y_i - \tilde{y}_{i\bar{k}}),\ \lambda\alpha)}{\frac{1}{N}\sum_{i=1}^{N} w_i x_{ik}^2 + \lambda(1-\alpha)} \tag{3.4}$$

where $\tilde{y}_{i\bar{k}} = \sum_{l\neq k} x_{il}\tilde{\beta}_l$ is the fitted value excluding the contribution from x_{ik}, $S(Z,\ \gamma) = sign(Z)\cdot(|Z|-\gamma)_+$ is the soft-thresholding operation; in addition, $sign(\cdot)$ is the signum function, and $(|Z| - \gamma)_+$ refers to the positive part, which is $|Z| - \gamma$ if $(|Z| - \gamma) > 0$ and 0 otherwise. This update is simply the univariate regression coefficient of the partial residual sum of squares $(y_i - \tilde{y}_{i\bar{k}})$ on the k^{th} variable. In each iteration, all of the p coefficient variables are repeatedly updated until convergence.

Now, one of the problems that arise during this optimization is the determination of w_i for each observation in each iteration. From Eq. (3.2), we can see that the value of w_i is determined by δ_i, y_i, and $X_i\beta$, where for a particular i^{th} observation, δ_i, y_i, and X_i will not change during the coordinate decent process, but each element of β

will be updated one by one based on the coordinate-wise method. In the optimization process, we use the latest updated coefficient vector to approximate β; thus, in the d^{th} iteration before updating the coefficient of k^{th} feature, the latest updated $\beta^{d,k-1}$ can be represented by

$$\beta^{(d,k-1)} = \{\beta_1^d, \ldots, \beta_{(k-1)}^d, \beta_k^{d-1}, \beta_{(k+1)}^{d-1}, \ldots, \beta_p^{d-1}\}$$

and we have

$$\begin{aligned} X_i\beta &\approx X_i\beta^{(d,k-1)} \\ &= X_i\beta^{(d,k-2)} + X_{i,k-1} \cdot \beta_{(k-1)}^d - X_{i,k-1} \cdot \beta_{(k-1)}^{d-1} \end{aligned} \tag{3.5}$$

Algorithm 1 outlines the basic learning methodology for the proposed RWRSS model. In line 1, we initialize the estimator of the parameter $\hat{\beta}$ to be the zero vector. In lines 4-7, we calculate the updated w_i for each training instance, and each element of the coefficient vector is updated using the coordinate-wise update in line 8. Eq.(3.5) can be updated in $O(1)$ based on the previous result; thus, for N instances, a complete cycle costs $O(Np)$ operations where p is the number of features. Hence, the overall time complexity of the proposed model is $O(Np)$.

3.2.3 Theoretical Analysis

We will now provide the convergence analysis of Algorithm 1. Since w_i is updated iteratively based on the approximation made in Eq.(3.5) (line 6 of Algorithm 1), the proposed objective function is an iteratively re-weighted least squares (IRLS) which is different from the standard weighted update of coordinate descent. Thus, the analysis of the descent property is needed to ensure that the proposed RWRSS algorithm indeed converges.

Algorithm 1: Regularized weighted residual sum-of-squares (RWRSS)

Input: Training data (X, δ, y), Regularization parameter λ, Adjustment Weight α

Output: $\hat{\beta}$

Initialize: $\hat{\beta} \leftarrow \mathbf{0}$;
repeat
  for $k = 1$ *to* p **do**
    for $i = 1$ *to* N **do**
      Calculate $X_i\beta$ using Eq.(3.5);
      Update w_i using Eq.(3.2);
    end
    $\tilde{\beta}_k \leftarrow \frac{S(\frac{1}{N}\sum_{i=1}^{N} w_i x_{ik}(y_i - \tilde{y}_{i\tilde{k}}),\, \lambda\alpha)}{\frac{1}{N}\sum_{i=1}^{N} w_i x_{ik}^2 + \lambda(1-\alpha)}$;
  end
  $\hat{\beta} \leftarrow \tilde{\beta}$;
until *Convergence of* β;

Lemma 1. *The optimum value of Eq.(3.3) with iteratively updated w_i is upper bounded by the optimum value of Eq.(3.3) with constant initial w_i (denoted by $w_i^{(0)}$).*

Proof. Since β is initialized by a zero vector, we have $X_i\beta = 0 \leq y_i$ for all i. Then based on Eq.(3.2), we have

$$w_i^{(0)} = \begin{cases} 1 & \text{if} \quad \delta_i = 1 \\ \tau & \text{if} \quad \delta_i = 0 \end{cases}$$

Let $w_i^{(f)}$ denote the final updated weight of i^{th} instance, then we have

$$w_i^{(0)} \geq w_i^{(f)} \text{ for all } i \tag{3.6}$$

Let $L(w_i^{(0)}, \hat{\beta}^{(0)})$ be the optimum value of Eq.(3.3) with constant initial w_i, where $\hat{\beta}^{(0)}$ is the corresponding learned coefficient. Similarly, let $L(w_i^{(f)}, \hat{\beta}^{(f)})$ be the optimum value of Eq.(3.3) with an iteratively updated w_i, where $\hat{\beta}^{(f)}$ is the corresponding

learned coefficient. Thus, we have

$$L(w_i^{(0)}, \hat{\beta}^{(0)}) \geq L(w_i^{(f)}, \hat{\beta}^{(0)}) \geq L(w_i^{(f)}, \hat{\beta}^{(f)})$$

where the first inequality is based on Eq.(3.6), and the second inequality is because $\hat{\beta}^{(f)}$ is the learned optimal coefficient with respect to $w_i^{(f)}$. Therefore, the optimum value of Eq.(3.3) with iteratively updated w_i is upper bounded. □

Theorem 1. *The objective function given in Eq.(3.3) converges during the learning process.*

Proof. In the d^{th} iteration of coordinate descent, the k^{th} coefficient is updated by

$$\beta_k^d \leftarrow \min_{\beta_k} L(w_i^{(d,k-1)}, \beta_1^d, \ldots, \beta_{(k-1)}^d, \beta_k, \beta_{(k+1)}^{d-1}, \ldots, \beta_p^{d-1}) \tag{3.7}$$

where $w_i^{(d,k-1)}$ is the latest updated weight of i^{th} instance based on $\beta^{(d,k-1)}$. Similarly, as discussed in Lemma (1) we have $w_i^{(0)} \geq w_i^{(d,k-1)}$, and

$$\begin{aligned}
&L(w_i^{(0)}, \beta_1^d, \ldots, \beta_{(k-1)}^d, \beta_k^{d(0)}, \beta_{(k+1)}^{d-1}, \ldots, \beta_p^{d-1}) \\
&\geq L(w_i^{(d,k-1)}, \beta_1^d, \ldots, \beta_{(k-1)}^d, \beta_k^{d(0)}, \beta_{(k+1)}^{d-1}, \ldots, \beta_p^{d-1}) \\
&\geq L(w_i^{(d,k-1)}, \beta_1^d, \ldots, \beta_{(k-1)}^d, \beta_k^d, \beta_{(k+1)}^{d-1}, \ldots, \beta_p^{d-1})
\end{aligned}$$

where $\beta_k^{d(0)}$ is the optimal value of the k^{th} coefficient in d^{th} iteration if the weight of the instances is initialized as $w_i^{(0)}$. Thus, we can say that in each step of the learning process, the value of the objective function is upper bounded by the constant weighted $(w_i^{(0)})$ objective function. Based on the convergence of coordinate descent we know that the value of constant weighted objective function is monotonically decreasing during the learning process and converges to an optimal value. *Therefore, the value of the objective function in Eq.(3.3) is upper bounded by a monotonic decreasing con-*

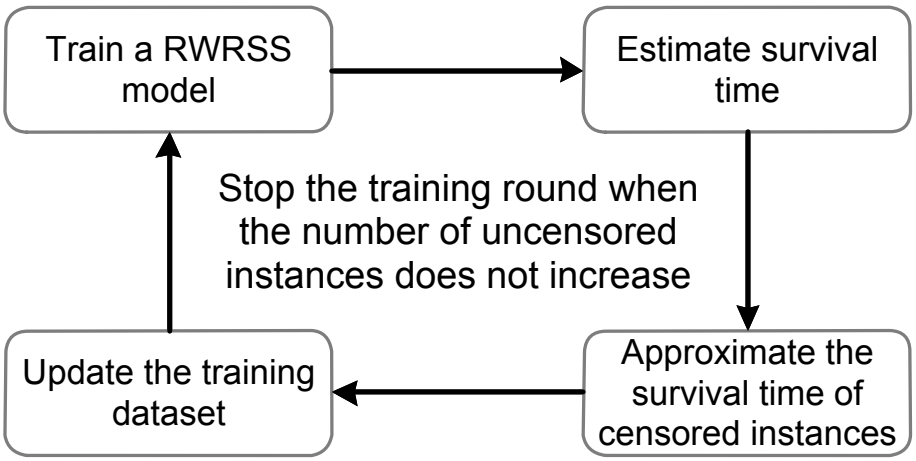


Figure 3.3: A self-training framework for right censored data.

vergence sequence, and we can conclude that the objective function converges during the learning process. □

3.2.4 Self-training Framework for Right Censored data

As pointed in the previous sections, mining from dataset with censored observations is closely related to semi-supervised learning. The censored observations can be considered as unlabeled instances since the event of interest can take place in the future. Most of the existing semi-supervised learning methods focus on classification rather than regression, and unlabeled observations do not contain any labeling information at all. However, in survival analysis, for each censored instance we can observe a lower bound of the target value. In this section, we propose a self-training framework for right censored data (STC). The goal of this framework is to infer the correct event labels for the given censored instances. In our proposed framework, we employ the RWRSS as our base learning method and label the censored instances based on both the prediction and the corresponding censored time.

Figure 3.3 shows the diagram of the proposed STC framework. In survival data, it is evident that the actual (unobserved) survival time of any right censored instance

should be no less than the observed censored time. Thus, in the t^{th} self-training learning round of the STC framework, if the estimated survival time of a censored instance is greater than or equal to the censored time, it will fall within the range of possible survival time, and it will be viewed as a correct prediction. Then in the $t+1^{th}$ learning round, this instance will be interpreted as an uncensored object whose survival time will be the estimated output in the t^{th} round and its corresponding censored indicator will be changed from 0 to 1. Therefore, in the $t+1^{th}$ learning round, the updated training data will contain more uncensored instances than the training data in the t^{th} round, and hence a robust model can be learned in the $t+1^{th}$ learning round.

Algorithm 2 describes the STC framework for right censored data. In line 3, we estimate the coefficients based on the proposed RWRSS algorithm. In lines 4-11, the objective values and the δ values for censored instances in the training dataset will be updated based on the predicted and the observed time. The training rounds will stop when the status (δ) and observed times (y) of the training data are not updated any further.

3.3 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.

3.3.1 Dataset Description

For our evaluation, we used several publicly available high-dimensional gene expression cancer survival benchmark datasets which can be downloaded from [1]. Here are the list of datasets that are used in our experiments.

- Norway/Stanford Breast Cancer Data (NSBCD).
- Van de Vijver's Microarray breast cancer (VDV).

[1] `http://user.it.uu.se/~liuya610/download.html`

Algorithm 2: Self-Training framework for right Censored Data (STC)

Input: Training data (X, δ, y), Regularization parameter λ, Adjustment weight α
Output: β

Initialize: $c \leftarrow 0$, $\hat{\beta} \leftarrow \mathbf{0}$;
repeat
  $\hat{\beta} = RWRSS(X, y, \delta, \lambda, \alpha)$;
  for $i = 1$ *to* N **do**
    if $\delta_i == 0$ **then**
      $\hat{y}_i = X_i\hat{\beta}$;
      if $\hat{y}_i > y_i$ **then**
        $y_i = \hat{y}_i$; $\delta_i = 1$;
      end
    end
  end
until δ *and* y *are not updated*;

- Lung adenocarcinoma (Lung).
- Mantle Cell Lymphoma (MCL) [2].
- The Dutch Breast Cancer Data (DBCD).
- Diffuse Large B-Cell Lymphoma (DLBCL).

All these datasets measure cancer survival using gene expression levels. Table 3.1 provides the details of the datasets that are being used. In this table, the column titled "# Censored" corresponds to the number of censored instances in each dataset. We used 5-fold cross validation when the number of instances is greater than 150 and 3-fold cross validation otherwise.

3.3.2 Evaluation Metrics

Concordance index (C-index) or the *concordance probability*, is used to measure the performance of prediction models in survival analysis [28]. 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

[2] `http://llmpp.nih.gov/MCL/`

Table 3.1: Details of the datasets used in this work.

Dataset	**# Instances**	**# Features**	**# Censored**
NSBCD	115	549	77
VDV	78	4705	44
Lung	86	7129	62
MCL	92	8810	28
DBCD	295	4919	216
DLBCL	240	7399	102

$\hat{y}_i$ is the predicted one. The concordance probability is defined as:

$$c = Pr(\hat{y}_1 > \hat{y}_2 | y_1 \geq y_2) \tag{3.8}$$

By definition, the C-index has the same scale as the 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 hazards ratio based regression models, the instances with a low hazard rate should survive longer, and the C-index will be calculated as follows:

$$c = \frac{1}{num} \sum_{i \in \{1 \cdots N\} \delta_i = 1} \sum_{y_j > y_i} I[X_i \hat{\beta} > X_j \hat{\beta}] \tag{3.9}$$

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

$$c = \frac{1}{num} \sum_{i \in \{1 \cdots N\} \delta_i = 1} \sum_{y_j > y_i} I[S(\hat{y}_j | X_j) > S(\hat{y}_i | X_i)] \tag{3.10}$$

where $S(\hat{y}_i | X_i)$ is the predicted target value for X_i.

3.3.3 Implementation Details

All of the seven methods used for comparisons are implemented in R. The Cox and Tobit regression models are obtained from the *survival* package [67]. In the *survival* package, the *coxph* function is employed to train the Cox model and the Efron's method [24] is used to handle the tied observations. The Tobit regression methods are trained using the *survreg* function with Gaussian distributions. Three sparse regression methods, namely, LASSO-COX, EN-COX, and EN-BJ, which are penalized versions using lasso and elastic net penalty terms are also used for our comparisons. LASSO-COX and EN-COX are built using the *cocktail* function in the *fastcox* package [82], while EN-BJ is implemented using the *bujar* package [80]. Boosting concordance index (BoostCI) [53] for survival data is an approach where the concordance index metric is modified to an equivalent smoothed criterion using the sigmoid function. In addition to the above mentioned six survival analysis methods, we also compared with the ordinary least squares (OLS) linear regression which has a similar form to the proposed methods. Note that, the OLS is only learned using the uncensored instances rather than the entire set of training instances since it cannot handle the censored instances, while the other methods are trained using both uncensored and censored instances.

In our experiments, we use the C-index of the training dataset as the training error rate to monitor the training round of the STC framework and terminate the learning round when the C-index decreases. In this scenario, the output of STC(RWRSS) is the model learned in the penultimate learning round.

3.3.4 Results and Discussion

Table 3.2 provides the C-index values obtained using various censored regression and OLS methods on the real-world high-dimensional cancer microarray datasets. The results show that our proposed model obtains higher C-index in most of the

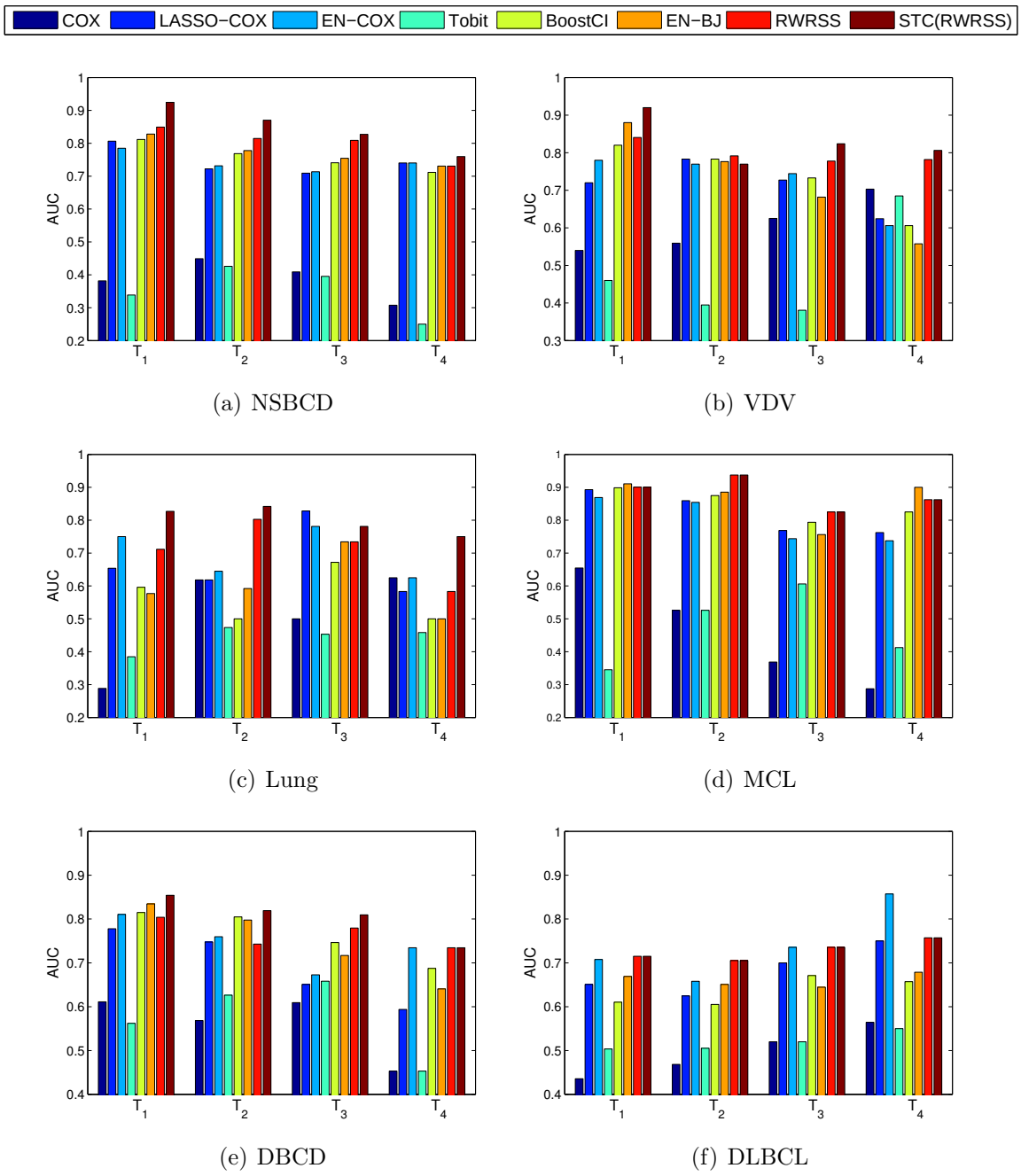
datasets and the STC framework is able to further improve the prediction performance of RWRSS in some of the cases.

