Journal of Biomedical Informatics 100 (2019) 103326

Contents lists available at ScienceDirect

Journal of Biomedical Informatics

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

Time-to-event estimation by re-defining time

Xi Hang Cao[a], Chao Han[a], Lucas M. Glass[b], Allen Kindman[c], Zoran Obradovic[a,*]

[a] Center for Data Analytics and Biomedical Informatics, Temple University, 386 SERC, 1925 N. 12th St., Philadelphia, PA 19122, USA
[b] IQVIA, 1 IMS Drive Plymouth Meeting, PA 19462, USA
[c] IQVIA, 4820 Emperor Blvd, Durham, NC 27703, USA

ARTICLE INFO

Keywords:
Time-to-event estimation
Survival analysis
Representation learning
Concept embedding
Knowledge discovery
Regime identification

ABSTRACT

The primary goal of a time-to-event estimation model is to accurately infer the occurrence time of a target event. Most existing studies focus on developing new models to effectively utilize the information in the censored observations. In this paper, we propose a model to tackle the time-to-event estimation problem from a completely different perspective. Our model relaxes a fundamental constraint that the target variable, time, is a univariate number which satisfies a partial order. Instead, the proposed model interprets each event occurrence time as a time concept with a vector representation. We hypothesize that the model will be more accurate and interpretable by capturing (1) the relationships between features and time concept vectors and (2) the relationships among time concept vectors. We also propose a scalable framework to simultaneously learn the model parameters and time concept vectors. Rigorous experiments and analysis have been conducted in medical event prediction task on seven gene expression datasets. The results demonstrate the efficiency and effectiveness of the proposed model. Furthermore, similarity information among time concept vectors helped in identifying time regimes, thus leading to a potential knowledge discovery related to the human cancer considered in our experiments.

1. Introduction

The primary goal of a time-to-event estimation model is to accurately infer the occurrence time of a target event. Time-to-event data analysis has been an active research topic due to its tremendous application values in a variety of disciplines including biology, healthcare, engineering, economics, and sociology [1]. Time-to-event estimation is also called survival analysis [2], reliability analysis, duration modeling, and event history analysis. The most unique characteristic of the time-to-event estimation problem is the presence of censored examples in the data. A censored example is an example whose event occurrence time is unobserved due to observation window limits or losing track during the observation window. The most common censoring cases are left censoring and right censoring. In the left censoring case, the event occurs before the beginning (left edge) of the observation window; similarly, in the right censoring case, the event occurs after the end (right edge) of the observation window. In this paper, we only consider the right censoring case.

Because of the uniqueness of the time-to-event data, most existing models focus on developing algorithms for effectively extracting information from the censored examples. Recently, machine learning methods have been adopted for time-to-event modeling [3], for instance, Survival Tree Models [4–6], Support Vector Machine for censored data [7–9], Random Survival Forest [10], and Survival Boosting Trees [11]. Highly innovative new approaches have been proposed; for example, active learning [12], transfer learning [13], multi-task survival analysis [14,15], deep survival analysis [16] and adversarial learning [17].

In this paper, we approach the time-to-event estimation problem from a different perspective. In a conventional time-to-event problem formulation, the target variable *time* is univariate and satisfies a partial order; i.e., if $t_1 < t_2 < t_3$, then $t_2 - t_1 < t_3 - t_1$. In our model, instead of directly using *time* as the target variable, we propose to treat each discretized event occurrence time as a concept with a vector representation (called a time concept vector), and use the vector representation to replace *time* as the target. Our hypothesis is that an event occurrence time should be treated as a word/concept which carries an abstract meaning. Therefore, instead of univariate numbers, multidimensional vectors are more suitable for capturing the similarities among the concepts representing the event occurrence times. By effectively exploiting the relations between features and time concepts, we can obtain a more accurate model for time-to-event estimation.

* Corresponding author.
E-mail addresses: xi.hang.cao@temple.edu (X.H. Cao), chao.han@temple.edu (C. Han), lucas.glass@iqvia.com (L.M. Glass), allen.kindman@iqvia.com (A. Kindman), zoran.obradovic@temple.edu (Z. Obradovic).

https://doi.org/10.1016/j.jbi.2019.103326
Received 14 May 2019; Received in revised form 5 September 2019; Accepted 28 October 2019
Available online 31 October 2019
1532-0464/ © 2019 Elsevier Inc. All rights reserved.

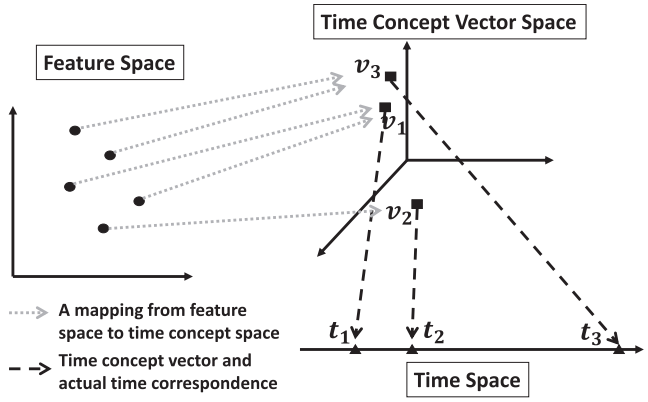

Fig. 1. The proposed model first determines the corresponding time concept of each example and then infers the event occurrence time.

Furthermore, the similarity among time concept vectors may reveal information for knowledge discovery; for example, time regime identification. The high-level idea of the proposed model is illustrated in Fig. 1. In the figure, each example is represented by a dot in a 2D feature space, each time concept is represented by a dot in a 3D vector space, and each event occurrence time is represented in a 1D time space (time line). Instead of learning a model to directly assign each example an event occurrence time (i.e. $\mathbb{R}^2 \rightarrow \mathbb{R}$), the proposed model assigns each example a time concept (i.e. $\mathbb{R}^2 \rightarrow \mathbb{R}^3$), and then the actual event time is inferred by the known concept-time correspondence. Because the time concept vector of an event occurrence time is unknown *a priori*, we developed a framework to learn the model parameters and the time concept vectors jointly. The contributions of our paper are summarized as follows:

- We propose a new time-to-event estimation model which indirectly infers event occurrence times via time concept vectors.
- We develop an efficient framework to learn model parameters and time concept vectors jointly.
- We conduct rigorous experiments to demonstrate the effectiveness, interpretability, and scalability of the proposed model.

2. Related work

The time-to-event model (or survival analysis model) is one of the most fruitful research topics in statistics. For a comprehensive survey on this topic, one can refer to [3] and references therein. In this section, we will briefly summarize some of the most widely used models.