Figure 3.4 provides the histogram plots of the AUC values for each dataset at four different time points corresponding to 25%, 50%, 75%, and 100% of events in each dataset. To demonstrate the time-dependent prediction capability of various survival analysis methods, the original problem has been reformulated into four classification problems which indicates the survival status of a patient at each time point. The prediction performance of each classifier is evaluated using AUC [13]; we exclude OLS model since it is not a censored regression method. The AUC values for our proposed models are higher than or close to those of the existing survival analysis methods indicating that the time-dependent prediction capability of our proposed models is higher than or as good as that of the other six survival models. It should be noted that the AUC values of the original RWRSS and the STC version of it are same for MCL and DLBCL datasets, because the STC framework did not improve the discriminative power of RWRSS for these two datasets, and hence the learning process is terminated after the second round. The AUC values of our proposed model on five datasets (NSBCD, VDV, Lung, MCL, and DBCD) are higher than or around 0.8, which indicates our proposed model is able to effectively predict the patient survival status at various time points.

Figure 3.5 presents the C-index values of the proposed RWRSS algorithm by varying the parameter τ from 1 to 3 in step of 0.2. We can see that the C-index values of all six datasets does not vary much when τ is greater than 1.6, and this phenomenon demonstrates that the RWRSS is not sensitive to the value of the parameter τ chosen. Note that the RWRSS does not reduce to the standard OLS model when τ equals to 1, because the weight of some censored instances is 0 according to Eq.(3.2).

Table 3.2: Performance comparison of the proposed methods and seven other existing related methods using C-index values (along with their standard deviation).

DataSet	**COX**	**LASSO-COX**	**EN-COX**	**BoostCI**	**OLS**	**Tobit**	**EN-BJ**	**RWRSS**	**STC(RWRSS)**
NSBCD	0.4411 (0.0589)	0.5910 (0.1086)	0.6046 (0.1000)	0.6263 (0.0831)	0.6333 (0.1108)	0.3733 (0.0214)	0.6215 (0.0924)	0.6766 (0.1277)	**0.7149** (**0.0836**)
VDV	0.5947 (0.0997)	0.6428 (0.0254)	0.6384 (0.0603)	0.6641 (0.0560)	0.5315 (0.0086)	0.5112 (0.1491)	0.6077 (0.0648)	0.7207 (0.0705)	**0.7445** (**0.0195**)
Lung	0.5139 (0.1372)	0.6684 (0.0867)	0.6639 (0.0661)	0.5708 (0.0883)	0.5716 (0.0610)	0.4695 (0.1321)	0.6634 (0.1284)	0.6969 (0.0430)	**0.7316** (**0.0313**)
MCL	0.5715 (0.0446)	0.6742 (0.0688)	0.6594 (0.0655)	0.6895 (0.0897)	0.4881 (0.0414)	0.4914 (0.0875)	0.7023 (0.1038)	**0.7118** (**0.0737**)	**0.7118** (**0.0737**)
DBCD	0.5294 (0.0634)	0.6850 (0.0417)	0.7188 (0.0304)	0.7045 (0.0380)	0.5599 (0.0717)	0.4869 (0.0784)	0.7175 (0.0396)	0.7216 (0.0446)	**0.7404** (**0.0475**)
DLBCL	0.5097 (0.0293)	0.6242 (0.0416)	**0.6372** (**0.0359**)	0.5954 (0.0170)	0.5052 (0.0891)	0.4917 (0.0524)	0.6228 (0.0611)	0.6265 (0.0657)	0.6265 (0.0657)



(a) NSBCD (b) VDV

(c) Lung (d) MCL

(e) DBCD (f) DLBCL

Figure 3.4: AUC values for different survival regression methods at different points of survival time. For each plot, T_1, T_2, T_3, and T_4 are the time points corresponding to the 25%, 50%, 75%, and 100% of events occurred, respectively.

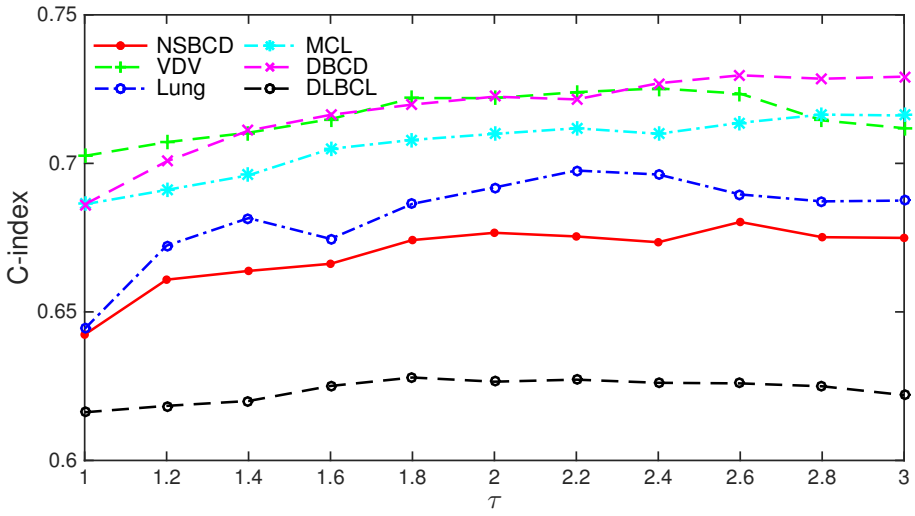


Figure 3.5: The effect of τ on C-index in the RWRSS algorithm.

CHAPTER 4 REGULARIZED PARAMETRIC REGRESSION

4.1 Motivation and Overview

Parametric regression is one of the fundamental tools in statistics and data analysis. In survival analysis, both the Cox proportional hazards model and parametric censored regression models are important foundational techniques for survival time prediction. Although not as widely studied as the Cox model, parametric censored regression has several advantages compared with the Cox model.

First, parametric censored regression models are more easily interpretable compared to the Cox model. In parametric censored regression, the probability of occurrence of an event of interest at a certain time is directly described by the density function of the selected distribution, and the probability of non-occurrence of the event of interest until a certain time is represented straightforwardly by a survival function. On the other hand, the Cox model does not model the probability of occurrence directly but learns it by maximizing the hazard ratio between the censored instances and their corresponding risk set.

Second, parametric censored regression is more efficient than the Cox model when tied observations (when survival times of multiple instances are exactly the same) occur during the study. Parametric censored regression can be directly used without any modification, while the Cox model has to use some approximation methods that suffer from either inducing bias (Breslow's approximation and Efron's approximation [24]) or bad scalability (Discrete method [67]).

To enable regularized parametric censored regression to handle high-dimensional censored datasets, in this chapter, we propose the "**URPCR**" model, which stands for "**U**nified model for **R**egularized **P**arametric **C**ensored **R**egression". Our proposed model unifies the learning process of regularized parametric censored regression with

different probability distributions; thus it improves the efficiency of model learning on an arbitrary probability distribution [48]. This efficiency is important because the performance of parametric censored regression is highly dependent on the choice of distribution.

In our proposed URPCR model, the elastic net is employed as the regularization term because it can both induce a sparse coefficient vector and handle correlated features. To unify the learning process of the proposed model with different distributions, we use a second-order Taylor expansion to approximate the log-likelihood; in this way, the URPCR model can be solved as a penalized iteratively reweighted least squares (IRLS). However, different from the standard linear model, a bias scale parameter has to be learned in addition to the coefficient vector in our proposed generalized linear model. Motivated by coordinate descent, in our learning scheme, this scale parameter is viewed as one coordinate and is iteratively updated based on Newton's method. Finally, the model is learned via a cyclical coordinate descent scheme.

In our empirical evaluation using several real-world high-dimensional cancer gene expression survival benchmark datasets, our model attains very competitive C-index values and outperforms most of the competing methods available in the literature of 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 various time points in the observed study period. This is accomplished by our URPCR model without the need to re-train a new classifier for each time point, which is one of the main advantages of this work.

4.2 Proposed Model

In this section, we will discuss the proposed URPCR model in detail, along with an efficient optimization approach to learn the model. The URPCR employs the basic

notion of generalized linear models (GLMs) and the framework of cyclical coordinate descent to solve the elastic net penalized parametric censored regression. Thus, the URPCR enables the parametric censored regressions to perform feature selection and handle high-dimensional data sets in survival analysis.

4.2.1 Objective Function

The URPCR aims at learning the relationship between the feature vectors and the target value in the same manner as done in the generalized linear model, which can be formulated as follows:

$$v = X\beta + \sigma\varepsilon, \quad \varepsilon \sim f \tag{4.1}$$

for some distribution f, where β is the coefficient vector, ε is the error term, and $\sigma > 0$ is an unknown bias scalar. For the i^{th} instance, v_i can either be the survival time ($v_i = O_i$) or the logarithm of the survival time ($v_i = \log(O_i)$). When $v_i = O_i$, Eq.(4.1) becomes an extended linear regression with a self-selected bias distribution; when $v_i = \log(O_i)$, then Eq.(4.1) represents an AFT model [81], which is a commonly used prediction method in survival analysis. Thus, the URPCR encompasses these two models within a unified framework, which can be solved with exactly the same learning process. This is the primary novel aspect of the proposed work.

Under this linear hypothesis, the likelihood function of all N instances can be represented as

$$L = \prod_{\delta_i=1} f(\varepsilon_i/\sigma) \prod_{\delta_i=0} (1 - F(\varepsilon_i)) \tag{4.2}$$

where $\varepsilon_i = \frac{y_i - X_i\beta}{\sigma}$, $y_i = T_i$ for extended linear regression, and $y_i = \log(T_i)$ for AFT model. The log-likelihood can be written in the form

$$ll = \sum_{\delta_i=1} g_1(\varepsilon_i) - \log(\sigma) + \sum_{\delta_i=0} g_2(\varepsilon_i) \tag{4.3}$$

where $g_1 = \log(f(\cdot))$ and $g_2 = \log(1 - F(\cdot))$.

To avoid overfitting the model, it may not be appropriate to build a prediction model that includes all of the features. This becomes even more important when the feature dimension (p) is either close to or larger than the sample size (N). Sparsity-inducing penalization is an effective method which can perform model estimation and feature selection simultaneously. The elastic net [86] is one of the most commonly used penalty terms in the data mining and machine learning communities and consists of a mixture of the L_1 (lasso) and L_2 (ridge regression) penalties. Therefore, it can obtain both sparsity in the coefficients and handle correlated feature spaces simultaneously. Mathematically, it is defined as follows:

$$P(\beta) = \alpha\|\beta\|_1 + \frac{1-\alpha}{2}\|\beta\|_2^2 \tag{4.4}$$

where $0 \leq \alpha \leq 1$ is used to adjust the weights of the L_1 and L_2 norm penalties. Hence, the Lagrangian of the penalized negative log-likelihood becomes

$$\min_{\beta,\sigma}\left[-\frac{2}{N}\left(\sum_{\delta_i=1} g_1(\varepsilon_i) - \log(\sigma_i) + \sum_{\delta_i=0} g_2(\varepsilon_i)\right) + \lambda P(\beta)\right] \tag{4.5}$$

where $\lambda \geq 0$ is the Lagrangian multiplier.

4.2.2 Optimization

To minimize the objective function proposed in Eq.(4.5), we use a second-order Taylor expansion to approximate the log-likelihood and a cyclical coordinate descent-based method, which solves a penalized iteratively reweighted least squares (IRLS) problem in each iteration [65], to solve the generalized linear model.

If we treat σ as fixed and let $\eta = X\beta$, a two-term Taylor series expansion of the

log-likelihood centered at $\tilde{\beta}$ has the following form.

$$\begin{aligned} ll(\beta) &\approx ll(\tilde{\beta}) + (\beta - \tilde{\beta})^T ll'(\tilde{\beta}) + \frac{(\beta-\tilde{\beta})^T ll''(\tilde{\beta})(\beta-\tilde{\beta})}{2} \\ &= ll(\tilde{\beta}) + (X\beta - \tilde{\eta})^T ll'(\tilde{\eta}) + \frac{(X\beta-\tilde{\eta})^T ll''(\tilde{\eta})(X\beta-\tilde{\eta})}{2} \end{aligned} \tag{4.6}$$

where $\tilde{\eta} = X\tilde{\beta}$; $ll'(\tilde{\beta})$, $ll''(\tilde{\beta})$, $ll'(\tilde{\eta})$, and $ll''(\tilde{\eta})$ denote the gradient and Hessian of the log-likelihood with respect to $\tilde{\beta}$ and $\tilde{\eta}$, respectively. By some simple algebra, we obtain

$$ll(\beta) \approx \frac{1}{2}(z(\tilde{\eta}) - X\beta)^T ll'(\tilde{\eta})(z(\tilde{\eta}) - X\beta) + C(\tilde{\eta}, \tilde{\beta}), \tag{4.7}$$

where $z(\tilde{\eta}) = \tilde{\eta} - ll'(\tilde{\eta})/ll''(\tilde{\eta})$ is the adjusted dependent variable, and $C(\tilde{\eta}, \tilde{\beta})$ is a constant term that does not depend on β. To speedup the algorithm, rather than using the full $N \times N$ $ll''(\tilde{\eta})$ matrix, we use the diagonal elements of $ll''(\tilde{\eta})$ in our algorithm. The i^{th} diagonal element is denoted as $ll''(\tilde{\eta})_i$. We define $z(\tilde{\eta})_i = \tilde{\eta}_i - ll'(\tilde{\eta})_i/ll''(\tilde{\eta})_i$. Therefore, Eq.(4.5) can be simplified as a penalized IRLS:

$$\min_{\beta} -\frac{1}{N}\sum_{i=1}^{N} ll''(\tilde{\eta})_i(z(\tilde{\eta})_i - X_i\beta)^2 + \lambda P(\beta) \tag{4.8}$$

The partial derivative of the IRLS with respect to the k^{th} coordinate, $k = 1, 2, \cdots, p$, can be calculated as:

$$-\frac{1}{N}\sum_{i=1}^{N} ll''(\tilde{\eta})_i x_{ik}(z(\tilde{\eta})_i - X_i\beta) + \lambda\alpha \cdot sgn(\beta_k) + \lambda(1-\alpha)\beta_k \tag{4.9}$$

where $sgn(\cdot)$ is the signum function. Hence, the coordinate-wise update of the penalized IRLS will take the following form:

$$\hat{\beta}_k = \frac{S(-\frac{1}{N}\sum_{i=1}^{N} ll''(\tilde{\eta})_i x_{ik}(z(\tilde{\eta})_i - \sum_{j\neq k} x_{ij}\beta_j), \lambda\alpha)}{-\frac{1}{N}\sum_{i=1}^{N} ll''(\tilde{\eta})_i x_{ik}^2 + \lambda(1-\alpha)} \tag{4.10}$$

where $S(Z,\ \gamma) = sgn(Z) \cdot (|Z| - \gamma)_+$ is the soft-thresholding operation, and $ll^{'}(\tilde{\eta})_i$ and $ll^{''}(\tilde{\eta})_i$ can be calculated as follows:

$$ll^{'}(\tilde{\eta})_i = \begin{cases} \frac{\partial g_1}{\partial \tilde{\eta}_i} = -\frac{1}{\sigma} \cdot \frac{f^{'}(\varepsilon_i)}{f(\varepsilon_i)} & \text{if} \quad \delta_i = 1 \\ \frac{\partial g_2}{\partial \tilde{\eta}_i} = -\frac{1}{\sigma} \cdot \frac{-f(\varepsilon_i)}{1-F(\varepsilon_i)} & \text{if} \quad \delta_i = 0 \end{cases}$$

$$ll^{''}(\tilde{\eta})_i = \begin{cases} \frac{\partial^2 g_1}{\partial \tilde{\eta}_i^2} = \frac{1}{\sigma^2} \cdot \frac{f^{''}(\varepsilon_i)}{f(\varepsilon_i)} - \left(\frac{\partial g_1}{\partial \tilde{\eta}_i}\right)^2 & \text{if} \quad \delta_i = 1 \\ \frac{\partial^2 g_2}{\partial \tilde{\eta}_i^2} = \frac{1}{\sigma^2} \cdot \frac{-f^{'}(\varepsilon_i)}{1-F(\varepsilon_i)} - \left(\frac{\partial g_2}{\partial \tilde{\eta}_i}\right)^2 & \text{if} \quad \delta_i = 0 \end{cases}$$

where $f(\cdot)$ is the density function of the selected distribution, $F(\cdot)$ is the corresponding cumulative distribution function, and $f^{'}(\cdot)$ and $f^{''}(\cdot)$ denote the gradient and Hessian of the density function [67], respectively. In this work, we choose the Gaussian distribution, Logistic distribution, and Extreme value distribution as the baseline distributions. It should be noted that, besides these three distributions that are being described in this work, our framework is suitable for all other parametric distributions once the corresponding functions for the distributions are calculated.