The Cox proportional hazard model [18] is one of the earliest and one of the most influential models in the topic of survival analysis. The Cox model is considered as a semi-parametric model because no assumptions are made about the nature or shape of the baseline hazard function [19]. The parameters in the Cox model are determined by optimizing a partial likelihood function. Models which use the partial likelihood as the optimization objective are considered as Cox-based models. The basic formulation of the Cox model is highly subject to overfitting in high-dimensional data. To address this shortcoming, variants of the Cox model based on different regularizations are proposed; for example, LASSO-Cox [20], is a Cox-based model with an L_1-norm regularization, and EN-Cox [21] is a Cox-based model with the elastic net regularization. One of the drawbacks of the Cox-based models is that they do not infer the event occurring time; instead, they infer a hazard score, which is usually used for ranking. In contrast, the proposed model directly infers the time vector which relates to an exact event occurring time.

Parametric models are another class of widely used survival analysis models. In parametric models, the event occurrence time is usually assumed to satisfy an underlying distribution of which the density, survivorship, hazard and cumulative hazard functions are easy to derive. The model parameters are then determined by optimizing a likelihood function based on the given data. Some popular choices of the underlying distributions include Logistic, Weibull, Log-Gaussian, and Log–Logistic [22]. The assumption about the underlying distribution may potentially limit the performance of the model. On the other hand, the proposed model does not have any assumption on the distribution of the target, and thus it is more flexible to capture to characteristics from data, leading to better performance.

The time-to-event problem is similar to the traditional regression problem in the sense that the target variable is continuous. However, the traditional regression model cannot be directly applied to the time-to-event problem due to the existence of censored examples. The Tobit model [23] is one of the earliest linear regression models which incorporate the censored examples in the optimization objective. The Buckley-James (BJ) regression model [24] handles the censored data by using an auxiliary Kaplan-Meier estimator [25]. The variant, BJ-EN, is proposed in [26] for high-dimensional data. One of the significant drawbacks of the regression-based models is that they tend to implicitly assume the target variable, the event occurring time, is normally distributed. The models are usually learned by minimizing a norm-based loss function. This assumption may limit the performance of the model. The proposed model does not assume the target variable distributions and thus avoids this drawback.

Machine learning techniques are adopted for time-to-event estimation problems [3]. A survival tree model was proposed in [4], in which the Wasserstin metric was used estimate the homogeneity and as the choice of splitting criterion. Support vector machines (SVM) were also adopted for survival analysis. In [7], an SVM-based model was proposed for censored data by using an updated asymmetirc loss function. In [8], an SVM-based model was proposed using a health index a proxy for the censored time. In [9], an SVM-based model was proposed to incorporate ranking and regression to solve the time-to-event estimation problem. Ensemble models were also adopted to solve the time-to-event estimation problem. In [10], a random survival forest was proposed to use survival trees [4] as week learners. In [11], a boosting algorithm was proposed to incorporate censored data. An active regularized cox regression model was proposed [12] to use an active learning algorithm to incorporate the Cox model by using a discriminative gradient sampling strategy. A transfer learning survival analysis model was proposed [13] to improve the Cox PH model by transferring knowledge from the source domain to the target domain in the context of survival analysis. The above models aim to improve the performance by mimicking different formulations in model architectures, loss function and regularizations. Recently, representation learning has been adopted for time-to-event estimation. In [16], a hierarchical generative approach to survival analysis in the context of the Electronic Health Records was proposed. In [17], a time-to-event model as proposed to focus on the estimation of time-to-event distributions by using generative adversarial approach. These two approaches tackle the time-to-event estimation task from the input representation learning perspective. The Multi-task learning method for survival analysis (MTLSA) [14] tackles the problem from an output representation approach by introducing an indicator matrix and learning a binary code (with special structures) for the multitask objective. While the proposed approach is similar to MTLSA in the sense that both are learning the model parameters and output representations simultaneously, in contrast to learning a special binary code, the proposed approach learns a distributed representation [27] for each event time point. Along with proper regularizations in the model parameters and time vectors, the proposed model is more capable of capturing the characteristics of the data so potentially is more accurate.

Table 1
Notations and definitions.

Notation	Definition
n	The number of examples in a dataset
m	The dimensionality of a feature vector
d	The dimensionality of a time vector
$\mathcal{T}$	$\mathcal{T} = \{t_1, t_2, \cdots, t_j, \cdots, t_{\|\mathcal{T}\|}\}$, the set of distinct time points in the dataset
$\boldsymbol{x}_i \in \mathbb{R}^m$	The feature vector of the i-th instance
$y_i \in \mathcal{T}$	The last observation time of the i-th example
$s_i \in \{0, 1\}$	The status of the i-th example in its last observation (0: no event; 1: event)
$\mathbf{V} \in \mathbb{R}^{d \times \|\mathcal{T}\|}$	A matrix whose j-th column vector, $\mathbf{V}_{:j} \in \mathbb{R}^d$, is the concept vector of time point, t_j
$\mathbf{W} \in \mathbb{R}^{d \times m}$	A linear transformation matrix

3. Method

3.1. Notations

In this paper, scalar variables are denoted by letters (e.g., t and N), vector variables are represented by boldface letters (e.g., $\boldsymbol{x}$), matrix variables are represented by boldface uppercase letters (e.g., $\mathbf{V}$). The j-th column vector and the i-th row vector of $\mathbf{V}$ are denoted by $\mathbf{V}_{:j}$ and $\mathbf{V}_{i:}$, respectively. We list the main symbols used in our subsequent derivations in Table 1.

3.2. Problem formulation

In the time-to-event model learning, each example in the dataset is represented by a triplet, $(\boldsymbol{x}_i, y_i, \boldsymbol{s}_i)$ with $i \in \{1, 2, \cdots, n\}$ being the index of the example and n being the number of examples. Within each triplet, $\boldsymbol{x}_i \in \mathbb{R}^m$ denotes an m-dimensional feature vector, $y_i \in \mathcal{T}$ denotes the last observation time, and $s_i \in \{0, 1\}$ denotes the status of the example at its last observation time. If $s_i = 1$, the target event occurs at time y_i and subsequent observations are not necessary. If $s_i = 0$, the target event has not occurred up to time y_i, and subsequent observations are not available; thus, we call this example censored, and the event may occur at anytime $t > y_i$. The symbol, $\mathcal{T} = \{t_1, t_2, \cdots, t_{|\mathcal{T}|}\}$ with $|\mathcal{T}| < \infty$, denotes the set of all possible event occurrence times. In this context, the time-to-event task is to learn a probabilistic model,

$$p(y|\boldsymbol{x}, \theta),$$

with model parameter, θ.

3.3. Objective Function Formulation

3.3.1. Time-to-event probability

In our model, we compute the time-to-event probability, i.e., the probability that "the target event occurs at time t_j", by

$$p(y = t_j|\boldsymbol{x}, \theta) = p(y = t_j|\boldsymbol{x}, \mathbf{W}, \mathbf{V}) = \frac{\exp(-\| \mathbf{W}\boldsymbol{x} - \mathbf{V}_{:j} \|_2^2)}{\sum_{j'=1}^{|\mathcal{T}|} \exp(-\| \mathbf{W}\boldsymbol{x} - \mathbf{V}_{:j'} \|_2^2)}, \tag{1}$$

with model parameter $\theta = \{\mathbf{W}, \mathbf{V}\}$, where $\mathbf{W} \in \mathbb{R}^{d \times m}$ denotes the linear transformation matrix which maps a vector from the m-dimensional feature space to the d-dimensional time concept vector space; $\mathbf{V} \in \mathbb{R}^{d \times |\mathcal{T}|}$ denotes a matrix whose j-th column vector, $\mathbf{V}_{:j}$, is the vector representation of the event occurrence time, t_j with $j = 1, 2, \cdots, |\mathcal{T}|$; $\|\cdot\|_2$ is the L-2 norm.

When an example is censored, the target event's exact occurrence time is unknown, therefore (1) is not appropriate for calculating the time-to-event probability. Instead, because we know the censoring time and the fact that the event has not occurred up to that time, we can model the probability of "the target event will occur after time t_j" by

$$p(y > t_j|\boldsymbol{x}, \theta) = p(y > t_j|\boldsymbol{x}, \mathbf{W}, \mathbf{V}) = \frac{\sum_{k=j+1}^{|\mathcal{T}|} \exp(-\| \mathbf{W}\boldsymbol{x} - \mathbf{V}_{:k} \|_2^2)}{\sum_{j'=1}^{|\mathcal{T}|} \exp(-\| \mathbf{W}\boldsymbol{x} - \mathbf{V}_{:j'} \|_2^2)}. \tag{2}$$

3.3.2. Optimization objective

For notation simplicity, q_i, defined by (2), is used to denote the event probability for the i-th example if it is censored, otherwise, p_i, defined by (1), is used to denote the event probability for the i-th example. Using the short-hand notations, p_i and q_i, the negative log-likelihood based on the dataset can be written as

$$\ell(\mathbf{W}, \mathbf{V}) = \sum_{i=1}^{n} -(\mathbb{1}(s_i = 1)\log p_i + \mathbb{1}(s_i = 0)\log q_i),$$

where $\mathbb{1}(\cdot)$ is an indicator function whose returned value is 1 if the condition is met, otherwise 0. It is important to note that the negative log-likelihood is a function of both the linear transformation matrix, $\mathbf{W}$, and the time concept vector matrix $\mathbf{V}$; therefore, minimizing the negative log-likelihood could provide an optimal solution for our model. To avoid overfitting and to enhance the generalization of the model, we apply several regularization terms (note: detailed discussions on the regularization terms are provided in the following section) and propose to solve the following minimization problem:

$$\underset{\mathbf{W},\mathbf{V}}{\text{minimize}} \; F(\mathbf{W}, \mathbf{V}) = \ell(\mathbf{W}, \mathbf{V}) + \lambda_1 \|\mathbf{W}\|_{2,1} + \lambda_2 \|\mathbf{V}\|_* + \frac{\lambda_3}{2}\|\mathbf{V}\mathbf{D}\|_F^2, \tag{3}$$

in which, $\|\cdot\|_{2,1}$ denotes the $L_{2,1}$-norm, $\|\cdot\|_*$ denotes the nuclear norm, $\|\cdot\|_F$ denotes the Frobenius norm, and $\{\lambda_1, \lambda_2, \lambda_3\}$ are the trade-off parameters. The definition of the matrix, $\mathbf{D}$, will be given in the following section.

3.3.3. Regularization

The $L_{2,1}$-norm is known for inducing column sparsity. Therefore, in addition to preventing overfitting, the $L_{2,1}$-norm regularizer also acts as a feature-selection mechanism. In practice, especially in the time-to-event estimation tasks in the medical domain, the feature vectors are often very high-dimensional. Being able to select the most relevant features can not only improve the performance but also enhance the interpretability of the model.

The nuclear norm is a low-rank matrix inducing regularizer. It has a high utility value in our model. First, because the time concept vector dimensionality is unknown in our optimization algorithm, we intend to over-estimate the dimensionality in initialization. Having a low-rank inducing regularizer could help determine the effective dimensionality. Second, the nuclear norm regularizer may help generate structures in the time concept vector space; i.e., similar time concepts vectors reside in the same linear subspace.

The Frobenius norm is used to enhance the temporal smoothness among time concept vectors. The time concept vectors corresponding to two consecutive time points are similar, and thus the cumulative differences of all consecutive time points should be small. The cumulative difference square is represented by

$$\sum_{j=1}^{|\mathcal{T}|-1} \|\mathbf{V}_{:j} - \mathbf{V}_{:j+1}\|_2^2 = \|\mathbf{V}\mathbf{D}\|_F^2,$$

where the matrix $\mathbf{D} \in \mathbb{R}^{|\mathcal{T}| \times (|\mathcal{T}|-1)}$ with the (i, j)-th element being

$$D_{ij} = \begin{cases} 1, & i = j \\ -1, & i = j + 1 \\ 0, & \text{otherwise.} \end{cases}$$

3.4. Optimization

There are two challenges to optimize our proposed objective function: first, the objective function is non-smooth; therefore, conventional gradient-based approaches (e.g. conjugate gradient methods) are not applicable; second, multiple variables (i.e. $\mathbf{W}$ and $\mathbf{V}$) are required to optimize, and their terms are not separable.

To address these challenges in this optimization problem, we propose an iterative proximal algorithm. In our proposed algorithm, we formulate two non-smooth sub-problems: in the first sub-problem, the time concept vector matrix, $\mathbf{V}$, is treated as a constant, and a proximal algorithm is applied to solve the optimal $\mathbf{W}$; in the second problem the linear transformation matrix, $\mathbf{W}$, is treated as a constant, and a proximal algorithm is applied to solve the optimal $\mathbf{V}$. Then these two sub-problems are solved repeatedly until the overall objective converges. As the procedures for solving these two sub-problems are similar, we will give a detailed description on solving the first sub-problem, and a similar procedure can be used to solve the second sub-problem.