All the analysis until this point has assumed a fixed σ. We will also vary the value of σ in our learning scheme by making σ another coordinate that is updated once all of the coefficient variables are updated. In the proposed algorithm, we use the Newton-Raphson method to update $\log \sigma$, which can be written in the following form:

$$\log \sigma = \log \tilde{\sigma} - ll^{'}(\log \tilde{\sigma}) / ll^{''}(\log \tilde{\sigma}) \tag{4.11}$$

where $\tilde{\sigma}$ is learned in the previous iteration, $ll^{'}(\log \tilde{\sigma}) = \frac{1}{N} \sum ll^{'}(\log \tilde{\sigma})_i$, and $ll^{''}(\log \tilde{\sigma}) =$

$\frac{1}{N}\sum ll^{''}(\log\tilde{\sigma})_i$. Additionally, $ll^{'}(\log\tilde{\sigma})_i$ and $ll^{''}(\log\tilde{\sigma})_i$ can be calculated as follows:

$$ll^{'}(\log\tilde{\sigma})_i = \begin{cases} \frac{\partial g_1}{\partial \log\tilde{\sigma}_i} = -\frac{\varepsilon_i f^{'}(\varepsilon_i)}{f(\varepsilon_i)} & \text{if} \quad \delta_i = 1 \\ \frac{\partial g_2}{\partial \log\tilde{\sigma}_i} = -\frac{-\varepsilon_i f(\varepsilon_i)}{1-F(\varepsilon_i)} & \text{if} \quad \delta_i = 0 \end{cases}$$

$$ll^{''}(\log\tilde{\sigma})_i = \begin{cases} \frac{\partial^2 g_1}{\partial(\log\tilde{\sigma}_i)^2} = \frac{\varepsilon_i^2 f^{''}(\varepsilon_i)+\varepsilon_i f^{'}(\varepsilon_i)}{f(\varepsilon_i)} - \left(\frac{\partial g_1}{\partial \log\tilde{\sigma}_i}\right)^2 & \text{if} \quad \delta_i = 1 \\ \frac{\partial^2 g_2}{\partial(\log\tilde{\sigma}_i)^2} = \frac{-\varepsilon_i^2 f^{'}(\varepsilon_i)}{1-F(\varepsilon_i)} - \frac{\partial g_1}{\partial \log\tilde{\sigma}_i}\cdot\left(1+\frac{\partial g_1}{\partial \log\tilde{\sigma}_i}\right) & \text{if} \quad \delta_i = 0 \end{cases}$$

Good initial values of the coefficients and σ turn out to be vital for successful optimization, especially in a high-dimensional data set. In coordinate descent, the coefficient vector usually starts with the zero vector because the L_1 norm penalty induces lot of zero elements in the coefficient vector. For σ, a clever starting point is introduced in [67], where the model is fit starting with the mean and variance of each feature. As the normalization makes the values of each feature and the target value in the dataset have zero mean and unit variance, the initial values of the iteration only depend on the mean and variance of the selected distribution, denoted as μ and s^2, respectively. The μ and s^2 of the selected baseline distributions can be summarized as follows:

- **Gaussian distribution**:

$$\begin{array}{ll} \mu = 0 & s^2 = 1 \\ F(x) = \Phi(x) & f(x) = \frac{1}{\sqrt{2\pi}} exp(-x^2/2) \\ f^{'}(x) = -xf(x) & f^{''}(x) = (x^2-1)f(x) \end{array}$$

- **Logistic distribution**:

$$\mu = 0 \qquad s^2 = \pi^2/3$$
$$F(x) = \frac{e^x}{1+e^x} \qquad f(x) = \frac{e^x}{(1+e^x)^2}$$
$$f^{'}(x) = f(x) \cdot \frac{1-e^x}{1+e^x} \quad f^{''}(x) = f(x) \cdot \frac{(e^x)^2-4e^x+1}{(1+e^x)^2}$$

- **Extreme value distribution**:

$$\mu = 0.5722 \qquad s^2 = \pi^2/6$$
$$F(x) = 1 - \exp(-e^x) \quad f(x) = e^x \cdot \exp(-e^x)$$
$$f^{'}(x) = (1-e^x) \cdot f(x) \quad f^{''}(x) = [(e^x)^2 - 3e^x + 1] \cdot f(x)$$

4.2.3 The URPCR Algorithm

Algorithm 3 outlines the overall steps involved in the proposed model including the main optimization method. In lines 1-5, the dependent variable is calculated based on the user setting. In line 6, the coefficient vector and $\hat{\sigma}$ are initialized by a zero vector and s (the standard deviation of the selected distribution), respectively. In lines 9-14, the weight and adjusted dependent variables of the IRLS for each training instance are calculated. In line 15, one coordinate is updated based on coordinate descent. In lines 17-21, the updated formulas are calculated after all the p coefficients have been updated. Finally, in lines 22-24, the σ is updated based on these new updated equations. Note that, at each time, only one variable in the vector $\tilde{\beta}$ is being updated, and hence $\tilde{\eta}_i$ can be updated based on the previous iteration's result in O(1). Thus, one complete cycle of coordinate descent through all p variables costs O(Np) operations, and the σ can be updated in O(N) operations. Hence the total computation cost for each optimization step of the proposed algorithm is O(Np).

Usually, in the learning process, the model has to be trained based on a series of

Algorithm 3: URPCR Algorithm

Input: Training data (X, T, δ), Regularization parameter λ, Adjustment Weight α, Selected Distribution, flag AFT

Output: $\hat{\beta}$, $\hat{\sigma}$

if $AFT{==}TRUE$ **then**
  $y = \log(T)$;
else
  $y = T$;
end
Initialize: $\hat{\beta} \leftarrow \mathbf{0}$, $\hat{\sigma} \leftarrow s$;
repeat
  for $k = 1$ *to* p **do**
    for $i = 1$ *to* N **do**
      Calculate $\tilde{\eta}_i = X_i\tilde{\beta}$, $\varepsilon_i = \frac{y_i - \tilde{\eta}_i}{\tilde{\sigma}}$;
      Calculate $f(\varepsilon_i)$, $F(\varepsilon_i)$, $f^{'}(\varepsilon_i)$, and $f^{''}(\varepsilon_i)$;
      Calculate $ll^{'}(\tilde{\eta})_i$ and $ll^{''}(\tilde{\eta})_i$;
      Update $z(\tilde{\eta})_i = \tilde{\eta}_i - ll^{'}(\tilde{\eta})_i / ll^{''}(\tilde{\eta})_i$;
    end
    $\tilde{\beta}_k \leftarrow \frac{S(-\frac{1}{N}\sum_{i=1}^{N} ll^{''}(\tilde{\eta})_i x_{ik}(z(\tilde{\eta})_i - \sum_{j \neq k} x_{ij}\tilde{\beta}_j), \lambda\alpha)}{-\frac{1}{N}\sum_{i=1}^{N} ll^{''}(\tilde{\eta})_i x_{ik}^2 + \lambda(1-\alpha)}$;
  end
  for $i = 1$ *to* N **do**
    Calculate $\tilde{\eta}_i = X_i\tilde{\beta}$, $\varepsilon_i = \frac{y_i - \tilde{\eta}_i}{\tilde{\sigma}}$;
    Calculate $f(\varepsilon_i)$, $F(\varepsilon_i)$, $f^{'}(\varepsilon_i)$, and $f^{''}(\varepsilon_i)$;
    Calculate $ll^{'}(\log\tilde{\sigma})_i$ and $ll^{''}(\log\tilde{\sigma})_i$;
  end
  Calculate $ll^{'}(\log\tilde{\sigma})$ and $ll^{''}(\log\tilde{\sigma})$;
  Update $\log\sigma$ based on Eq.(4.11);
  $\tilde{\sigma} = exp(\log\sigma)$;
until *Convergence of* $\tilde{\beta}$ *and* $\tilde{\sigma}$;
$\hat{\beta} \leftarrow \tilde{\beta}$, $\hat{\sigma} \leftarrow \tilde{\sigma}$;

values for λ, and the best λ is selected via cross-validation. In this work, we build a pathwise solution similar to the approach given in [65]; initialize λ to a sufficiently large number, which forces β to a zero vector, and then gradually decrease λ in each learning iteration. For a new λ, the initial values of β and σ are the estimated β and σ learned from the previous λ as a warm start, so the initial values of β and σ are not far from the optimal value, and the algorithm can converge in few iterations. The

convergence of the Newton step in the algorithm is not guaranteed; it may become unstable if the initial parameter is far from the optimal value. However, in a pathwise solution, the warm start is not far from the optimal value, so it solves the convergence problem to a large extent.

4.3 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.

4.3.1 Dataset Description

For our evaluation, we used several publicly available high-dimensional gene expression cancer survival benchmark datasets[1]. The datasets we used in our experiments are as follows:

- The Norway/Stanford Breast Cancer Data (NSBCD) contains gene expression measurements of 115 women with breast cancer. The missing values are imputed using 10-nearest neighbor imputation (which is a common practice in the biomedical domain).
- Lung adenocarcinoma (Lung) is a dataset containing observations of 86 early-stage lung adenocarcinoma patients.
- The Dutch Breast Cancer Data (DBCD) contains information on 4919 gene expression levels of a series of 295 women with breast cancer. Measurements were taken from the fresh-frozen-tissue bank of the Netherlands Cancer Institute.
- Diffuse Large B-Cell Lymphoma (DLBCL) is a dataset that contains Lymphochip DNA microarrays from 240 biopsy samples of DLBCL tumors.

All of these datasets measure cancer survival using gene expression levels. Table 4.1 provides the details of the datasets that are being used in this work. In this table, the column titled "# Censored" corresponds to the number of censored instances in

[1] `http://user.it.uu.se/~liuya610/download.html`

Table 4.1: Details of the datasets used in this work.

Dataset	**# Instances**	**# Features**	**# Censored**
NSBCD	115	549	77
Lung	86	7129	62
DBCD	295	4919	216
DLBCL	240	7399	102

each dataset. We used 5-fold cross validation when the number of instances is greater than 150 and 3-fold cross validation otherwise.

4.3.2 Implementation Details

The proposed model is implemented using C++ with the Eigen library[2], and in each iteration, the weight updates for all N instances (lines 9-14 and lines 17-21 of Algorithm 3) are calculated in parallel.

All of the methods used in our comparisons are implemented in R. The Cox and unregularized parametric censored regression are obtained from the *survival* package [67]. In the *survival* package, the *coxph* function is employed to train the Cox model. The Tobit regression is trained using the *survreg* function. The parametric censored regressions are trained using the *survreg* function with Normal, Log-normal, Logistic, Log-logistic, and Weibull distributions. Three sparse regression methods, namely, LASSO-COX, EN-COX, and EN-BJ, which are penalized versions using lasso and elastic net penalty terms, are also used for our comparisons. LASSO-COX and EN-COX are built using the *cocktail* function in the *fastcox* package [82], while EN-BJ is implemented using the *bujar* package [80].

Boosting concordance index (BoostCI) [53] for survival data is an approach where the concordance index metric is modified to an equivalent smoothed criterion using the sigmoid function. In addition to the above survival methods, we also compared our methods with ordinary least squares (OLS) because URPCR is a generalized

[2] `http://eigen.tuxfamily.org/`

linear model. In our experiments, the Gaussian distribution, Logistic distribution, and Extreme value distribution are chosen as the baseline distributions. For each dataset, the validation data is used to select the appropriate distribution and to decide whether the dependent variable y should be the observed time T or the logarithm of the observed time $\log(T)$.

4.3.3 Results and Discussion

Table 4.2 provides the C-index values obtained by various regression methods on the real-world high-dimensional micro-array cancer datasets. The results show that our proposed URPCR model obtains higher C-index in most of the datasets.

Table 4.3 provides the C-index values obtained from the original censored regression and the URPCR regularized parametric censored regression methods based on different distributions, where Log-normal and Log-logistic denote that the logarithm of the observed time is assumed to follow the normal distribution and logistic distribution, respectively. It should be noted that Weibull distribution is a special case of the generalized extreme value distribution. The results show that, with sparsity-inducing penalization, our proposed model is able to improve the prediction performance of the parametric censored regression on the high-dimensional datasets for different kinds of distributions.

Figure 4.1 provides histogram plots of the AUC values for the binary classification task on each dataset with four different time splits corresponding to the time points when 25%, 50%, 75%, and 100% of events have occurred. The AUC values for our proposed models are higher than those of the existing survival prediction methods in all but one task, which further reinforces the accuracy of our proposed model compared to the other survival prediction methods; we exclude OLS in the plots since it is not a survival regression method. These results demonstrate that our proposed model is able to predict temporal event occurrence at different time points

effectively without the need to re-train a new classifier at each time point.

Table 4.2: Performance comparison of the proposed URPCR method and seven other existing related methods using C-index values (along with their standard deviations).

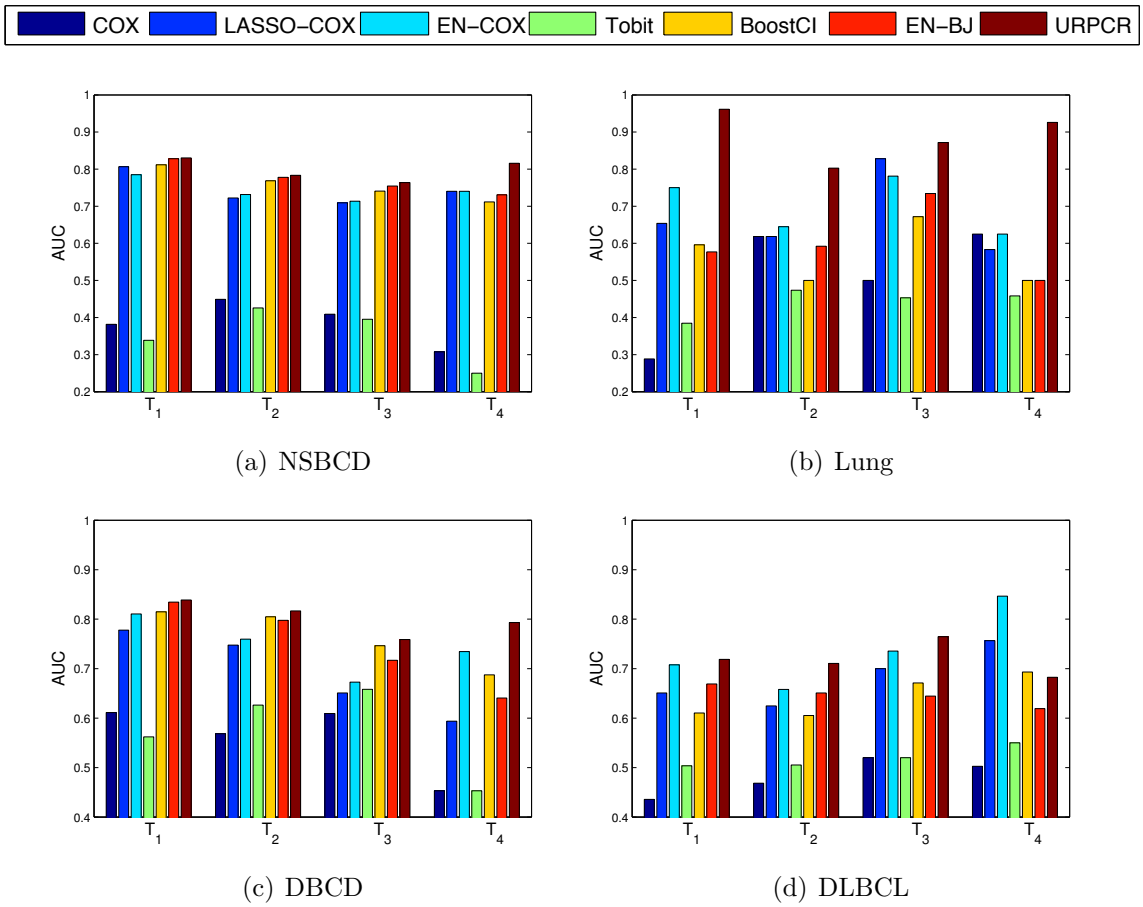
DataSet	**COX**	**LASSO-COX**	**EN-COX**	**BoostCI**	**OLS**	**Tobit**	**EN-BJ**	**URPCR**
NSBCD	0.441	0.591	0.605	0.626	0.633	0.373	0.622	**0.693**
	(0.059)	(0.109)	(0.100)	(0.083)	(0.111)	(0.021)	(0.092)	(**0.056**)
Lung	0.514	0.668	0.664	0.571	0.572	0.470	0.663	**0.771**
	(0.137)	(0.087)	(0.066)	(0.088)	(0.061)	(0.132)	(0.128)	(**0.039**)
DBCD	0.529	0.685	0.719	0.705	0.560	0.487	0.718	**0.735**
	(0.063)	(0.042)	(0.030)	(0.038)	(0.072)	(0.078)	(0.040)	(**0.027**)
DLBCL	0.510	0.624	**0.637**	0.595	0.505	0.492	0.623	0.631
	(0.029)	(0.042)	(**0.036**)	(0.017)	(0.089)	(0.052)	(0.061)	(0.056)

Table 4.3: Performance comparison of the proposed regularized censored regressions and unregularized censored regressions with different distributions using C-index values (along with their standard deviations).

	Normal		Log-normal		Logistic		Log-logistic		Weibull	
	original	URPCR	original	URPCR	original	URPCR	original	URPCR	original	URPCR
NSBCD	0.373 (0.021)	**0.667** (**0.065**)	0.444 (0.054)	**0.682** (**0.039**)	0.379 (0.020)	**0.693** (**0.056**)	0.238 (0.050)	**0.667** (**0.042**)	0.304 (0.153)	**0.688** (**0.074**)
Lung	0.470 (0.132)	**0.736** (**0.028**)	0.411 (0.075)	**0.712** (**0.020**)	0.566 (0.095)	**0.771** (**0.039**)	0.587 (0.066)	**0.762** (**0.041**)	0.428 (0.101)	**0.762** (**0.068**)
DBCD	0.487 (0.078)	**0.716** (**0.030**)	0.491 (0.057)	**0.735** (**0.027**)	0.490 (0.088)	**0.721** (**0.063**)	0.527 (0.025)	**0.723** (**0.059**)	0.458 (0.104)	**0.708** (**0.036**)
DLBCL	0.492 (0.052)	**0.626** (**0.057**)	0.320 (0.078)	**0.625** (**0.056**)	0.491 (0.044)	**0.498** (**0.279**)	0.431 (0.125)	**0.581** (**0.099**)	0.396 (0.084)	**0.631** (**0.056**)

4.3.4 Scalability Experiments

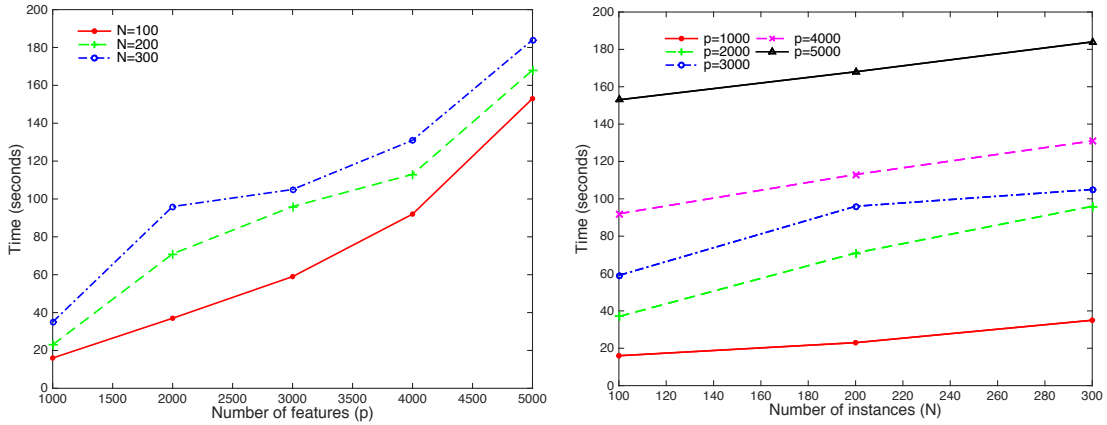
We also empirically evaluate the scalability of the proposed algorithm with respect to sample size (N) and the number of features (p). All synthetic datasets are generated using the function "simple.surv.sim" in *survsim* package [55] with different sample sizes and feature dimensionality. All the features are generated based on the uniform distribution, and each of them have a different randomly set interval. The coefficient vector is also randomly generated and remain within $[-1, 1]$. The observed time is assumed to follow a Log-logistic distribution, and time to censorship follows a Weibull distribution. All timing calculations are carried out on an Intel Xeon 3 GHz processor with 16 cores (32 threads). Figure 4.2(a) shows runtimes for fixed N and varying p, and Figure 4.2(b) shows runtimes for fixed p and varying N. These two plots suggest that the runtime of URPCR is close to being *linear in both N and p*. Notice that the lines in Figure 4.2(b) increase more slowly than the lines in Figure 4.2(a), which indicates that our proposed URPCR model has better scalability with respect to N than with respect to p. This is because, in our implementation, the weight updates of all N instances (lines 9-14 and lines 17-21 of Algorithm 3) are calculated in parallel.