3.4.1. Sub-Problem 1: solving $\mathbf{W}$, with $\mathbf{V}$ as a constant

By treating $\mathbf{V}$ as a constant, the original optimization problem with objective function (3) is reduced to

$$\min_{\mathbf{W}} F_1(\mathbf{W}) = \ell(\mathbf{W}) + \lambda_1 ||\mathbf{W}||_{2,1}. \tag{4}$$

It is important to note that the objective function $F_1(\mathbf{W})$ is convex because its first term (a log-softmax function) and second term (a norm) are both convex. However, this problem is challenging due to the non-smoothness in the $L_{2,1}$-norm regularization. To solve this problem, we make use of an auxiliary function

$$Q(\mathbf{W}, \mathbf{W}') = \ell(\mathbf{W}') + tr((\mathbf{W} - \mathbf{W}')^\top \Delta\ell(\mathbf{W}')) + \frac{L}{2}||\mathbf{W} - \mathbf{W}'||_F^2 + \lambda_1 ||\mathbf{W}||_{2,1}. \tag{5}$$

Proposition 1. Let L(ℓ) denote the Lipschitz constant of $\ell(\mathbf{W})$, then for any L $\geqslant$L(ℓ), we have

$$F_1(\mathbf{W}) \leqslant Q(\mathbf{W}, \mathbf{W}').$$

This is straightforward to prove based on the convexity and the smoothness of $\ell(\mathbf{W})$ and the definition of the Lipschitz constant. It is also straightforward to verify

$$Q(\mathbf{W}', \mathbf{W}') = F_1(\mathbf{W}') \text{ and } Q(\mathbf{W}, \mathbf{W}) = F_1(\mathbf{W}).$$

Therefore, letting $\mathbf{W}' = \mathbf{W}^t$ and

$$\mathbf{W}^{t+1} = \text{argmin}_{\mathbf{W}} Q(\mathbf{W}, \mathbf{W}^t), \tag{6}$$

we arrive at the following relations:

$$F_1(\mathbf{W}^{t+1}) \leqslant Q(\mathbf{W}^{t+1}, \mathbf{W}^t) \leqslant Q(\mathbf{W}^t, \mathbf{W}^t) = F_1(\mathbf{W}^t).$$

Namely, (6) can be used as an updating rule for optimizing the objective function, $F_1(\mathbf{W})$. By completing the square in (5), we can find that (6) is equivalent to

$$\begin{aligned} \mathbf{W}^{t+1} = & \\ \text{argmin}_{\mathbf{W}} \frac{L}{2}||\mathbf{W} - (\mathbf{W}^t - \frac{1}{L}\Delta\ell(\mathbf{W}^t))||_F^2 + \lambda_1 ||\mathbf{W}||_{2,1} & \\ = \text{prox}_{\frac{\lambda_1}{L}||\cdot||_{2,1}}(\mathbf{W}^t - \frac{1}{L}\Delta\ell(\mathbf{W}^t)) & \end{aligned} \tag{7}$$

where $\text{prox}_{\frac{\lambda_1}{L}||\cdot||_{2,1}}$ is the proximal operator [28] of $\frac{\lambda_1}{L}||\cdot||_{2,1}$. Knowing the analytical form of the proximal operator for the scaled $L_{2,1}$-norm, $\alpha||\cdot||_{2,1}$, being

$$\text{prox}_{\alpha||\cdot||_{2,1}}(\mathbf{A}) = \mathbf{A}_{:j}(1 - \frac{\alpha}{||\mathbf{A}_{:j}||_2})_+, \quad \forall j, \tag{8}$$

and combining (7) and (8), we obtain the updating rule for $\mathbf{W}_{:j}^{t+1}$, $\forall j$

$$\mathbf{W}_{:j}^{t+1} = (\mathbf{W}^t - \frac{1}{L}\Delta\ell(\mathbf{W}^t))_{:j}(1 - \frac{\lambda_1}{L\,||\,(\mathbf{W}^t - \frac{1}{L}\Delta\ell(\mathbf{W}^t))_{:j}\,||_2})_+, \tag{9}$$

where $(\cdot)_+ = max(\cdot, 0)$. The procedure for solving sub-Problem 1 is described in Algorithm 1.

Algorithm 1. Sub-Problem 1: solving $\mathbf{W}$, while treating $\mathbf{V}$ as a constant

1: initialize $\mathbf{W}^0$
2: $t \leftarrow 0$
3: **do**
4: $\quad \mathbf{W}_{:j}^{t+1} \leftarrow (\mathbf{W}_{:j}^t - \frac{1}{L}\Delta\ell(\mathbf{W}^t))_{:j}(1 - \frac{\lambda_1}{L\,||\,(\mathbf{W}^t - \frac{1}{L}\Delta\ell(\mathbf{W}^t))_{:j}\,||_2})_+, \ \forall j$
5: $\quad t++$
6: **while** $t <$ max_iter **and not** converge
7: **Return** $\mathbf{W}$

3.4.2. Sub-Problem 2: solving $\mathbf{V}$, with $\mathbf{W}$ as a constant

By treating $\mathbf{W}$ as a constant, the original optimization problem with objective function (3) is reduced to

$$\min_{\mathbf{W}} F_2(\mathbf{V}) = \ell(\mathbf{V}) + \lambda_2 ||\mathbf{V}||_* + \frac{\lambda_3}{2}||\mathbf{V}\mathbf{D}||_F^2. \tag{10}$$

Following the derivations as in solving sub-Problem 1, we arrive at the updating rule of $\mathbf{V}$ in a proximal operator form:

$$\mathbf{V}^{t+1} = \text{prox}_{\frac{\lambda_2}{L}||\cdot||_*}(\mathbf{V}^t - \frac{1}{L}(\Delta\ell(\mathbf{V}^t) + \lambda_3 \mathbf{V}^t\mathbf{D}\mathbf{D}^\top)). \tag{11}$$

The proximal operator of a scaled nuclear norm [28], $\alpha||\cdot||_*$, is

$$\text{prox}_{\alpha||\cdot||_*}(\mathbf{A}) = \mathbf{U}\text{diag}(\text{prox}_{\alpha||\cdot||_1}(\sigma(\mathbf{A})))\mathbf{R}^\top, \tag{12}$$

where $\sigma(\mathbf{A})$ denotes the vector of singular values of a matrix, $\mathbf{A}$; $\{\mathbf{U}, \text{diag}(\sigma(\mathbf{A})), \mathbf{R}^\top\}$ is the singular value composition of $\mathbf{A}$, and

$$\text{prox}_{\alpha||\cdot||_1}(\boldsymbol{a}) = (\boldsymbol{a} - \alpha\boldsymbol{1})_+ - (-\boldsymbol{a} - \alpha\boldsymbol{1})_+ \tag{13}$$

is the proximal operator for the scaled vector L_1-norm. The symbol, $\boldsymbol{1}$, denotes a one-vector with an appropriate dimension. Combining (11)–(13), the updating rule of $\mathbf{V}$ can be written as

$$\mathbf{V}^{t+1} = \mathbf{U}\text{diag}((\sigma(\hat{\mathbf{V}}) - \frac{\lambda_2}{L}\boldsymbol{1})_+ - (-\sigma(\hat{\mathbf{V}}) - \frac{\lambda_2}{L}\boldsymbol{1})_+)\mathbf{R}^\top, \tag{14}$$

where $\{\mathbf{U}, \text{diag}(\sigma(\hat{\mathbf{V}})), \mathbf{R}^T\}$ is the singular value decomposition of $\mathbf{V}^t - \frac{1}{L}(\Delta\ell(\mathbf{V}^t) + \lambda_3\mathbf{V}^t\mathbf{D}\mathbf{D}^\top)$. The procedure of solving sub-Problem 2 is given in Algorithm 2.