(a) NSBCD (b) Lung

(c) DBCD (d) DLBCL

Figure 4.1: AUC values for binary classification of survival times for four different time thresholds. The URPCR is compared to six different survival regression methods. For each plot, T_1, T_2, T_3, and T_4 are the time thresholds corresponding to the timepoints at which 25%, 50%, 75%, and 100% of events have occurred, respectively.



(a) Scalability w.r.t. p (b) Scalability w.r.t. N

Figure 4.2: *Scalability results*: Plots of the runtimes of URPCR with the extreme value distribution. The times denote total runtimes for ten λ values averaged over five trials.

CHAPTER 5 A MULTI-TASK LEARNING FORMULATION

5.1 Motivation and Overview

In survival analysis, the Cox proportional hazards model and parametric censored regression models are important foundational techniques for survival time prediction. These two methods (and their extensions) have been extensively studied in the fields of statistical learning and data mining. However, they are both 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, which have been discussed in detail in the introduction of this thesis, we reformulate the survival analysis problem as a multi-task learning problem and propose a new multi-task learning based formulation, MTLSA, to predict the survival time by estimating the survival status at each time interval during the study duration.

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 learn from both uncensored and censored instances (details can be found in Section 5.2.1) 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 then 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) [8] algorithm for coefficient estimation. In the proposed ADMM method, the *non-negative non-increasing list* constraint optimization has been transformed into an Euclidean projection problem that can be learned efficiently [47].

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.

5.2 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.

5.2.1 Transform to multi-task learning problem

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 trans-*

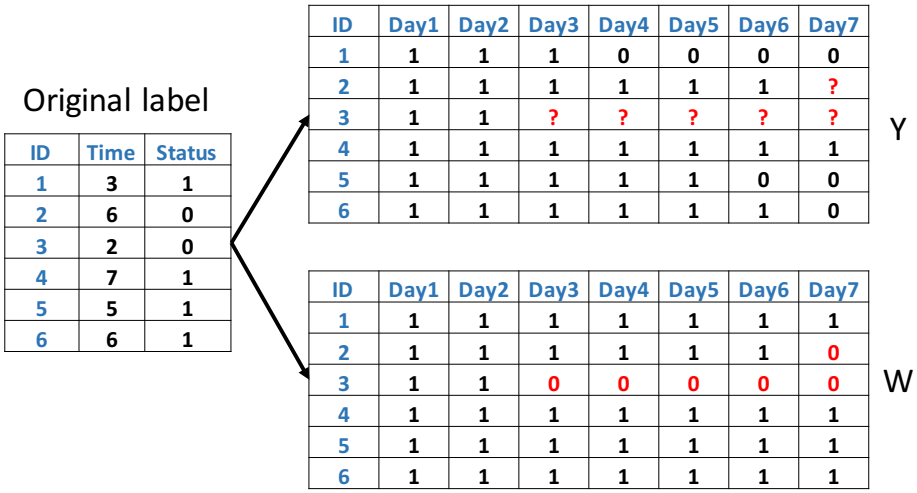


Figure 5.1: Illustration of generating Y and W from the original label in a simple survival dataset.

forming 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 5.1 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.

In this work, we only focus on 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 cannot change back to "1". Thus, we can see that each row of Y will have a *non-negative non-increasing list* structure:

$$P = \{Y \geq 0, Y_{ij} \geq Y_{il} | j \leq l, \forall j = 1, \cdots, k, \forall l = 1, \cdots, k\} \tag{5.1}$$

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

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

$$\underset{XB \in P}{\text{minimize}} \quad \frac{1}{2} \parallel Y - XB \parallel_F^2 + R(B) \tag{5.2}$$

where $B \in \mathbb{R}^{m \times k}$ 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 5.1, we also show the corresponding W for the example survival data considered. The optimization problem in Eq.(5.2) is re-defined as:

$$\underset{XB \in P}{\text{minimize}} \quad \frac{1}{2} \parallel \Pi_W(Y - XB) \parallel_F^2 + R(B) \tag{5.3}$$

where

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

5.2.2 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.(5.3) 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.(5.3) 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 work, the non-negative max-heap projection [51] 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 (please refer to the section 5.2.4 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 [2]. 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:

$$\underset{XB \in P}{\text{minimize}} \quad \frac{1}{2} \parallel \Pi_W(Y - XB) \parallel_F^2 + \frac{\lambda_1}{2} \parallel B \parallel_F^2 + \lambda_2 \parallel B \parallel_{2,1} \tag{5.4}$$

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

5.2.3 The proposed MTLSA algorithm

The solution of the optimization problem proposed in Eq.(5.4) 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.(5.4). By introducing a new matrix $M = XB$, Eq.(5.4) can be rewritten in ADMM form as

$$\begin{aligned} &\underset{M \in P}{\text{minimize}} \quad \frac{1}{2} \parallel \Pi_W(Y - M) \parallel_F^2 + \frac{\lambda_1}{2} \parallel B \parallel_F^2 + \lambda_2 \parallel B \parallel_{2,1} \\ &\text{subject to} \quad M = XB \end{aligned} \tag{5.5}$$

Using the scaled dual variable μ and penalty parameter $\rho > 0$, the resulting augmented Lagrangian of Eq.(5.5) is

$$L_\rho(M, B, \mu) = \frac{1}{2} \parallel \Pi_W(Y - M) \parallel_F^2 + \frac{\lambda_1}{2} \parallel B \parallel_F^2 + \lambda_2 \parallel B \parallel_{2,1} + \frac{\rho}{2} \parallel M - XB + \mu \parallel_F^2 \tag{5.6}$$

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

$$M^{t+1} := \arg\min_{M \in P} \left(\frac{1}{2} \parallel \Pi_W(Y - M) \parallel_F^2 + \frac{\rho}{2} \parallel M - XB^t + \mu^t \parallel_F^2 \right) \tag{5.7}$$

$$B^{t+1} := \arg\min_{B \in \mathbb{R}^{p \times k}} \left(\frac{\lambda_1}{2} \parallel B \parallel_F^2 + \lambda_2 \parallel B \parallel_{2,1} + \frac{\rho}{2} \parallel M^{t+1} - XB + \mu^t \parallel_F^2 \right) \tag{5.8}$$

$$\mu^{t+1} := \mu^t + M^{t+1} - XB^{t+1} \tag{5.9}$$

Now the main task of our model is to solve the optimization problems proposed in Eq.(5.7) and Eq.(5.8). Next we will present the algorithm for solving Eq.(5.7) and Eq.(5.8) in detail.

Step 1: Update M^{t+1} given B^t and μ^t (solve Eq.(5.7))

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

$$\min_{M \in P} g(M) \tag{5.10}$$

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.(5.1). Let $S = \mu^t - XB^t$.

The objective function can be reformulated as:

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

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}. Therefore, the optimization problem in Eq.(5.10) equals to an Euclidean projection

$$M^{t+1} = \min_{M \in P} \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{k} \left(M_{ij} - \tilde{M}_{ij} \right)^2 \tag{5.11}$$

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.(5.11) is a special case of the non-negative max-heap projection which can be efficiently solved [51], and some details of the non-negative max-heap projection can be found in section 5.2.4.

Step 2: Update B^{t+1} given M^{t+1} and μ^t (solve Eq.(5.8))

The updating of B^{t+1} in Eq.(5.8) can be considered as a standard $l_{2,1}$-norm regularization problem:

$$\arg\min_{B \in \mathbb{R}^{p \times k}} \left(\frac{1}{2} \parallel \mathscr{L} - XB \parallel_F^2 + \rho_{L2} \parallel B \parallel_F^2 + \rho_{L1} \parallel B \parallel_{2,1} \right) \tag{5.12}$$

where $\mathscr{L}$ is the corresponding label matrix. To solve Eq.(5.8), 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.(5.8) can be solved via alternating minimization algorithm proposed in [2] or Nesterov's method with efficient Euclidean projection [50]. In this work, we choose the method proposed in [50] 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 4. We initialize 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 4: Proposed MTLSA Algorithm

Input: Feature matrix X, Target matrix Y, Weight matrix W, ρ, λ_1, λ_2
Output: $\hat{B}$

Initialize: $t = 0, M^t = Y, \mu^t = \mathbf{0}, B^t = \mathbf{0}$;
repeat
  Compute M^{t+1} by solving Eq.(5.11);
  Let $\mathscr{L} = M^{t+1} + \mu^t$, $\rho_{L1} = \frac{\lambda_2}{\rho}$, and $\rho_{L2} = \frac{\lambda_1}{2\rho}$;
  Compute B^{t+1} via solving Eq.(5.12) by standard $l_{2,1}$ solvers;
  Compute $\mu^{t+1} = \mu^t + M^{t+1} - XB^{t+1}$;
  $t = t + 1$;
until *Convergence*;
$\hat{B} = B^t$;

5.2.4 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:

$$P = \{X \geq 0, x_i \geq x_j | \forall (x_i, x_j) \in E^t\} \tag{5.13}$$

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.(5.1) is a special case of non-negative max-heap where the T^t is a sequential list.

In [51], a maximal root-tree based algorithm (Atda) was proposed to solve the non-negative max-heap projection

$$\pi_P(V) = \min_{X \in P} \frac{1}{2} \parallel X - V \parallel^2 \tag{5.14}$$

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:*

$$M(T) = \arg \max_{\tilde{T}=(\tilde{V},\tilde{E}), \tilde{T} \in R(T), m(\tilde{T}) = m_{max}(T)} |\tilde{V}| \tag{5.15}$$

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.(5.14) is given in [51], and for the non-negative non-increasing list structure Atda has a (worst-case) linear time complexity.

5.2.5 Algorithm analysis

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

Convergence analysis

The problem proposed in Eq.(5.5) follows the standard ADMM form:

$$\begin{aligned} \underset{M \in P,\, B \in \mathbb{R}^{p \times k}}{\text{minimize}} \quad & f_1(M) + f_2(B) \\ \text{subject to} \quad & M = XB \end{aligned} \tag{5.16}$$

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 $\mathbb{R}^{p \times k}$ are closed convex sets. Due to space constraints, we do not provide the proof of convergence and the readers are referred to [8,30] which provide the details of the convergence of the standard ADMM form (5.16). Based on the analysis in [30], the Algorithm 4 requires $O(\frac{1}{\epsilon})$ iterations to achieve an accuracy of ϵ.

Complexity analysis

We first analyze the time complexity of Step 1. In Eq.(5.11) $\tilde{M}_{ij}$ can be calculated with a time complexity of $O(p)$ 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 p elements. For one instance (row), the Euclidean projection in Eq.(5.11) can be efficiently calculated with a worst-case complexity of $O(k)$ [51], so for all n instances the Euclidean projection can be calculated

in $O(nk)$. Therefore, the updating of M^{t+1} needs in total $O(npk)$ calculations, where n, p, and k denote the training sample size, feature dimensionality, and the number of tasks, respectively.

From [50], we know that in Step 2 standard $l_{2,1}$-norm regularized multi-task least squares problem in Eq.(5.12) with a time complexity of $O(\frac{1}{\sqrt{\epsilon}}(npk+pk)) = O(\frac{1}{\sqrt{\epsilon}}npk)$ to achieve an accuracy of ϵ. Moreover, based on the convergence rate of Algorithm 4, we can conclude that the total time complexity of the proposed method is $O(\frac{1}{\epsilon\sqrt{\epsilon}}npk)$ for achieving an ϵ-level optimal solution.

5.2.6 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. The new variant, **MTLSA.V2** will just employ the indicator matrix W proposed in Section 5.2.1 to handle the censored instances and the $l_{2,1}$-norm for sparse learning, in the training phase. Mathematically, **MTLSA.V2** is defined as:

$$\underset{B\in\mathbb{R}^{p\times k}}{\arg\min}\frac{1}{2}\parallel \Pi_W(Y-XB)\parallel_F^2+\frac{\lambda_1}{2}\parallel B\parallel_F^2+\lambda_2\parallel B\parallel_{2,1} \tag{5.17}$$

This problem can be efficiently solved using the fast iterative shrinkage thresholding algorithm (FISTA) algorithm with $l_{2,1}$ projection [50] by incorporating the $\Pi_W(\cdot)$ when calculating the objective function and its gradient. However, in the testing phase of **MTLSA.V2**, the *non-negative non-increasing list* structure regularization will be used to enforce the estimated output to follow the property of nonrecurring event survival analysis.

5.3 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.

5.3.1 Dataset description

For our evaluation, we used several publicly available high-dimensional gene expression cancer survival benchmark datasets [1]. The datasets used in our experiments are as follows:

- The Norway/Stanford breast cancer data (NSBCD) [66] 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) [73] 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) [6] is a dataset containing observations of 86 early-stage lung adenocarcinoma patients for a period of 110 months.
- Mantle Cell Lymphoma (MCL) [2] [63] 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. [35] con-

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

Table 5.1: Details of the datasets used in this work.

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

tains 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 5.1 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 5.2.1. 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.

5.3.2 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 provide the implementation details.

- **Cox based models**: The Cox proportional hazards model [18] 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 [67]. 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 [82].
- **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 by the maximum-likelihood estimation (MLE) procedure [43]. 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 [71] is an extension of the linear regression $y_j = X_j\beta + \varepsilon_j, \varepsilon_j \sim N(0, \sigma^2)$, but the parameter is 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 [80]. 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) [53] 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 [53] [3].
- **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 [85]. Note that, these two methods cannot deal with censored instances, so the model is learned using only the uncensored instances rather than the entire set of training instances, and the labeling matrix is generated based on the scheme presented in Section 5.2.1.

5.3.3 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 [31]. Instead, the concordance index (C-index), or the *concordance probability*, is used to measure the performance of prediction models

[3] `files.figshare.com/1339232/Text_S1.pdf`

in survival analysis [28]. 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:

$$c = Pr(\hat{y}_1 > \hat{y}_2 | y_1 \geq y_2) \tag{5.18}$$

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:

$$c = \frac{1}{num} \sum_{i \in \{1 \cdots N\} \delta_i = 1} \sum_{y_j > y_i} I(X_i \hat{\beta} > X_j \hat{\beta}) \tag{5.19}$$

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:

$$c = \frac{1}{num} \sum_{i \in \{1 \cdots N\} \delta_i = 1} \sum_{y_j > y_i} I[S(\hat{y}_j | X_j) > S(\hat{y}_i | X_i)] \tag{5.20}$$

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 5.2, 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[4].

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, (as shown in Figure 5.1, all 6 instances are labeled in Day1 and Day2, and only 5 instances are labeled in Day3). Thus, in Table 5.3, we present the comparison of weighted average AUC values of different tasks (time intervals). The weighted average AUC is defined as

$$\mathrm{AUC}_{\mathrm{avg}} = \frac{\sum_{i=1}^{k} \mathrm{AUC}^{(i)} n_{\bar{c}}^{(i)}}{\sum_{i=1}^{k} n_{\bar{c}}^{(i)}} \tag{5.21}$$

where $\mathrm{AUC}^{(i)}$ is the AUC value of the i^{th} task, and $n_{\bar{c}}^{(i)}$ is the number of instances who have a known survival status label in the i^{th} time interval. The results in Table 5.3 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.

5.3.4 Scalability experiments

We empirically evaluate the scalability of the proposed **MTLSA** method with respect to the sample size (n), the number of features (p) and the number of tasks (k). The synthetic datasets are generated using the the function "simple.surv.sim" in *survsim* package [55] with different sample sizes, feature dimensionality, and maximum follow-up times (which corresponds to 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 $[-1, 1]$.

[4]The method described in [49] was not able run on our high-dimensional datasets and hence we were not able to obtain any results for that method.

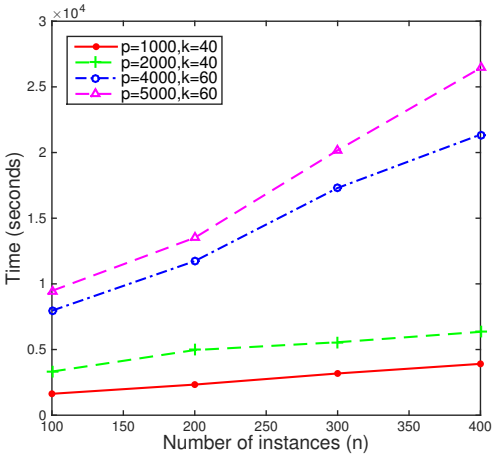
The observed time is assumed to follow a Log-logistic distribution and the time to censorship follows a Weibull distribution (which is a standarded practice in survival analysis). Figure 5.2(a) shows the runtimes for fixed p-k combination and varying n, Figure 5.2(b) shows the runtimes for fixed n-k combination and varying p, and Figure 5.2(c) shows the runtimes for fixed n-p combination and varying k. These three plots clearly demonstrate that the runtime of **MTLSA** is close to being *linear with respect to n, p and k.*

Table 5.2: Performance comparison of the proposed methods and other existing related methods using C-index values (along with their standard deviations).

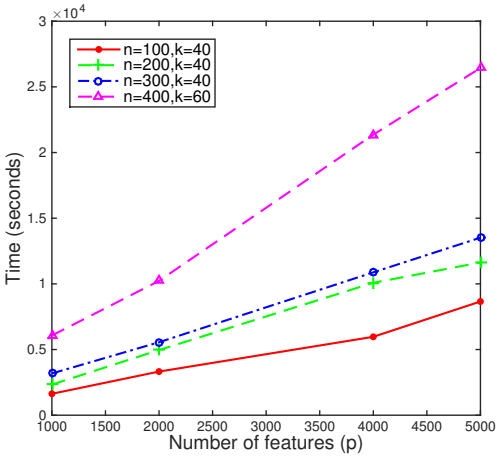
		NSBCD	**VDV**	**AML**	**Lung**	**MCL**	**DBCD**	**DLBCL**
COX based	COX	0.4411 (0.0589)	0.5973 (0.1097)	0.5515 (0.0683)	0.5158 (0.1333)	0.5773 (0.0591)	0.5539 (0.1233)	0.4553 (0.0718)
	LASSO-COX	0.5910 (0.1086)	0.6484 (0.0276)	0.5995 (0.0307)	0.6698 (0.0910)	0.6824 (0.0701)	0.6880 (0.0429)	0.6344 (0.0421)
	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 based	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)

Table 5.3: Performance comparison of the proposed methods and other existing related methods using weighted average of AUC (along with their standard deviations).