Algorithm 2. Sub-Problem 2: solving $\mathbf{V}$, while treating $\mathbf{W}$ as a constant

1: initialize $\mathbf{V}^0$
2: $t \leftarrow 0$
3: **do**
4: $\quad \{\mathbf{U}, \text{diag}(\sigma(\hat{\mathbf{V}})), \mathbf{R}^\top\} \leftarrow SVD(\mathbf{V}^t - \frac{1}{L}(\Delta\ell(\mathbf{V}^t) + \lambda_3\mathbf{V}^t\mathbf{D}\mathbf{D}^\top))$
5: $\quad \mathbf{V}^{t+1} = \mathbf{U}\text{diag}((\sigma(\hat{\mathbf{V}}) - \frac{\lambda_2}{L}\boldsymbol{1})_+ - (-\sigma(\hat{\mathbf{V}}) - \frac{\lambda_2}{L}\boldsymbol{1})_+)\mathbf{R}^\top$
6: $\quad t++$
7: **while** $t <$ max_iter **and not** converge
8: **Return** $\mathbf{V}$

To solve the optimization problem with the objective function (3) involving both variables $\mathbf{W}$ and $\mathbf{V}$, we propose to use Algorithm 3, in which sub-Problems 1 and sub-Problem 2 are solved iteratively until a convergence is reached.

Algorithm 3. An iterative proximal algorithm for solving (3)

1: initialize $\mathbf{W}^0$ and $\mathbf{V}^0$
2: **repeat**
3: Solve sub-Problem 1 using Algorithm 1
4: Solve sub-Problem 2 using Algorithm 2
5: **until**converge
6: **Return** $\mathbf{W}$ and $\mathbf{V}$

4. Experiments

In this section, we describe our conducted experiments on the proposed model. The first experiment focuses on demonstrating the effectiveness of the proposed model by comparing the performance with those from state-of-the-art models on several benchmark datasets for the time-to-event estimation tasks. In the second experiment, we visualize the learned time concept vectors and show a few interesting observations. The third experiment focuses on empirically characterizing the scalability of the proposed model by running on a number of synthetic learning tasks in different settings.

4.1. Setup

The proposed model is implemented in Python 3.6.6 based on the *Numpy* package. We also use the *Autograd* package for computing the derivatives; i.e., $\Delta\ell(\mathbf{W}^t)$ in (7) and $\Delta\ell(\mathbf{V}^t)$ in (11). The code runs on a workstation with an Intel Core i7 CPU 4 GHz and 32 GB of RAM. Each of the hyper-parameters $\{\lambda_1, \lambda_2, \lambda_3\}$ were determined by grid searches in values {0, 0.1, 1.0} using cross-validation within the training set. The time concept vector dimensionality, d, is set to $d = 5|\mathcal{T}|$, where $\mathcal{T}$ is the set of event times in the training set. One should notice that parameter d is a number greater or equal to the effective dimension of the time vectors, and the effective dimension should be less or equal to $|\mathcal{T}|$; therefore, in our experiments, setting $d = 5|\mathcal{T}|$ is rather arbitrary because the effective dimensionality of the time concept vector will be automatically determined by the regularizations.

4.2. Experiment #1: comparisons to state-of-the-art models on benchmark datasets

In this experiment, we compare the performance, in terms of the concordance index, of the proposed model with the state-of-the-art baseline models on several benchmark datasets for the time-to-event estimation tasks.

4.2.1. Datasets

The 7 benchmark datasets used in this experiment are publicly available for download. Details information of the datasets and the task descriptions are provided as follows:

- *Van de Vijvers Microarray Breast Cancer data (VDV)*[29] contains 4,707 gene expression values on 78 (44 censored) breast cancer patients for predicting occurrence of death in terms of year, up to 13 years.
- *Gene-expression profiles of lung adenocarcinoma (Lung)* [30] contains 7,129 gene expression values on 86 (62 censored) early-stage lung adenocarcinoma patients for predicting occurrence of death in terms of month, up to 110 months.
- *Mantle Cell Lymphoma (MCL)*[31] contains 8,810 gene expression values on 92 (28 censored) MCL patients for predicting occurrence of death in terms of year, up to 14 years.

Table 2
Summaries of the 7 benchmark datasets (ordered by # example).

Dataset	example size	censored size	feature size	No. of distinct time
VDV	78	44	4705	13
Lung	86	62	7129	110
MCL	92	28	8810	14
NSBCD	115	77	549	188
AML	116	49	6283	54
DLBCL	240	102	7399	21
DBCD	295	216	4919	18

- *Norway/Stanford breast cancer data (NSBCD)* [32] contains 549 gene expression values on 115 (77 censored) breast cancer women for predicting occurrence of death in terms of month, up to 188 months.
- *Adult myeloid leukemia (AML)*[33] contains 6,283 gene expression values on 116 (77 censored) AML patients for predicting occurrence of death in terms of month, up to 54 months.
- *Diffuse Large B-Cell Lymphoma (DLBCL)* [34] contains 7,399 gene expression values on 240 (102 censored) DLBCL patients for predicting occurrence time of death in terms of year, up to 21 years.
- *Dutch Breast Cancer Data (DBCD)* [35] contains 4,919 gene expression values on 295 (216 censored) breast cancer women for predicting occurrence time of death in terms of year, up to 18 years.

The 7 benchmark datasets are summarized in Table 2. To have a fair comparisons to the baseline models, we adopt the evaluation settings in [14], in which 5-fold cross-validation is used when the number of examples is greater than 150 and 3-fold cross-validation otherwise.

4.2.2. Baseline models

Based on the popularity and accessibility, we compared our proposed model to 16 baseline models.

Cox based models. The Cox proportional hazards model [18] is the most commonly used survival analysis model. Besides the basic formulation, the L_1-norm regularized variant, *LASSO- Cox* [20], and the elastic net regularized variant, *EN-Cox* [21], are also widely used.

Censored regression models. Standard likelihood function estimation incorporates censored examples [22]. Based on the assumptions of the underlying distributions, it has four variants: Weibull, Logistic, Log-Logistic, and Log-Gaussian.

Linear models. Tobit model [23] is an extension of linear regression that incorporates censored examples with parameters estimated by the maximum likelihood method rather than using least squares error. Buckley-James regression [26] (BJ-EN), is a linear model which incorporates censored examples by estimating the censored value using the Kaplan-Meier estimator [25] with elastic net regularization.

Pairwise ranking based models. Boosting concordance index [36] (Boost-CI) is a gradient boosting algorithm to optimize the smoothened version of the concordance index.

Multi-task learning models. MTLSA is multi-task formulation tailored for handling censored examples [14] by introducing an indication table and a non-negative non-increasing list structure constraint.