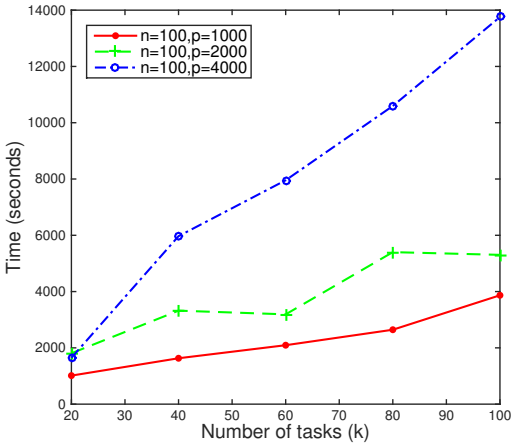
		NSBCD	**VDV**	**AML**	**Lung**	**MCL**	**DBCD**	**DLBCL**
COX based	COX	0.4611 (0.1893)	0.6352 (0.1666)	0.5351 (0.0814)	0.5464 (0.1632)	0.4695 (0.1701)	0.5334 (0.1620)	0.4480 (0.1079)
	LASSO-COX	0.5986 (0.1589)	0.6857 (0.0456)	0.7277 (0.0346)	0.7499 (0.1780)	0.7401 (0.0166)	0.7068 (0.0292)	0.7104 (0.0533)
	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. p



(c) Scalability w.r.t. k

Figure 5.2: Scalability results of the MTLSA model. The times denote total runtime for 100 λ values averaged over three trials.

CHAPTER 6 TRANSFER LEARNING FOR SURVIVAL ANALYSIS

6.1 Motivation and Overview

Collecting labeling information of survival analysis is very time consuming, i.e., one has to wait for the occurrence of the event of interest from sufficient number of training instances to build robust models. Moreover, in many practical applications, appropriate feature collection can also be extremely expensive and tedious. A naive solution for this insufficient data problem is to merely integrate the data from related tasks into a consolidated form and build prediction models on such integrated data. However, such approaches often show poor performance since the target task (where the prediction needs to be done) will be overwhelmed by auxiliary data with different distributions. In such scenarios, knowledge transfer between related tasks will usually produce much better results compared to a mere integration scheme. Transfer learning methods have been extensively studied to solve classification and standard regression problems. However, transfer learning for survival analysis has not been studied in the literature so far in spite of the clear practical need for this problem. In this thesis, we employ the Cox proportional hazards model, one of the most popular survival analysis methods, for modeling time-to-event data.

The main objective of this work is to improve the prediction performance of the Cox model in the target domain through knowledge transfer from the source domain in the context of survival models built on multiple high-dimensional datasets. The key component of our transfer learning method called **Cox-$l_{2,1}$** is to identify the "useful" knowledge that can potentially improve the performance on the target data and transfer knowledge into the model to be learned on the target domain. Specifically, we propose to employ the $l_{2,1}$-norm to penalize the sum of the loss functions (Cox proportional hazards model) for both source and target domains [2,50]. The $l_{2,1}$-norm

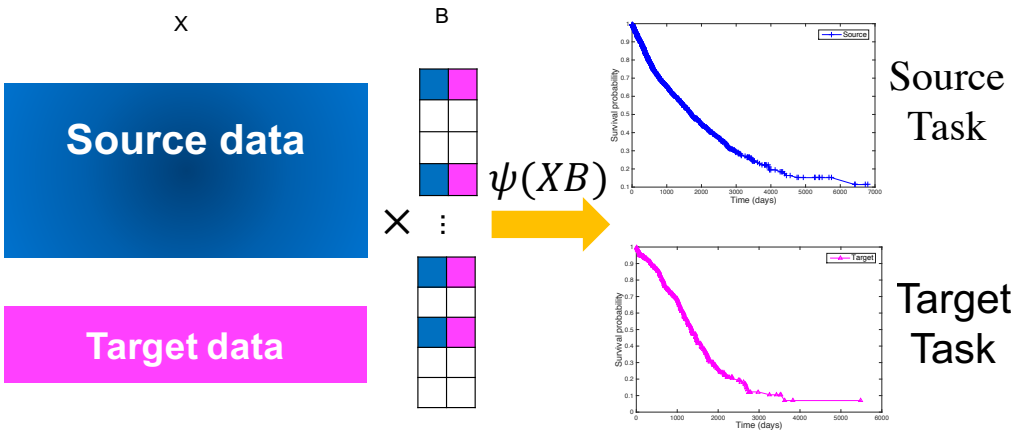


Figure 6.1: The concept map of the proposed transfer learning method for survival analysis

penalty encourages multiple predictors to share similar sparsity patterns; thus, it cannot only select important features but also learn a shared representation across source and target domains to improve the model performance on the target task. Figure 6.1 presents the concept map of the proposed model. The proposed transfer learning formulation is solved via the fast iterative shrinkage thresholding algorithm (FISTA) [5], where $\psi(\cdot)$ denotes a survival prediction model. In addition, with the help of a risk set updating method [65], the proposed **Cox-$l_{2,1}$** algorithm achieves a linear time complexity with respect to both training sample size and feature dimensionality.

We demonstrate the prediction performance of our **Cox-$l_{2,1}$** model using real-world high-dimensional microarray gene expression datasets which include patients from various cancer types. Our results demonstrate the power of the proposed **Cox-$l_{2,1}$** model in transferring knowledge from the related cancer types to improve the survival prediction for one particular cancer type. Although the proposed algorithm is efficient, it is still time consuming due to the high dimensionality of the dataset ($19,171$ features). To this end, we adapt the idea of *screening* so that it is applicable to censored data by utilizing the strong rules [69]. Screening is a state-of-the-art technology which is able to efficiently identify the number of features whose corresponding

coefficients are guaranteed to be zero. Removal of these features will dramatically reduce the dimensionality of the feature space. In this thesis, we extend the strong rule to sparse survival analysis models with multiple datasets and significantly increase the efficiency of the algorithm without compromising the prediction performance.

The main contributions of this work are summarized as follows:

- Propose a novel transfer learning method **Cox-$l_{2,1}$** for survival analysis which can select a subset of joint features to transfer the knowledge from the source domain to the target domain in the presence of censored data.
- Develop screening mechanism for censored data by extending the strong rule to sparse survival models with multiple datasets and use it to improve the efficiency of the algorithm without compromising the prediction performance.
- Demonstrate the performance of the proposed transfer learning method using several synthetic and high-dimensional microarray gene expression benchmark datasets, and compare it with state-of-the-art survival analysis methods.

6.2 Related Work on Transfer learning

In this section, we present the related works in the area of transfer learning and highlight the primary distinctions of the proposed work compared to the existing methods that are available in the literature.

Transfer learning methods have been successfully applied in many real-world applications such as text mining [57], collaborative filtering [59] and biomedical data analysis [45]. In transfer learning, the primary goal is to adapt a model built on source domain D_S (or distribution) for performing prediction on the target domain D_T. Pan *et al.* [58] categorized transfer learning methods into three different types, namely, *inductive*, *transductive* and *unsupervised* transfer learning, based on different settings for transfer. The model we propose in this thesis belongs to the inductive transfer learning approach, more specifically, similar to *multi-task learning* [2]. In

Table 6.1: Relationship between the proposed model and traditional multi-task learning related inductive transfer learning methods.

Tasks	Source & Target Labels	Related Literatures
Classification	Categorical / Fully informative	[21], [37], [57], $\cdots$
Regression	Numeric / Fully informative	[2], [60], [26], $\cdots$
Survival analysis	Numeric / Partially informative	Our work

multi-task learning different tasks are learned simultaneously and equally weighted, while in the case of transfer learning, one set of data is selected as the target domain and the remaining is used as the source domains. Furthermore, it is very convenient to adapt a multi-task learning algorithm to transfer learning algorithm by merely enhancing the importance (weight) of the target task [58].

In all the methods described above and other related works (refer to [58]), the source and target tasks are either classification or regression problems. However, in this thesis, the source and target tasks are the corresponding regression-based loss functions which include censored information. We propose a proportional hazards [18] based transfer learning model to transfer the useful and relevant knowledge from the source to the target domain. Our approach can effectively handle censored information based on the partial likelihood function which makes it unique. Table 6.1 summarizes the relationship between traditional multi-task learning related inductive transfer learning methods and the proposed model. It should be noted that in survival analysis, the label information is available but is partially informative for censored instances. Hence, techniques like self-taught learning [61] and transductive learning [3] which handle scenarios with missing label information are not suitable for handling such partially imformative label information.

Based on the underlying learning mechanism, transfer learning methods can be grouped into four categories [58]: *instance-based*, *feature-based*, *parameter-based*, and *relational knowledge-based*. Our proposed model is a feature-based transfer learning

paradigm, which employs the $l_{2,1}$-norm [2,50] to penalize the sum of the loss functions (Cox proportional hazards model) of both source and target tasks. The $l_{2,1}$-norm encourages multiple predictors to share similar sparsity patterns; thus, it cannot only select important features and alleviate over-fitting in high-dimensional datasets but also learn a shared representation across source domain and target domain to improve the model performance on the target task.

6.3 Proposed Model

In this section, we will first introduce some basic concepts of survival analysis and Cox proportional hazard regression. Then we will propose the transfer learning methods based on $l_{2,1}$-norm regularized Cox model and the optimization approach.

6.3.1 Preliminaries

In survival analysis, for each data instance, we observe either a failure time (O_i) or a censored time (C_i), but not both. The dataset is said to be right-censored if and only if $y_i = \min(O_i, C_i)$ can be observed during the study. An instance in the survival data is usually represented by a triplet (X_i, T_i, δ_i), where X_i is a $1 \times p$ 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 failure time O_i for uncensored instances and C_i otherwise, i.e.

$$T_i = \begin{cases} O_i & \text{if} \quad \delta_i = 1 \\ C_i & \text{if} \quad \delta_i = 0 \end{cases} \tag{6.1}$$

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 survival analysis, one of the most important concepts in modeling such censored

data is the *hazards function* $h_i(t) = \lim_{\Delta t \to 0} \frac{Pr(t \leq O_i < t+\Delta t | O_i \geq t)}{\Delta t}$, which is the event rate at time t conditional on survival until time t or later. In the Cox model, the proportional hazards assumption is

$$h(t, X_i) = h_0(t) \exp(X_i \beta) \tag{6.2}$$

for $i = 1, 2, ..., N$, where the $h_0(t)$ is the *baseline hazard function*, which can be an arbitrary non-negative function of time, and β is a $p \times 1$ regression coefficient vector of the Cox proportional hazards model. The Cox model is a semi-parametric model since all the instances share a same baseline hazard function and the coefficient estimation is independent of the form of $h_0(t)$. Let $O_1 < O_2 < \cdots < O_K$ be the increasing list of unique failure times of all N instances; given the fact that an event occurs at O_i, the conditional probability of the individuals corresponding covariate is X_i can be formulated as

$$Pr(X_i | O_i) = \frac{h(O_i, X_i)\Delta t}{\sum_{j \in R_i} h(O_i, X_j)\Delta t} = \frac{\exp(X_i \beta)}{\sum_{j \in R_i} \exp(X_j \beta)} \tag{6.3}$$

where R_i is the risk set at O_i which consists of all instances whose failure times are equal to or greater than O_i. Thus, the β can be learned via maximizing the partial likelihood:

$$L(\beta) = \prod_{i=1}^{K} \frac{exp(X_i \beta)}{\sum_{j \in R_i} \exp(X_j \beta)} \tag{6.4}$$

6.3.2 L2,1-norm regularized Cox model

In this thesis, we propose a feature-based transfer learning method which aims at finding "good" features to transfer knowledge from source domain to target domain and miminize the prediction error of target task. In standard transfer learning the source and target tasks are either classification or standard regression, but in survival

analysis the source and target tasks are censored regression. Cox model is the most widely used survival analysis method, and we employ it as the loss function for both source and target tasks. However, Eq.(6.4) fails to handle the tied failures, i.e., two or more failure events that occur at same time. In this work, the Breslow approximation [11] is used to deal with the tied failures. The partial likelihood is reformulated as follows:

$$L(\beta) = \prod_{i=1}^{K} \frac{\exp(\sum_{j \in D_i} X_j \beta)}{[\sum_{j \in R_i} \exp(X_j \beta)]^{d_i}} \tag{6.5}$$

where D_i contains all instances whose failure time is O_i and $d_i = |D_i|$ is the size of D_i. Therefore, the coefficient vector can be learned via minimizing the negative log-partial likelihood.

$$l(\beta) = -\sum_{i=1}^{K} \left\{ \sum_{j \in D_i} X_j \beta - d_i \log \left[\sum_{j \in R_i} \exp(X_j \beta) \right] \right\} \tag{6.6}$$

To find "good" features for knowledge transfer, we propose a model which is able to learn a shared representation across source and target tasks. The $l_{2,1}$-norm is chosen to be one penalty term for our model because it encourages multiple coefficient vectors to share similar sparsity patterns. Therefore, the regularized model learns a shared representation across source and target tasks. In addition, a sparsity inducing penalty also helps the model deal with high-dimensional datasets and alleviate model over-fitting. The proposed transfer learning model "**Cox-$\boldsymbol{l_{2,1}}$**" can be learned via solving the following minimization problem.

$$\min_{B} \sum_{t \in \{S,T\}} -\frac{w_t}{N_t} l(\beta_t) + \frac{\mu}{2} \parallel B \parallel_F^2 + \lambda \parallel B \parallel_{2,1} \tag{6.7}$$

where S and T denote the tasks in the source domain and target domain, respectively.

$B = (\beta_S, \beta_T), B \in \mathbb{R}^{p\times 2}$, N_S and N_T are the number of training instances in the source domain and target domain, respectively. w_S and w_T are two empirically determined weight parameters, and usually $w_S < w_T$ which induces the model focusing more on the target task. The l_2 regularization on the coefficient matrix B is introduced to further reduce the variance of B and alleviate model over-fitting.

6.3.3 Optimization

The optimization problem proposed in Eq.(6.7) follows the standard $l_{1,2}$-norm regularization problem:

$$\min_{B\in\mathbb{R}^{p\times 2}} g(B) + \lambda \parallel B \parallel_{2,1} \tag{6.8}$$

where $\lambda > 0$ is the regularization parameter, and

$$g(B) = \sum_{t\in\{S,T\}} -\frac{w_t}{N_t} l(\beta_t) + \frac{\mu}{2} \parallel B \parallel_F^2$$

is a smooth convex loss function, and its first order derivative can be calculated as:

$$g^{'}(B) = \left[\frac{w_S}{N_S} l^{'}(\beta_S) + \mu\beta_S,\ \frac{w_T}{N_T} l^{'}(\beta_T) + \mu\beta_T\right] \tag{6.9}$$

where $l^{'}(\beta_S)$ and $l^{'}(\beta_T)$ are the gradient of the negative log-partial likelihood as shown in Eq.(6.6), and these two terms share the same formulation

$$l^{'}(\beta) = -\sum_{i=1}^{K}\left\{\sum_{j\in D_i} X_j - d_i \frac{\sum_{j\in R_i} X_j \exp(X_j\beta)}{\sum_{j\in R_i}\exp(X_j\beta)}\right\} \tag{6.10}$$

coresponding to the source and target datasets, respectively.

The optimization problem in Eq.(6.8) can be solved efficiently via the FISTA

based algorithm with the general updating step,

$$B^{(i+1)} = \pi_P(S^{(i)} - \frac{1}{\gamma_i} g^{'}(S^{(i)})) \tag{6.11}$$

where $S^{(i)} = B^{(i)} + \alpha_i(B^{(i)} - B^{(i-1)}) = [S_S^{(i)},\ S_T^{(i)}]$ are two search points of the source task and target task, respectively. α_i is the combination scaler, $g^{'}(S^{(i)})$ is the gradient of $g(\cdot)$ at point $S^{(i)}$, $\frac{1}{\gamma_i}$ is the possible biggest stepsize which is chosen by line search, and $\pi_P(\cdot)$ is the $l_{1,2}$-regularized Euclidean projection:

$$\pi_P(G(S^{(i)})) = \min \frac{1}{2} \parallel B - G(S^{(i)})) \parallel_F^2 + \lambda \parallel B \parallel_{2,1} \tag{6.12}$$

where $G(S^{(i)}) = S^{(i)} - \frac{1}{\gamma_i} g^{'}(S^{(i)})$. An efficient solution (Theorem 2) of Eq.(6.12) has been proposed in [50].

Theorem 2. *Given λ, the primal optimal point $\hat{B}$ of Eq.(6.12) can be calculated as:*

$$\hat{B}_j = \begin{cases} \left(1 - \frac{\lambda}{\|G(S^{(i)})_j\|_2}\right) G(S^{(i)})_j & \text{if} \quad \lambda > 0, \parallel G(S^{(i)})_j \parallel_2 > \lambda \\ 0 & \text{if} \quad \lambda > 0, \parallel G(S^{(i)})_j \parallel_2 \leq \lambda \\ G(S^{(i)})_j & \text{if} \quad \lambda = 0 \end{cases} \tag{6.13}$$

where $G(S^{(i)})_j$ is the j^{th} row of $G(S^{(i)})$, and $\hat{B}_j$ is the j^{th} row of $\hat{B}$.

Algorithm 5 outlines the learning procedure of FISTA algorithm to solve optimization problem in Eq.(6.7). In lines 4-11, the optimal γ_i is chosen by the backtracking rule; thus, based on [5, Lemma 2.1, page 189], γ_i is greater than or equal to the Lipschitz constant of $g(\cdot)$ at $S^{(i)}$, which means γ_i is satisfied for $S^{(i)}$ and $\frac{1}{\gamma_i}$ is the biggest possible stepsize. In line 7, $Q_\gamma(S^{(i)}, B^{(i+1)})$ is the tangent line of $g(\cdot)$ at $S^{(i)}$,

Algorithm 5: FISTA algorithm for **Cox-$l_{2,1}$**

Input: Source dataset D_S, Target dataset D_T, Initial coefficient matrix $B^{(0)}$, w, μ, λ

Output: $\bar{B}$

Initialize: $B^{(1)} = B^{(0)}$, $d_{-1} = 0$, $d_0 = 1$,$\gamma_0 = 1$,$i = 1$;
repeat
  Set $\alpha_i = \frac{d_{i-2}-1}{d_{i-1}}$, $S^{(i)} = B^{(i)} + \alpha_i(B^{(i)} - B^{(i-1)})$;
  for $j = 1, 2, \cdots$ **do**
    Set $\gamma = 2^j\gamma_{i-1}$;
    Calculate $B^{(i+1)} = \pi_P(S^{(i)} - \frac{1}{\gamma}g^{'}(S^{(i)}))$;
    Calculate $Q_\gamma(S^{(i)}, B^{(i+1)})$;
    if $g(B^{(i+1)}) \leq Q_\gamma(S^{(i)}, B^{(i+1)})$ **then**
      $\gamma_i = \gamma$, **break** ;
    end
  end
  $d_i = \frac{1+\sqrt{1+4d_{i-1}^2}}{2}$;
  $i = i + 1$;
until *Convergence of* $B^{(i)}$;
$\bar{B} = B^{(i)}$;

which can be calculated as

$$
\begin{aligned}
&Q_\gamma(S^{(i)}, B^{(i+1)}) \\
&= g(S^{(i)}) + \frac{\gamma}{2} \parallel B^{(i+1)} - S^{(i)} \parallel^2 + \langle B^{(i+1)} - S^{(i)}, g^{'}(S^{(i)}) \rangle
\end{aligned}
\tag{6.14}
$$

6.3.4 Complexity Analysis

The main cost per iteration of our optimization scheme is the computation of $g(\cdot)$ and $g^{'}(\cdot)$, more specifically, the computation of the negative log-partial likelihood and its gradient. From Eq.(6.6) and Eq.(6.10), we can see that, at each failure time point O_i, one needs to calculate $\sum_{j \in R_i} e^{X_j\beta}$ and $\sum_{j \in R_i} X_j e^{X_j\beta}$; thus, for all failure times, it needs $O(N^2p)$ calculations, because R_i has $O(N)$ elements. To speedup the training process, we employ the risk set updating method proposed in [65] which is given as

follows.

$$\sum_{j \in R_{i+1}} e^{X_j \beta} = \sum_{j \in R_i} e^{X_j \beta} - \sum_{j \in (R_i - R_{i+1})} e^{X_j \beta} \tag{6.15}$$
$$\sum_{j \in R_{i+1}} X_j e^{X_j \beta} = \sum_{j \in R_i} X_j e^{X_j \beta} - \sum_{j \in (R_i - R_{i+1})} X_j e^{X_j \beta}$$

Here, we only need to calculate $\sum_{j \in R_1} e^{X_j \beta}$ and $\sum_{j \in R_1} X_j e^{X_j \beta}$. Then for the subsequent failure time point O_i, we subtract the contribution from instances which are failed or censored between O_{i-1} and O_i. Therefore, the calculations of $l(\beta)$ and $l^{'}(\beta)$ are both reduced to $O(Np)$, and the computation cost of $g(\cdot)$ and $g^{'}(\cdot)$ are both $O((N_S + N_T)p)$.

In our transfer learning problem, there are only two tasks (source and target tasks), so the Euclidean projection in Eq.(6.12) can be efficiently calculated in $O(2p) = O(p)$. Therefore, the optimization procedure solves the optimization problem in Eq.(6.7) with a time complexity of $O(\frac{1}{\sqrt{\varepsilon}}(N_S + N_T)p)$ for achieving an accuracy of ε.

6.3.5 Solution Path and Strong Rule

Usually, in the learning process, the model has to be trained based on a series of values for λ, and the best λ is selected via cross-validation. In this thesis, we employ the warm-start approach given in [25] to build the solution path; initialize λ to a sufficiently large number, which forces B to a zero matrix, and then gradually decreases λ in each learning iteration. For a new λ, the initial value of B is the estimated B learned from the previous λ, so the initial value of B is not far from the optimal value, and the algorithm will converge within a few iterations.

Firstly, λ_{max}, the smallest tuning parameter value which forces B to a zero matrix, needs to be calculated. From Eq.(6.13) we can see that if $\| G(S^{(0)})_j \|_2 < \lambda$ for all j,

then $B = \mathbf{0}$ is the optimal solution. Thus, we set

$$\begin{aligned} \lambda_{max} &= \max_j \parallel G(S^{(0)})_j \parallel_2 \\ &= \max_j \parallel S_j^{(0)} - \frac{1}{\gamma_0} g^{'}(S^{(0)})_j \parallel_2 = \max_j \parallel g^{'}(\mathbf{0})_j \parallel_2 \end{aligned} \tag{6.16}$$

to be the first λ, where $g^{'}(\cdot)_j$ is the j^{th} row of $g^{'}(\cdot)$. If $\min(N_S, N_T) \geq p$ we set $\lambda_{min} = 0.0001\lambda_{max}$, else we set $\lambda_{min} = 0.05\lambda_{max}$. In our experiments, we search m different λ values in total, and for the k^{th} step $\lambda_k = \lambda_{max}(\lambda_{min}/\lambda_{max})^{k/m}$.

The FISTA based learning scheme is an efficient method to solve the transfer learning problems proposed in Eq.(6.7). However, if the feature dimensionality (p) is extremely large, the proposed optimization approach will still take substantial amount of time. *Screening* is a state-of-the-art technology which is able to efficiently identify features whose corresponding coefficients are guaranteed to be zero. Removal of these features will dramatically reduce the feature dimension; thus, screening is able to improve the efficiency of many sparse models [77]. Our optimization problem in Eq.(6.7) can be rewritten as

$$\min_B g(B) + \lambda \sum_{j=1}^{p} \parallel B_j \parallel_2 \tag{6.17}$$

where B_j stands for the j^{th} row of B. Eq.(6.17) belongs to the general Lasso-type problems, and based on the the Karush-Kuhn-Tucker (KKT) conditions, Tibshirani et al. proposed the strong rules for this type of problems [69]. The KKT conditions for Eq.(6.17) are

$$g^{'}(\hat{B})_j = \lambda \boldsymbol{\theta}_j \text{ for } j = 1, 2, ..., p \tag{6.18}$$

where $\hat{B}$ is the optimal solution and $\boldsymbol{\theta}_j$ is a subgradient of $\parallel \hat{B}_j \parallel_2$, which satisfies

$\| \boldsymbol{\theta}_j \|_2 \leq 1$ and $\| \boldsymbol{\theta}_j \|_2 < 1$ implies $\hat{B}_j = 0$. Based on the above KKT conditions and [69, Section 6, page 17], for our problem the sequential strong rule for Eq.(6.17) to discard inactive features (corresponding coefficients are zero) is as follows.

Theorem 3. *Given a sequence of parameter values* $\lambda_{max} = \lambda_0 > \lambda_1 > \cdots > \lambda_m$, *and suppose the optimal solution* $\hat{B}(k-1)$ *at* λ_{k-1} *is known. Then for any* $k = 1, 2, \cdots, m$ *the* j^{th} *feature will be discarded if*

$$\| g^{'}(\hat{B}(k-1))_j \|_2 < 2\lambda_k - \lambda_{k-1} \tag{6.19}$$

and the corresponding coefficient $\hat{B}(k)_j$ *will be set to* 0.

However, based on the experimental analysis in [69], we know that, Theorem 3 might mistakenly discard active features (corresponding coefficients are nonzero), so we need to check KKT conditions of the discarded features. Let V^d and V^s denote the index set of discarded features and selected features, respectively. From Theorem 3, we get $\hat{B}(k)_j = 0$, $\forall j \in V^d$, and based on Eq.(6.13) we know that if

$$\| g^{'}(\hat{B}(k))_j \|_2 \leq \lambda_k \quad \forall j \in V^d$$

is true, then $\hat{B}(k)$ is the optimal solution at λ_k. Otherwise, V^s need to be updated via $V^s = V^s \cup V^v$ where

$$V^v = \left\{ j \middle| j \in V^d,\ \| g^{'}(\hat{B}(k))_j \|_2 > \lambda_k \right\} \tag{6.20}$$

is the index set of mis-discarded features.

Above all, Figure 6.2 summarizes our proposed model with solution path and strong rule. Firstly, λ_{max} will be calculated by Eq.(6.16) as the starting searching point. Next, the strong rule will be used to discard inactive features, and the model

will be trained with the selected features. To prevent mis-discarding, we then have to check the KKT conditions. If there are any mis-discarded features, we have to update the set of selected features and retrain the model; if not, we can train a model based on a new λ.

6.4 Experimental Results

In this section, we will first describe the datasets used in our evaluation and demonstrate the prediction performance of the proposed **Cox-$l_{2,1}$** model. Then we will experimentally demonstrate the efficiency of the screening methods and the scalability of the proposed algorithm. Finally, we perform a detailed study on the biomarkers selected by the proposed algorithm on different cancer types and show their biological significance as well.

6.4.1 Dataset Description

For our model evaluation, we use publicly available [1] high-dimensional gene expression cancer survival benchmark datasets from The Cancer Genome Atlas (TCGA) [38]. In this thesis, we perform knowledge transfer in survival analysis by analyzing the microarray gene expressions for different cancer types. We have labeled data in all cancer types. The dataset contains somatic mutational profiles for $3,096$ cancer patients with survival information, and for each patient the relative activity of $19,171$ genes are measured. These gene values are considered to be the features in our data. The cancer patients belong to one of the 12 major cancer types: bladder urothelial carcinoma (BLCA), breast adenocarcinoma (BRCA), colon carcinoma (COAD), rectal carcinoma (READ), glioblastoma multiforme (GBM), head and neck squamous cell carcinoma (HNSC), kidney renal clear cell carcinoma (KIRC), acute myeloid leukaemia (LAML), lung adenocarcinoma (LUAD), lung squamous cell carcinoma (LUSC), ovarian serous carcinoma (OV) and uterine corpus endometrial carcinoma

[1] Downloaded from `https://cran.r-project.org/web/packages/dnet/index.html`

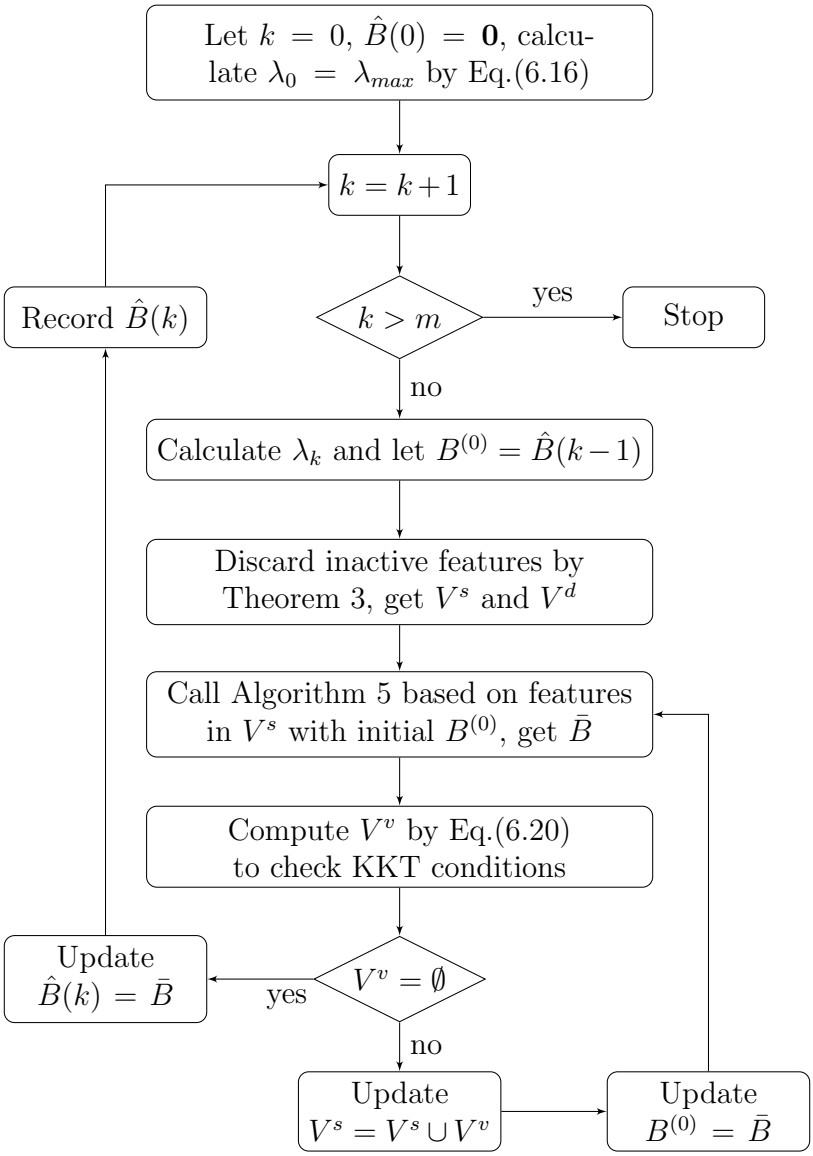


Figure 6.2: Flowchart for **Cox-$l_{2,1}$** algorithm with strong rule.

Table 6.2: Basic statistics of the selected 8 cancer types.

Data	**# Instances**	**# Uncensored**	**# Censored**
BRCA	763	90	673
GBM	275	176	99
HNSC	300	119	181
KIRC	417	136	281
LAML	185	117	68
LUAD	155	50	105
LUSC	171	68	103
OV	315	181	134

(UCEC).

In our experiments, the goal is to improve the performance of survival prediction for a particular cancer type. Hence, in our transfer learning setting, this specific cancer type would be considered to be the target domain and the data from remaining types is considered to be the source domain. Table 6.2 shows the basic statistics of the cancer types considered for our analysis. The number of uncensored instances in four cancer types are too small and hence these four cancer types were eliminated for our evaluation. In our experiments, for the remaining 8 cancer types, each of them will be considered as the target domain and the data from the remaining cancer types will be considered as the source domain. In this table, the columns titled "# Uncensored" and "# Censored" correspond to the number of uncensored and censored instances in each cancer type, respectively. For these cancer types, the event of interest is patient death; therefore, an uncensored instance refers to the corresponding patient is dead during the study, while a censored instance refers to the corresponding patient is still alive at the last observed time (censored time).

6.4.2 Performance Comparison

To the best of our knowledge, neither transfer learning nor multi-task learning for survival analysis have been studied in the literature. Hence, we can only compare our

proposed **Cox-$l_{2,1}$** with standard related survival analysis methods. As our **Cox-$l_{2,1}$** is a Cox-based model and $l_{2,1}$-norm is a Lasso-type penalty, we choose Cox model and other two popular regularized Cox models: LASSO-COX and EN-COX as comparison methods. In our experiments, these three survival analysis methods are applied both on the target dataset (the specified cancer type) and the entire dataset. For simplicity, they are referred to as "Local" models and "Global" models, respectively. It should be noted that, in "Global" models, although each model is built on the entire dataset, the performance is measured only on the target dataset. For a "Local" model, the training and testing are performed only on the target cancer type (using cross validation). For a global model, the training is done on the source+target samples and the testing is done on the target cancer samples.

The concordance index (C-index), or *concordance probability*, is used to measure the performance of prediction models in survival analysis [28]. 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

$$c = Pr(\hat{y}_1 > \hat{y}_2 | y_1 \geq y_2). \tag{6.21}$$

By definition, the C-index has the same scale as the area under the ROC curve (AUC) in binary classification, and if y_i is binary, then the C-index is same as the AUC. In the standard Cox model and regularized Cox models, the hazard ratio is modeled to describe the time-to-event data. The instances with a low hazard rate should survive longer, so the C-index is calculated as follows:

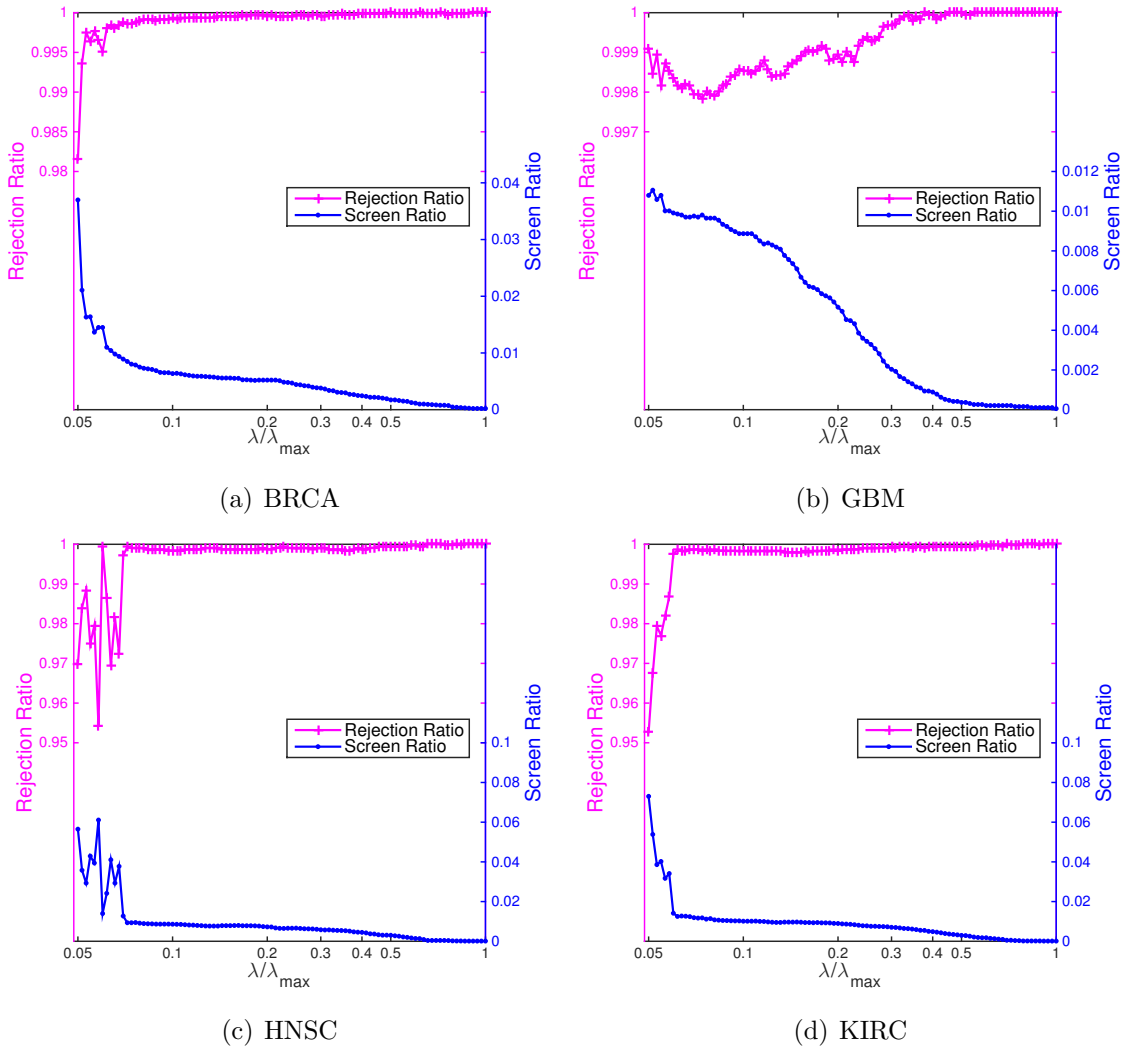
$$c = \frac{1}{num} \sum_{i \in \{1 \cdots N | \delta_i = 1\}} \sum_{y_j > y_i} I[X_i \hat{\beta} > X_j \hat{\beta}] \tag{6.22}$$

where num denotes the number of comparable pairs and $I[\cdot]$ is the indicator function.