Others. SurvGB [11] is a gradient boosting model for survival analysis. *SurvSVM-Linear* [37] is a rank and regression based SVM adoption for survival analysis using a linear kernel. *SurvSVM-RBF* [38] is an efficient SVM-based survival analysis model using the radial basis function kernel. *BJ-Neural* is a 2-layer neural network with Rectified Linear Unit Activation functions to optimize an objective of Buckley-James regression [26].

4.2.3. Performance metric

Due to the presence of censored examples, concordance index (CI or

Table 3
Concordance Indices (CIs) and corresponding standard deviations of all the models; The ranking of a model among all compared models in each dataset is included. The highest ranked method in each dataset is highlighted with boldface.

	VDV	Lung	MCL	NSBCD	AML	DLBCL	DBCD
CoxPH	0.597 (7) ± 0.011	0.516 (13) ± 0.133	0.577 (11) ± 0.059	0.441 (12) ± 0.059	0.552 (8) ± 0.068	0.455 (13) ± 0.072	0.554 (11) ± 0.123
CoxLasso	0.648 (4) ± 0.028	0.670 (3) ± 0.091	0.682 (8) ± 0.07	0.591 (10) ± 0.109	0.600 (4) ± 0.031	0.634 (4) ± 0.042	0.688 (8) ± 0.043
CoxEN	0.642 (5) ± 0.068	0.665 (4) ± 0.07	0.673 (9) ± 0.073	0.605 (9) ± 0.1	0.572 (6) ± 0.06	0.649 (3) ± 0.039	0.721 (4) ± 0.031
Logistic	0.528 (8) ± 0.14	0.571 (11) ± 0.094	0.483 (12) ± 0.068	0.379 (13) ± 0.02	0.454 (15) ± 0.077	0.484 (12) ± 0.05	0.491 (13) ± 0.087
Weibull	0.316 (16) ± 0.132	0.429 (15) ± 0.01	0.474 (14) ± 0.075	0.305 (15) ± 0.153	0.529 (12) ± 0.055	0.251 (16) ± 0.063	0.456 (16) ± 0.105
Log-Gaussian	0.521 (10) ± 0.165	0.412 (16) ± 0.075	0.256 (16) ± 0.072	0.444 (11) ± 0.054	0.405 (16) ± 0.065	0.317 (15) ± 0.091	0.488 (14) ± 0.055
Log–Logistic	0.527 (9) ± 0.107	0.592 (9) ± 0.066	0.480 (13) ± 0.072	0.238 (16) ± 0.05	0.468 (14) ± 0.08	0.425 (14) ± 0.124	0.526 (12) ± 0.023
Tobit	0.519 (11) ± 0.158	0.469 (14) ± 0.136	0.459 (15) ± 0.032	0.373 (14) ± 0.021	0.473 (13) ± 0.076	0.497 (11) ± 0.053	0.487 (15) ± 0.076
BJ-EN	0.608 (6) ± 0.065	0.665 (4) ± 0.132	0.723 (3) ± 0.11	0.622 (8) ± 0.092	0.650 (3) ± 0.059	0.629 (5) ± 0.073	0.709 (6) ± 0.039
Boost-CI	0.665 (3) ± 0.059	0.571 (11) ± 0.093	0.705 (5) ± 0.096	0.626 (7) ± 0.083	0.582 (5) ± 0.05	0.608 (7) ± 0.03	0.710 (5) ± 0.043
MTLSA	**0.701 (1)** ± 0.033	0.633 (8) ± 0.075	0.727 (2) ± 0.096	0.682 (3) ± 0.045	0.715 (2) ± 0.049	0.653 (2) ± 0.071	0.758 (2) ± 0.03
SurvGB	0.509 (12) ± 0.085	0.586 (10) ± 0.07	0.650 (10) ± 0.048	0.634 (5) ± 0.089	0.539 (11) ± 0.051	0.564 (10) ± 0.056	0.665 (10) ± 0.043
SurvSVM-Linear	0.481 (14) ± 0.055	**0.706 (1)** ± 0.075	0.719 (4) ± 0.059	0.647 (4) ± 0.052	0.552 (8) ± 0.044	0.599 (8) ± 0.039	0.709 (6) ± 0.027
SurvSVM-RBF	0.491 (13) ± 0.05	0.656 (6) ± 0.053	0.704 (6) ± 0.063	**0.725 (1)** ± 0.04	0.566 (7) ± 0.029	0.611 (6) ± 0.025	0.753 (3) ± 0.033
BJ-Neural	0.467 (15) ± 0.073	0.643 (7) ± 0.066	0.687 (7) ± 0.075	0.628 (6) ± 0.042	0.542 (10) ± 0.047	0.579 (9) ± 0.037	0.679 (9) ± 0.038
Proposed	0.681 (2) ± 0.128	0.685 (2) ± 0.035	**0.742 (1)** ± 0.038	0.712 (2) ± 0.032	**0.719 (1)** ± 0.042	**0.669 (1)** ± 0.068	**0.759 (1)** ± 0.081

c-index) [39] is used to evaluate the performance of time-to-event estimation models. The CI of a model, g, is defined as [40]:

$$\mathrm{CI}(g) = \frac{1}{num} \sum_{s_i=1} \sum_{y_j > y_i} \mathbb{I}(g(\boldsymbol{x}_j) > g(\boldsymbol{x}_i)),$$

where $\mathbb{I}(\cdot)$ is an indication function, and num is a normalization factor such that $\mathrm{CI}(g) \leqslant 1$, and $\mathrm{CI}(g) = 1$ is the best possible performance.

4.2.4. Comparisons

The results of *SurvSVMs* and *SurvGB* are obtained by using the functions in the *Scikit-Survival* package. The result of *BJ-Neural* is obtained by using the *Scikit-Learning* package. Results of the other baseline models are adopted from [14]. To avoid the sampling bias introduced in the random splits in the cross-validations, experiments are performed 5 times for each dataset-approach pair, and the means and the standard deviations are calculated over the repeats. Because such a procedure is not explicitly stated in [14], we assume that its reported results have eliminated the sampling bias, and thus the performance across all the baseline methods and datasets can be compared in a fair manner. The mean CIs and corresponding standard deviations of all baseline models and the proposed model on the benchmark datasets are shown in Table 3, with the best performing method highlighted in boldface.

Because the datasets in the experiments are different (as shown in Table 2), the performance in their corresponding tasks are not commensurable; therefore, a one-side Sign Test [41] is conducted to test the null hypothesis: *the proposed method and the baseline method perform the same*. Table 4 shows the pairwise comparison of the proposed method and each of the baseline methods. Column 2 shows the number of wins that the proposed method obtains in 7 benchmark datasets; column 3 shows the p-values of the Sign Test; column 4 and column 5 show whether the null hypothesis can be rejected if the significance level are set to 0.05 and 0.1, respectively. Although the Sign Test is suitable for comparing model performances across multiple tasks/datasets, it is also known to be weak [42]. Therefore, if the significance level is set to 0.05, the p-values show that the proposed method does not statistically outperform MTLSA, SurvSVM-Linear, and SurvSVM-RBF; if the significance level is set to 0.1, the proposed method statistically outperforms all the baseline methods.

Table 4
The pairwise comparison of the proposed method and each of the baseline methods. Column 2 shows the number of wins that the proposed method obtains in 7 benchmark datasets; column 3 shows the p-values of the Sign Test; column 4 and column 5 show whether the null hypothesis, *the proposed method and the baseline methods perform the same*, can be rejected if the significance level are set to 0.05 and 0.1, respectively.

	Proposed method Wins	Sign Test p-value	p-val < 0.05	p-val < 0.1
CoxPH	7	0.0078	YES	YES
CoxLasso	7	0.0078	YES	YES
CoxEN	7	0.0078	YES	YES
Logistic	7	0.0078	YES	YES
Weibull	7	0.0078	YES	YES
Log-Gaussian	7	0.0078	YES	YES
Log–Logistic	7	0.0078	YES	YES
Tobit	7	0.0078	YES	YES
BJ-EN	7	0.0078	YES	YES
Boost-CI	7	0.0078	YES	YES
MTLSA	6	0.0625	NO	YES
SurvGB	7	0.0078	YES	YES
SurvSVM-Linear	6	0.0625	NO	YES
SurvSVM-RBF	6	0.0625	NO	YES
BJ-Neural	7	0.0078	YES	YES

The comparison results suggest that representing each discretized event occurrence time as a concept with a vector representation and using the time vectors as the proxies to the event occurring time are

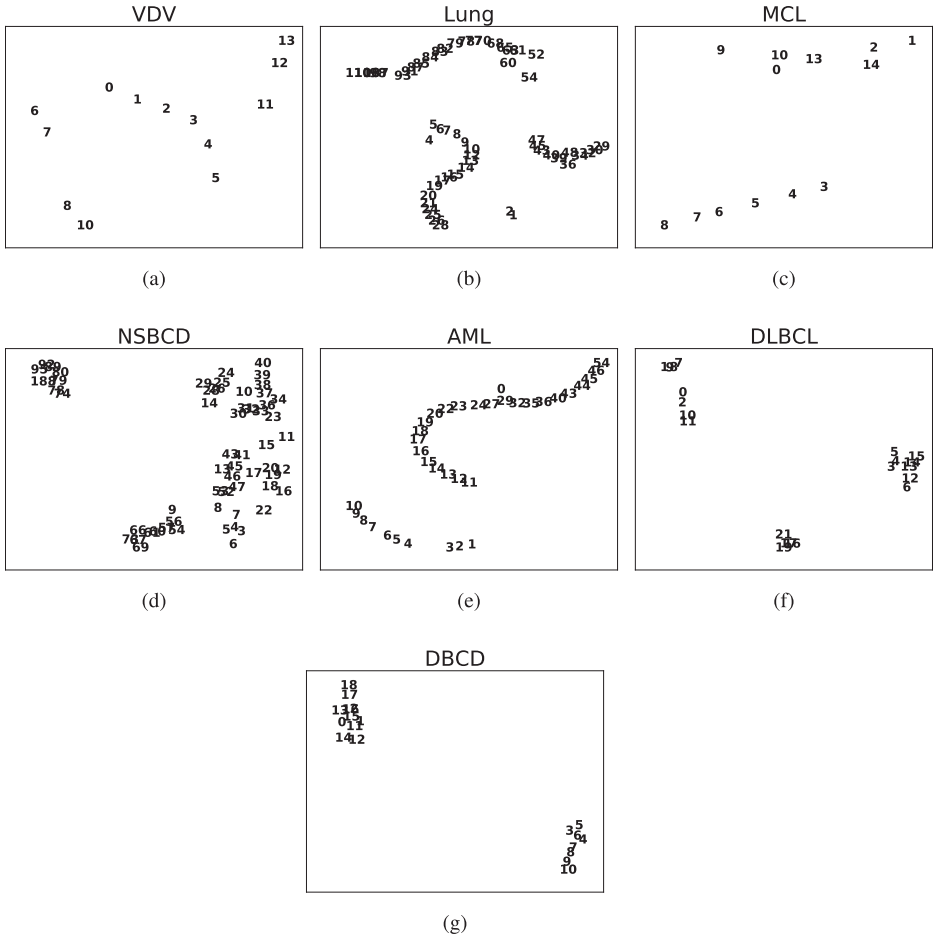

Fig. 2. 2D scatter plots of the learned time concept vectors in the benchmark datasets. The numbers in each plot indicate the corresponding time concepts, and their cluster formations may help in revealing time regimes.

helpful for the time-to-event tasks. By effectively exploiting the relations between features and time concepts, one can obtain a more accurate model for time-to-event estimation.

4.3. Experiment #2: regimes identification by visualizing the learned time concept vectors

The proposed learning framework does not only learn the model parameters (i.e. matrix $\mathbf{W}$) but also learns the time concept vectors (i.e. column vectors in matrix, $\mathbf{V}$). In this experiment, we visualize the learned time concept vectors and point out some interesting findings based on cluster formations. To generate time concept vectors, we apply the proposed model to each of the benchmark datasets using all available examples within a dataset. Using the best hyper-parameter settings found in *Experiment #1*. After each learning process is finished, the learned time concept vectors, which are high-dimensional, are projected on to a 2-dimensional space using t-SNE [43] for visualization. In each of the 2-dimensional scatter plots in Fig. 2, the 2D projected time concept vectors are indicated by their corresponding event occurrence times. The two axes in the 2D plot usually have no specific meanings, but the projections using t-SNE preserve the time concept vector distribution; in other words, time concept vectors that are in the same manifold in the high-dimensional space will be projected to the same cluster in the low-dimensional space.

Clusters can be easily identified in each scatter plot. For example, three major clusters can be identified in the *Lung* dataset (Fig. 2)b and two clusters can be identified in the *AML* dataset (Fig. 2e). The cluster formations reveal the time regime information. For example, in the *AML* dataset, one cluster consists of event occurrence time points 1–10, and another cluster consists of time points 11 and above; this may suggest that when a patient is estimated to have a survival time in-between 1 month and 10 months, they are not likely to achieve a complete remission, or are likely to suffer fatal complications of therapy. If a patient is estimated to have a survival time 11 months or above, the patient is more likely to achieve a durable complete remission. Another interesting finding is that early occurrence time concepts are able to form a cluster with the late occurrence time concepts. For example, in the plot of the *DBCD* time concepts, the early time concepts (0, 1, 2) are in the same cluster as late occurrence time concepts (11 and above), and time concepts 3–10 form a second cluster. Clinical experience bears this out: some patients will present critically ill, with most known prognostic factors not in their favor, yet they survive and achieve a durable complete remission.