Table 6.3: Performance comparison of the proposed **Cox-$l_{2,1}$** method and other existing related methods using C-index values (along with their standard deviations).

Dataset	**Local**			**Global**			**Cox-$l_{2,1}$**
	COX	**LASSO-COX**	**EN-COX**	**COX**	**LASSO-COX**	**EN-COX**	
BRCA	0.4348 (0.0756)	0.3868 (0.0418)	0.4055 (0.0426)	0.5547 (0.0238)	0.5822 (0.0394)	0.5811 (0.0411)	**0.5869** (**0.0456**)
GBM	0.5064 (0.0677)	0.5741 (0.0181)	0.5613 (0.0260)	0.5592 (0.0243)	0.5841 (0.0131)	0.5842 (0.0130)	**0.6136** (**0.0300**)
HNSC	0.5663 (0.0759)	0.5591 (0.0527)	0.5788 (0.0491)	0.5794 (0.0059)	0.5528 (0.0776)	0.5542 (0.0789)	**0.6157** (**0.0379**)
KIRC	0.5689 (0.0322)	0.6001 (0.0206)	0.6061 (0.0216)	0.5553 (0.0603)	0.5903 (0.0401)	0.5908 (0.0386)	**0.6255** (**0.0393**)
LAML	0.5599 (0.0887)	0.6861 (0.0189)	0.6838 (0.0227)	0.6057 (0.0397)	0.6591 (0.0103)	0.6580 (0.0141)	**0.6939** (**0.0305**)
LUAD	0.3832 (0.1371)	0.5327 (0.0840)	0.5435 (0.0337)	0.4463 (0.0443)	0.5354 (0.1026)	0.5378 (0.1040)	**0.5877** (**0.0409**)
LUSC	0.5250 (0.0719)	0.4670 (0.1009)	0.4861 (0.0598)	0.5520 (0.0426)	0.5798 (0.0465)	0.5770 (0.0584)	**0.5905** (**0.0374**)
OV	0.5132 (0.0260)	0.4991 (0.1043)	0.4971 (0.0911)	0.5438 (0.0826)	0.5708 (0.0850)	0.5697 (0.0825)	**0.6167** (**0.0342**)

In Table 6.3, we show the performance results of C-index values of different algorithms using 5-fold cross validation. The best results are highlighted in bold. The results show that our proposed **Cox-$l_{2,1}$** model outperforms the other state-of-the-art models. We also notice that for 7 out of the 8 cancer types, the "Global" Cox model performs better than the "Local" Cox model, which indicates that having more samples from other cancer types will help in generalization and alleviate over-fitting. However, for 4 cancer types in the "Local" regularized Cox models perform better than the "Global" regularized Cox models; this phenomenon reflects that, from the genetics perspective, the preventable factors and reflections of different cancer types are clearly different.



(a) BRCA

(b) GBM

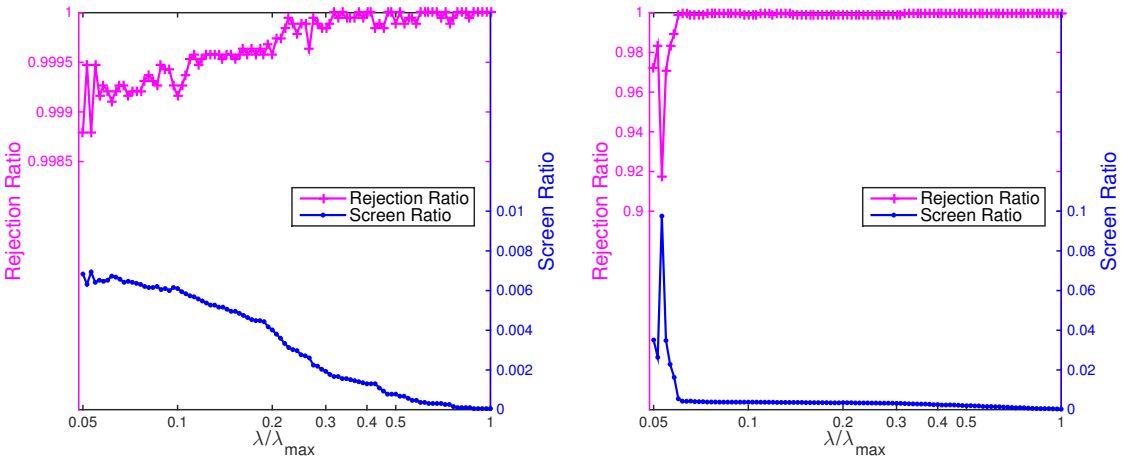
(c) HNSC

(d) KIRC

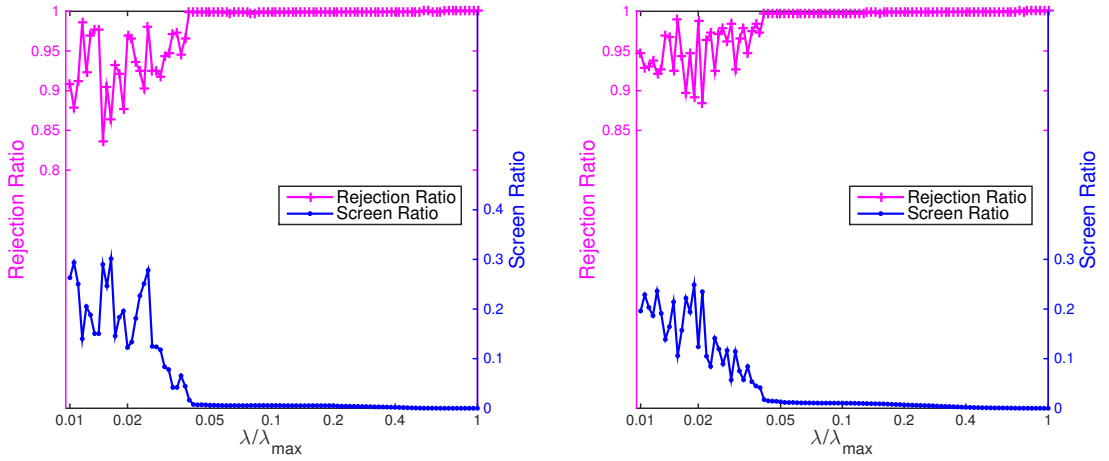
Figure 6.3: *Efficiency of strong rule*: Plots of the rejection ratio and screen ratio on gene expression data for 4 cancer types.

6.4.3 Empirical Analysis of Efficiency

In this section, we will demonstrate the efficiency of the strong rule and also show the scalability performance of the proposed **Cox-$l_{2,1}$** algorithm.



(a) LAML (b) LUAD



(c) LUSC (d) OV

Figure 6.4: *Efficiency of strong rule* (Cont.): Plots of the rejection ratio and screen ratio on gene expression data for 4 remaining cancer types.

Efficiency of strong rule

To measure the efficiency of applying strong rule, we measure the *rejection ratio* and *screen ratio* which are defined as follows:

$$\text{rejection ratio} = \frac{\text{number of identified inactive features}}{\text{number of true inactive features}}$$

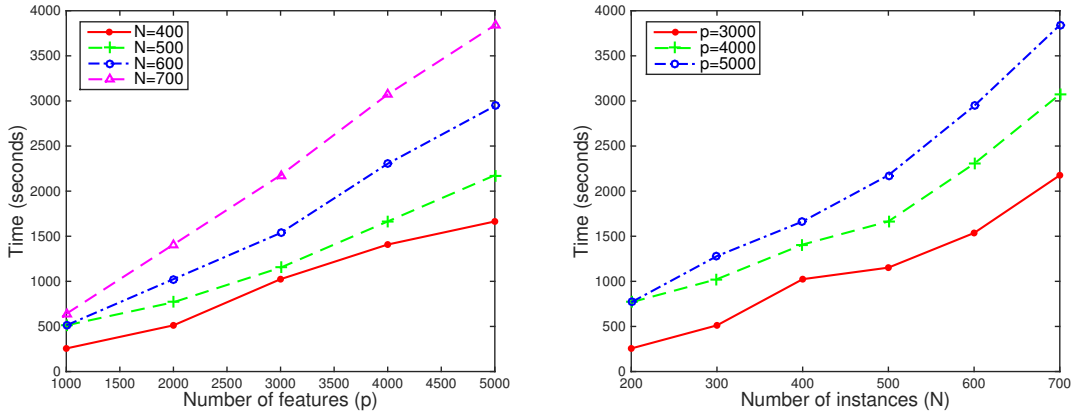
$$\text{screen ratio} = \frac{\text{number of selected features}}{\text{original feature dimension}}$$

Figure 6.4 shows the *rejection ratio* and *screen ratio* of the strong rule on gene expression data of 8 cancer types. In Figure 6.3 and Figure 6.4 (a)–(b) the λ_{min} is set equal to $0.05\lambda_{max}$, as mentioned in Section 6.3.5. However, under this setting, less than one hundred features will be selected as active features in "LUSC" and "OV", so we set $\lambda_{min} = 0.01\lambda_{max}$ and draw the screening ratio in Figure 6.4(c) and Figure 6.4(d), for these two cancer types. All plots in Figure 6.3 and Figure 6.4 reflect that the strong rule in the proposed **Cox-$l_{2,1}$** model can successfully identify a majority of the inactive features (high rejection ratio) and dramatically decrease the feature dimensionality (low screen ratio) in the learning phase.

Table 6.4 presents the running time of the **Cox-$l_{2,1}$** with and without strong rule and the *speedup* achieved. All the timing calculations are based on running the experiments on an Intel Xeon 3 GHz processor with 12 cores (24 threads). *Speedup* is the ratio of the running time of **Cox-$l_{2,1}$** without screening to it's running time with screening. We only show the result on one cancer type "LUAD" because without screening the computation of **Cox-$l_{2,1}$** takes very long (more than one day). In addition to this cancer data, we also generated two synthetic datasets "Syn1" and "Syn2" using the the function "simple.surv.sim" in *survsim* package [55]. These two datasets have 500 instances in the source domian, 100 instances in the target domian, and a maximum follow-up time of $1,000$ days. All the features are generated based on the uniform distribution, and each of them have a different random setted value range. The coefficient vector is also randomly generated and remain in $[-1, 1]$. The observed time is assumed to follow a Log-logistic distribution and time to censorship follows a Weibull distribution. The datasets in scalability analysis are also generated in the same manner with different sample size and feature dimension. The results show that the screening method can dramatically speed up the algorithm and become more effective as the feature dimension increases (see Table 6.4).

Table 6.4: Running time comparison for the **Cox-$l_{2,1}$** model with and without screening rule for 100 λ values with default setting ($\lambda_{min} = 0.05\lambda_{max}$).

Data	p	**With screening**	**Without screening**	**speedup**
Syn 1	5,000	768 (s)	2944 (s)	3.83
Syn 2	10,000	1286 (s)	6084 (s)	4.73
LUAD	19,171	3.59 (hrs)	35.13 (hrs)	9.78



(a) Scalability w.r.t. p

(b) Scalability w.r.t. N

Figure 6.5: *Scalability resuts*: Plots of the runtimes for the **Cox-$l_{2,1}$** model. The times correspond to the total runtimes for 100 λ values averaged over five trials.

Scalability of Cox-$l_{2,1}$

We empirically evaluate the scalability of the proposed **Cox-$l_{2,1}$** model with respect to the sample size ($N = N_S + N_T$) and the number of features (p). In this experiment, we did not use strong rule as it will influence the the scalability analysis of **Cox-$l_{2,1}$** with respect to p. This is because the strong rule will discard features and make p unstable with different λ values, which is clearly shown in Figure 6.4. Figure 6.5(a) shows runtimes for fixed N and varying p, and Figure 6.5(b) shows runtimes for fixed p and varying N. These two plots suggest that the runtime of **Cox-$l_{2,1}$** is close to being *linear with respect to both N and p.*

Table 6.5: Top 10 gene expression features obtained for each cancer type using **Cox-$l_{2,1}$** model.

Cancer Type: BRCA		**Cancer Type**: GBM	
Symbol	Description	Symbol	Description
SSBP4	single stranded DNA binding protein 4	CALR3	calreticulin 3
METTL22	methyltransferase like 22	AK5	adenylate kinase 5
SGCD	sarcoglycan (35kDa dystrophin-associated glycoprotein)	PSMF1	proteasome inhibitor subunit 1 (PI31)
LYPD6B	LY6/PLAUR domain containing 6B	SLC31A2	solute carrier family 31 (copper transporter), #2
FCGR1B	Fc fragment of IgG, high affinity Ib, receptor (CD64)	PPP1R12C	protein phosphatase 1, regulatory subunit 12C
CXCL14	chemokine (C-X-C motif) ligand 14	CHORDC1	cysteine and histidine-rich domain containing 1
RAB27A	RAB27A, member RAS oncogene family	PROX2	prospero homeobox 2
CD6	CD6 molecule	OR1A1	olfactory receptor, family 1, subfamily A, member 1
GOLGA8DP	golgin A8 family, member D, pseudogene	SRRM4	serine/arginine repetitive matrix 4
PTGR2	prostaglandin reductase 2	IL32	interleukin 32
Cancer Type: HNSC		**Cancer Type**: KIRC	
Symbol	Description	Symbol	Description
MRPL48	mitochondrial ribosomal protein L48	ITGB4	integrin, beta 4
MAP2K1	mitogen-activated protein kinase kinase 1	CCT8L2	chaperonin containing TCP1, subunit 8 (theta)-like 2
NSUN4	NOP2/Sun domain family, member 4	TP73-AS1	TP73 antisense RNA 1
OVOL1	ovo-like zinc finger 1	PGM1	phosphoglucomutase 1
SSX9	synovial sarcoma, X breakpoint 9	MEPE	matrix extracellular phosphoglycoprotein
INSIG1	insulin induced gene 1	CXorf40B	chromosome X open reading frame 40B
SP9	Sp9 transcription factor	BANK1	B-cell scaffold protein with ankyrin repeats 1
DDIT4	DNA-damage-inducible transcript 4	C10orf120	chromosome 10 open reading frame 120
EPN2	epsin 2	PRDX3	peroxiredoxin 3
EWSR1	EWS RNA-binding protein 1	C10orf91	chromosome 10 open reading frame 91

Table 6.6: Top 10 gene expression features obtained for each cancer type using **Cox-$l_{2,1}$** model. (Cont.)

Cancer Type: LAML		**Cancer Type**: LUSC	
Symbol	Description	Symbol	Description
NPIPB15	nuclear pore complex interacting protein family, #B15	C10orf76	chromosome 10 open reading frame 76
LOC285696	uncharacterized LOC285696	APOA1BP	apolipoprotein A-I binding protein
GSTM1	glutathione S-transferase mu 1	MIR1267	microRNA 1267
IGKV2-24	immunoglobulin kappa variable 2-24	MIR509-2	microRNA 509-2
C2orf83	chromosome 2 open reading frame 83	IGLC7	immunoglobulin lambda constant 7
MIR1255A	microRNA 1255a	MIR1251	microRNA 1251
MTFR1L	mitochondrial fission regulator 1-like	P3H2	prolyl 3-hydroxylase 2
CCKBR	cholecystokinin B receptor	IMP4	IMP4, U3 small nucleolar ribonucleoprotein
SULT4A1	sulfotransferase family 4A, member 1	MAPKAPK3	mitogen-activated protein kinase-activated protein kinase 3
LINC00313	long intergenic non-protein coding RNA 313	CNTROB	centrobin, centrosomal BRCA2 interacting protein

Cancer Type: LUAD		**Cancer Type**: OV	
Symbol	Description	Symbol	Description
MIR655	microRNA 655	NADSYN1	NAD synthetase 1
GATC	glutamyl-tRNA(Gln) amidotransferase, subunit C	DNMT1	DNA (cytosine-5-)-methyltransferase 1
ZNF419	zinc finger protein 419	VMO1	vitelline membrane outer layer 1 homolog (chicken)
TRIM34	tripartite motif containing 34	OR2A1	olfactory receptor, family 2, subfamily A, member 1
PAK1IP1	PAK1 interacting protein 1	ST14	suppression of tumorigenicity 14 (colon carcinoma)
C11orf72	chromosome 11 open reading frame 72	FCRL4	Fc receptor-like 4
KLRC2	killer cell lectin-like receptor subfamily C, member 2	LRRTM1	leucine rich repeat transmembrane neuronal 1
WNT7A	wingless-type MMTV integration site family, # 7A	RFX7	regulatory factor X, 7
MST1P2	macrophage stimulating 1 (hepatocyte growth factor-like) pseudogene 2	SPRED1	sprouty-related, EVH1 domain containing 1
AP2S1	adaptor-related protein complex 2, sigma 1 subunit	BET1L	Bet1 golgi vesicular membrane trafficking protein-like

6.4.4 Biomarker Discovery

Biomarkers are important indicators used to diagnose a particular disease in a clinical setting. From the clinical perspective, it is known that different cancer types should share some common significant biomarkers such as one particular anticarcinogen, chemotherapy dose, and radiotherapy dose. However, at the genetic level, the preventable factors and reflections of different cancer types are clearly different. In Table 6.5 and Table 6.6, we show a list of top 10 gene expression features for each cancer type based on their contributions (coefficient weights) in the **Cox-$l_{2,1}$** model and find that most of the top-ranked gene expression features are usually related to the genetics of the corresponding cancer types. For example, in BRCA, CD6 is heterotypic adhesion with activated leukocyte cell adhesion molecule which is in breast cancer lines acting in melanoma tumor progression and resected breast tumors [40]. In GBM, AK5 affects the cyclophilin B depletion on GBM cell line[2]. In HNSC, the content of MRPL48 is one of the high expression genes to influence the HNSC as one essential mitochondrial ribosomal protein[3]. ITGB4 is the high expression of cell adhesion models as heterogeneous immunohistochemical feature in KIRC [1]. The lack of GSTM1 will increase the risk of LAML when GSTM1 gene is null genotype among LAML patients [4]. For LUSC, APOA1BP is in human serum, cerebrospinal and spinal fluid to help body's transport and metabolism related to lung cancer cell lines reported in human body fluids in pathological conditions [84]. For LUAD, MIR655 is one of the discovered class of small RNS which is linked to the development and progression of cancer in lung [79]. For OV, ST14 is in the papillary serous subtype of ovarian tumors reported by cytogenetic analysis of primary ovarian carcinomas and ovarian cancer cell lines [76]. It should be noted that some of the remaining features listed can be strong potential candidates for further biological testing for generat-

[2] `http://www.ncbi.nlm.nih.gov/geoprofiles/104754733`
[3] `http://amp.pharm.mssm.edu/Harmonizome/gene/MRPL48`

ing new hypotheses in the future. From this analysis, it is clear that the proposed **Cox-$l_{2,1}$** algorithm not only provides better results in an efficient manner, but also inherently provides insights about the critical features for further analysis.

CHAPTER 7 CONCLUSION

In this thesis, we briefly review the basic concepts in survival analysis and comprehensively go through multiple categories of survival prediction methods. We analyze the weakness of commonly used survival prediction methods and propose several new regularized regression methods for high-dimensional survival analysis.

By using the notion that the latent survival time of censored instances should be no earlier than censored time, we proposed a weighted scheme, Regularized Weighted Residual Sum-of-Squares (**RWRSS**), which induces more penalty for the incorrectly predicted censored instances. The elastic net penalty is used to induce sparseness into the resulting coefficients thus alleviate over-fitting the data especially in high-dimensional datasets. In addition, we also developed a self-training framework for censored regression based on this linear model.

We developed a unified model, unified model for Regularized Parametric Censored Regression (**URPCR**), for regularized parametric censored regression that is able to efficiently handle high-dimensional (right) censored data. In order to unify the learning scheme for various popular distributions, we used Taylor expansion to approximate the objective function as a generalized linear model and solved the penalized iterative reweighted least squares problem via a cyclical coordinate descent-based method.

In addition, we also formulated the survival analysis problem as a multi-task learning problem and proposed a new multi-task learning algorithm, Multi-Task Learning model for Survival Analysis (**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.

Finally, we developed a novel transfer learning model for survival analysis. The proposed **COX-$l_{2,1}$** is a regularized Cox regression model that is able to efficiently select common hidden features in high-dimensional (right) censored data to transfer knowledge from the source domain to the target domain. The $l_{2,1}$-norm penalty is used to induce common sparseness into both source and target domains thus learning a shared low-dimensional feature representation for knowledge transfer and alleviating over-fitting the data, especially in high-dimensional scenarios. In order to speedup the learning scheme, we use the idea of screening and extend the strong rule to the proposed **COX-$l_{2,1}$** model. Thus, our model is able to efficiently identify most of the inactive features, and the computational cost of learning **COX-$l_{2,1}$** is dramatically reduced by the removal of the inactive features in the training phase.

We extensively compared the performance of the proposed algorithms 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.

BIBLIOGRAPHY

[1] Andreadis, D., Nomikos, A., Barbatis, C.: Metastatic renal clear cell carcinoma in the parotid gland: a study of immunohistochemical profile and cell adhesion molecules (cams) expression in two cases. Pathology & Oncology Research **13**(2), 161–165 (2007)

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

[3] Arnold, A., Nallapati, R., Cohen, W.W.: A comparative study of methods for transductive transfer learning. In: Data Mining Workshops, 2007. ICDM Workshops 2007. Seventh IEEE International Conference on, pp. 77–82. IEEE (2007)

[4] Arruda, V.R., Lima, C.S.P., Grignoli, C.R.E., De Melo, M.B., Lorand-Metze, I., Alberto, F.L., Saad, S.T.O., Costa, F.F.: Increased risk for acute myeloid leukaemia in individuals with glutathione s-transferase mu 1 (gstm1) and theta 1 (gstt1) gene defects. European journal of haematology **66**(6), 383–388 (2001)

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

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

[7] Binder, H., Schumacher, M.: Allowing for mandatory covariates in boosting estimation of sparse high-dimensional survival models. BMC bioinformatics **9**(1), 14 (2008)

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

[9] Breiman, L.: Bagging predictors. Machine learning **24**(2), 123–140 (1996)

[10] Breiman, L.: Random forests. Machine learning **45**(1), 5–32 (2001)

[11] Breslow, N.E.: Contribution to the discussion of the paper by dr cox. Journal of the Royal Statistical Society, Series B **34**(2), 216–217 (1972)

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

[13] Chambless, L.E., Diao, G.: Estimation of time-dependent area under the roc curve for long-term risk prediction. Statistics in medicine **25**(20), 3474–3486 (2006)

[14] Cho, H.J., Hong, S.M.: Median regression tree for analysis of censored survival data. Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on **38**(3), 715–726 (2008)

[15] Ciampi, A., Bush, R.S., Gospodarowicz, M., Till, J.E.: An approach to classifying prognostic factors related to survival experience for non-hodgkin's lymphoma patients: Based on a series of 982 patients: 1967–1975. Cancer **47**(3), 621–627 (1981)

[16] Ciampi, A., Chang, C.H., Hogg, S., McKinney, S.: Recursive partition: A versatile method for exploratory-data analysis in biostatistics. In: Biostatistics, pp. 23–50. Springer (1986)

[17] Ciampi, A., Thiffault, J., Nakache, J.P., Asselain, B.: Stratification by stepwise regression, correspondence analysis and recursive partition: a comparison of three methods of analysis for survival data with covariates. Computational statistics & data analysis **4**(3), 185–204 (1986)

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

[19] Crowther, M.J., Lambert, P.C.: A general framework for parametric survival

analysis. Statistics in medicine **33**(30), 5280–5297 (2014)

[20] Cutler, S.J., Ederer, F.: Maximum utilization of the life table method in analyzing survival. Journal of chronic diseases **8**(6), 699–712 (1958)

[21] Dai, W., Yang, Q., Xue, G.R., Yu, Y.: Boosting for transfer learning. In: Proceedings of the 24th international conference on Machine learning, pp. 193–200. ACM (2007)

[22] Davis, R.B., Anderson, J.R.: Exponential survival trees. Statistics in Medicine **8**(8), 947–961 (1989)

[23] Dunn, O.J., Clark, V.A.: Basic statistics: a primer for the biomedical sciences. Wiley. com (2009)

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

[25] Friedman, J., Hastie, T., Tibshirani, R.: Regularization paths for generalized linear models via coordinate descent. Journal of statistical software **33**(1), 1 (2010)

[26] Garcke, J., Vanck, T.: Importance weighted inductive transfer learning for regression. In: Machine Learning and Knowledge Discovery in Databases, pp. 466–481. Springer (2014)

[27] Gordon, L., Olshen, R.A.: Tree-structured survival analysis. Cancer treatment reports **69**(10), 1065 (1985)

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

[29] Hastie, T., Tibshirani, R., Friedman, J.J.H.: The elements of statistical learning, vol. 1. Springer New York (2001)

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

(2012)

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

[32] Hoerl, A.E., Kennard, R.W.: Ridge regression: Biased estimation for nonorthogonal problems. Technometrics **12**(1), 55–67 (1970)

[33] Hothorn, T., Lausen, B., Benner, A., Radespiel-Tröger, M.: Bagging survival trees. Statistics in medicine **23**(1), 77–91 (2004)

[34] van Houwelingen, H., Putter, H.: Dynamic Prediction in Clinical Survival Analysis. CRC Press, Inc. (2011)

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

[36] Ishwaran, H., Kogalur, U.B., Blackstone, E.H., Lauer, M.S.: Random survival forests. The Annals of Applied Statistics pp. 841–860 (2008)

[37] Jebara, T.: Multi-task feature and kernel selection for svms. In: Proceedings of the twenty-first international conference on Machine learning, p. 55. ACM (2004)

[38] Kandoth, C., McLellan, M.D., Vandin, F., Ye, K., Niu, B., Lu, C., Xie, M., Zhang, Q., McMichael, J.F., Wyczalkowski, M.A., et al.: Mutational landscape and significance across 12 major cancer types. Nature **502**(7471), 333–339 (2013)

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

[40] King, J.A., Ofori-Acquah, S.F., Stevens, T., Al-Mehdi, A.B., Fodstad, O., Jiang, W.G.: Activated leukocyte cell adhesion molecule in breast cancer: prognostic indicator. Breast Cancer Res **6**(5), R478–R487 (2004)

[41] Klein, J.P., Zhang, M.J.: Survival analysis, software. Wiley Online Library (2005)

[42] LeBlanc, M., Crowley, J.: Relative risk trees for censored survival data. Biometrics pp. 411–425 (1992)

[43] Lee, E.T., Wang, J.: Statistical methods for survival data analysis, vol. 476. Wiley. com (2003)

[44] Li, Y., Rakesh, V., Reddy, C.K.: Project success prediction in crowdfunding environments. In: Proceedings of the Ninth ACM International Conference on Web Search and Data Mining, WSDM '16, pp. 247–256 (2016). URL `http://doi.acm.org/10.1145/2835776.2835791`

[45] Li, Y., Vinzamuri, B., Reddy, C.K.: Constrained elastic net based knowledge transfer for healthcare information exchange. Data Mining and Knowledge Discovery **29**(4), 1094–1112 (2015)

[46] Li, Y., Vinzamuri, B., Reddy, C.K.: Regularized weighted linear regression for high-dimensional censored data. In: Proceedings of SIAM International Conference on Data Mining, pp. 45–53 (2016)

[47] Li, Y., Wang, J., Ye, J., Reddy, C.K.: A multi-task learning formulation for survival analysis. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16 (2016)

[48] Li, Y., Xu, K.S., Reddy, C.K.: Regularized parametric regression for high-dimensional survival analysis. In: Proceedings of SIAM International Conference on Data Mining, pp. 765–773 (2016)

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

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

[51] Liu, J., Sun, L., Ye, J.: Projection onto a nonnegative max-heap. In: Advances in Neural Information Processing Systems, pp. 487–495 (2011)

[52] Marubini, E., Valsecchi, M.G.: Analysing survival data from clinical trials and observational studies; e. marubini & mg valsecchi published by john wiley & sons 414 pages isbn 0–971-93987-0. British Journal of Clinical Pharmacology **41**(1), 76–76 (1996)

[53] Mayr, A., Schmid, M.: Boosting the concordance index for survival data–a unified framework to derive and evaluate biomarker combinations. PloS one **9**(1), e84,483 (2014)

[54] Miller Jr, R.G.: Survival analysis, vol. 66. John Wiley & Sons (2011)

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

[56] Padhukasahasram, B., Reddy, C.K., Li, Y., Lanfear, D.E.: Joint impact of clinical and behavioral variables on the risk of unplanned readmission and death after a heart failure hospitalization. PloS one **10**(6), e0129,553 (2015)

[57] Pan, S.J., Tsang, I.W., Kwok, J.T., Yang, Q.: Domain adaptation via transfer component analysis. Neural Networks, IEEE Transactions on **22**(2), 199–210 (2011)

[58] Pan, S.J., Yang, Q.: A survey on transfer learning. Knowledge and Data Engineering, IEEE Transactions on **22**(10), 1345–1359 (2010)

[59] Pan, W., Xiang, E.W., Liu, N.N., Yang, Q.: Transfer learning in collaborative filtering for sparsity reduction. In: Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI (2010)

[60] Pardoe, D., Stone, P.: Boosting for regression transfer. In: Proceedings of the 27th international conference on Machine learning (ICML-10), pp. 863–870 (2010)

[61] Raina, R., Battle, A., Lee, H., Packer, B., Ng, A.Y.: Self-taught learning: transfer learning from unlabeled data. In: Proceedings of the 24th international conference on Machine learning, pp. 759–766. ACM (2007)

[62] Reddy, C.K., Li, Y.: A review of clinical prediction models. In: C.K. Reddy, C.C. Aggarwal (eds.) Healthcare Data Analytics. Chapman and Hall/CRC Press (2015)

[63] Rosenwald, A., Wright, G., Wiestner, A., Chan, W.C., Connors, J.M., Campo, E., Gascoyne, R.D., Grogan, T.M., Muller-Hermelink, H.K., Smeland, E.B., 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)

[64] Segal, M.R.: Regression trees for censored data. Biometrics pp. 35–47 (1988)

[65] Simon, N., Friedman, J., Hastie, T., Tibshirani, R.: Regularization paths for coxs proportional hazards model via coordinate descent. Journal of statistical software **39**(5), 1–13 (2011)

[66] Sørlie, T., Tibshirani, R., Parker, J., Hastie, T., Marron, J., Nobel, A., Deng, S., Johnsen, H., Pesich, R., Geisler, S., 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)

[67] Therneau, T.: A package for survival analysis in s. r package version 2.37-4. URL http://CRAN. R-project. org/package= survival. Box **980032**, 23,298–0032 (2013)

[68] Tibshirani, R.: Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society. Series B (Methodological) pp. 267–288 (1996)

[69] Tibshirani, R., Bien, J., Friedman, J., Hastie, T., Simon, N., Taylor, J., Tibshirani, R.J.: Strong rules for discarding predictors in lasso-type problems. Journal

of the Royal Statistical Society: Series B (Statistical Methodology) **74**(2), 245–266 (2012)

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

[71] Tobin, J.: Estimation of relationships for limited dependent variables. Econometrica: journal of the Econometric Society pp. 24–36 (1958)

[72] Uno, H., Cai, T., Pencina, M.J., D'Agostino, R.B., Wei, L.: On the c-statistics for evaluating overall adequacy of risk prediction procedures with censored survival data. Statistics in medicine **30**(10), 1105–1117 (2011)

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

[74] Verweij, P.J.M., Van Houwelingen, H.C.: Penalized likelihood in cox regression. Statistics in Medicine **13**(23-24), 2427–2436 (1994)

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

[76] Wan, M., Sun, T., Vyas, R., Zheng, J., Granada, E., Dubeau, L.: Suppression of tumorigenicity in human ovarian cancer cell lines is controlled by a 2 cm fragment in chromosomal region 6q24-q25. Oncogene **18**(8), 1545–1551 (1999)

[77] Wang, J., Zhou, J., Wonka, P., Ye, J.: Lasso screening rules via dual polytope projection. In: Advances in Neural Information Processing Systems, pp. 1070–1078 (2013)

[78] Wang, S., Nan, B., Zhu, J., Beer, D.G.: Doubly penalized buckley–james method

for survival data with high-dimensional covariates. Biometrics **64**(1), 132–140 (2008)

[79] Wang, Y., Zang, W., Du, Y., Ma, Y., Li, M., Li, P., Chen, X., Wang, T., Dong, Z., Zhao, G.: Mir-655 up-regulation suppresses cell invasion by targeting pituitary tumor-transforming gene-1 in esophageal squamous cell carcinoma. J Transl med **11**, 301 (2013)

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

[81] Wei, L.: 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)

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

[83] Ye, J., Liu, J.: Sparse methods for biomedical data. ACM SIGKDD Explorations Newsletter **14**(1), 4–15 (2012)

[84] Yousefi, Z., Sarvari, J., Nakamura, K., Kuramitsu, Y., Ghaderi, A., Mojtahedi, Z.: Secretomic analysis of large cell lung cancer cell lines using two-dimensional gel electrophoresis coupled to mass spectrometry. Folia Histochemica et Cytobiologica **50**(3), 368–374 (2012)

[85] Zhou, J., Chen, J., Ye, J.: MALSAR: Multi-tAsk Learning via StructurAl Regularization. Arizona State University (2011). URL `http://www.public.asu.edu/~jye02/Software/MALSAR`

[86] Zou, H., Hastie, T.: Regularization and variable selection via the elastic net. Journal of the Royal Statistical Society: Series B (Statistical Methodology) **67**(2), 301–320 (2005)

ABSTRACT

NOVEL REGRESSION MODELS FOR HIGH-DIMENSIONAL SURVIVAL ANALYSIS

by

Yan Li

August 2016

Advisor: Dr.Chandan Reddy

Major: Computer Science

Degree: Doctor of Philosophy

Survival analysis aims to predict the occurrence of specific events of interest at future time points. The presence of incomplete observations due to *censoring* brings unique challenges in this domain and differentiates survival analysis techniques from other standard regression methods. In this thesis, we propose *four* models for survival analysis in high-dimensional data. Firstly, we propose a regularized linear regression model with weighted least-squares to handle the survival prediction in the presence of censored instances. We employ the elastic net penalty term for inducing sparsity into the linear model to effectively handle high-dimensional data. As opposed to the existing censored linear models, the parameter estimation of our model does not need any prior estimation of survival times of censored instances. The second model we proposed is a *unified* model for regularized parametric survival regression for an arbitrary survival distribution. We employ a generalized linear model to approximate the negative log-likelihood and use the elastic net as a sparsity-inducing penalty to effectively deal with high-dimensional data. The proposed model is then formulated as a penalized iteratively reweighted least squares and solved using a cyclical coordinate descent-based method. The widely 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 many real-world applications. To address these drawbacks, we reformulate the survival analysis problem as a multi-task learning problem in the third model which predicts 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. The proposed formulation is solved via an Alternating Direction Method of Multipliers (ADMM) based algorithm. Besides the above three methods which aim at solving standard survival prediction problem, we also propose a transfer learning model for survival analysis. Obtaining sufficient labeled training instances for learning a robust prediction model is a very time consuming process and can be extremely difficult in practice. To tackle this problem, we also proposed a Cox based model which uses the $l_{2,1}$-norm penalty to encourage source predictors and target predictors share similar sparsity patterns and hence learns a shared representation across source and target domains to improve the model performance on the target task. We demonstrate the performance of the proposed models using several real-world high-dimensional biomedical benchmark datasets and our experimental results indicate that our models outperform other state-of-the-art related competing methods and attain very competitive performance on various datasets.

AUTOBIOGRAPHICAL STATEMENT

Yan Li

Yan Li is currently a Ph.D. candidate working as a research assistant at the Data Mining and Knowledge Discovery (DMKD) Laboratory in Department of Computer Science, Wayne State University, directed by Prof. Chandan K. Reddy. He received his M.S. in Computer Science from WSU in 2014. Before joining WSU, he got his Bachelor of Science degree in Electronic Information Engineering from XiDian University (Xi'an, China) in 2011. His primary research interests are Data Mining and Machine Learning with applications to Healthcare Analytics and Bioinformatics. His research works have been published in leading conferences and journals including SIGKDD, WSDM, SDM, CIKM, DMKD, and Information Sciences.