4.4. Experiment #3: model scalability evaluation on synthetic datasets

We empirically evaluate the scalability of the proposed model with respect to the number of examples (n), the number of features (m), the number of possible event times (T), and the dimensionality of the time concept vector space (d). In this experiment, we synthetically generate n 3-tuples $\{\boldsymbol{x}_i, y_i, s_i\}_{i=1}^{n}$. Each elements in $\boldsymbol{x}_i \in \mathbb{R}^m$ is generated by using a uniform distribution in the open interval of (-1, 1); the event occurrence time, $y_i \in \{1, 2, \cdots, T\}$, are generated by using a discrete uniform distribution between one and the number of possible event occurrence times, namely, $P(y_i = 1) = P(y_i = 2) \cdots = P(y_i = T) = \frac{1}{T}$; and the event status, $s_i \in \{0, 1\}$ are generated by a Bernoulli distribution with mean 0.5, namely $P(s_i = 0) = P(s_i = 1) = 0.5$. Multiple runs are conducted for

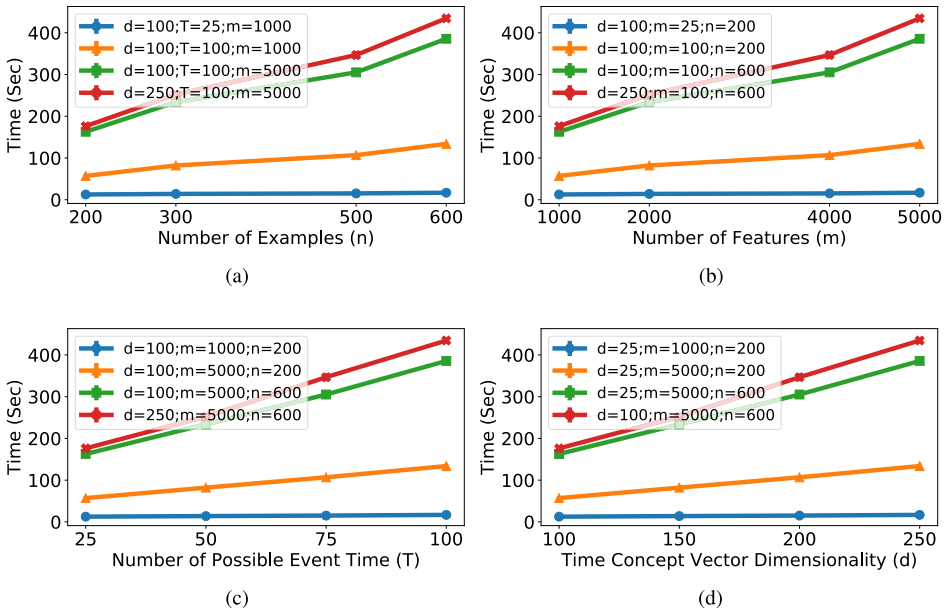

Fig. 3. Learning time of the proposed model with different n, m, T, and d settings. (a) learning time changes as n changes while other variables are constant; (b) learning time changes as m changes while other variables are constant; (c) learning time changes as T changes while other variables are constant; (d) learning time changes as d changes while other variables are constant.

each (n, m, T, d) setting; however, because the learning time variations of the runs within the same setting are very small, the error bars (standard deviations) of each point in the plots in Fig. 3 are not obvious. Each curve in Fig. 3a shows the learning time of the model as the number of examples (n) takes {100, 200, 500, 600} while m, T, and d remain constant. Each curve in Fig. 3b shows the learning time of the model as the number of features (m) takes {1000, 2000, 4000, 5000} while n, d, and T remain constant. Each curve in Fig. 3c shows the learning time of the model as the number of possible event times (T) takes {25, 50, 75, 100} while n, m, and d remain constant. Each curve in Fig. 3d shows the learning time of the model as the time concept dimensionality (d) takes {100, 150, 200, 250} while n, m, and T remain constant. These figures demonstrate that the learning time of the proposed model is approximately linear with respect to n, m, T, and d. Therefore, the time complexity of the proposed algorithm is approximately $O(nmTd)$. That means, in each sub-figure, the slope of the top (red) curve is approximately 50 times as the slope of the bottom (blue) curve, and thus the bottom (blue) curve looks very "flat".

5. Conclusion

In this study, we propose a new model for the time-to-event estimation problem. In the proposed model, instead of using the actual time points, time concept vectors are used as the target. A scalable optimization framework is also developed to learn the model parameters and time concept vectors jointly. Empirical results show that the proposed model is effective and efficient. It yields results consistent with clinical observation. Using this methodology may reveal previously unrecognized associations between specific clinical characteristics and survival, generating hypotheses to drive further prospective investigation.

Declaration of Competing Interest

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

References

[1] J.F. Tierney, L.A. Stewart, D. Ghersi, S. Burdett, M.R. Sydes, Practical methods for incorporating summary time-to-event data into meta-analysis, Trials 8 (1) (2007) 16.
[2] J.P. Klein, M.L. Moeschberger, Survival Analysis: Techniques for Censored and Truncated Data, Springer Science & Business Media, 2006.
[3] P. Wang, Y. Li, C.K. Reddy, Machine learning for survival analysis: a survey, ACM Comput. Surv. (CSUR) 51 (6) (2019) 110.
[4] L. Gordon, R.A. Olshen, Tree-structured survival analysis, Cancer Treatment Rep. 69 (10) (1985) 1065–1069.
[5] M. LeBlanc, J. Crowley, Relative risk trees for censored survival data, Biometrics (1992) 411–425.
[6] I. Bou-Hamad, D. Larocque, H. Ben-Ameur, et al., A review of survival trees, Stat. Surv. 5 (2011) 44–71.
[7] F.M. Khan, V.B. Zubek, Support vector regression for censored data (svrc): a novel tool for survival analysis, Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on, IEEE, 2008, pp. 863–868.
[8] V. Van Belle, K. Pelckmans, J. Suykens, S. Van, Huffel, Support vector machines for survival analysis, Proceedings of the Third International Conference on Computational Intelligence in Medicine and Healthcare (CIMED2007), 2007, pp. 1–8.
[9] V. Van Belle, K. Pelckmans, S. Van Huffel, J.A. Suykens, Support vector methods for survival analysis: a comparison between ranking and regression approaches, Artif. Intell. Med. 53 (2) (2011) 107–118.
[10] H. Ishwaran, U.B. Kogalur, X. Chen, A.J. Minn, Random survival forests for high-dimensional data, Stat. Anal. Data Min.: ASA Data Sci. J. 4 (1) (2011) 115–132.
[11] T. Hothorn, P. Bühlmann, S. Dudoit, A. Molinaro, M.J. Van Der Laan, Survival ensembles, Biostatistics 7 (3) (2005) 355–373.
[12] B. Vinzamuri, Y. Li, C.K. Reddy, Active learning based survival regression for censored data, Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, ACM, 2014, pp. 241–250.
[13] Y. Li, L. Wang, J. Wang, J. Ye, C.K. Reddy, Transfer learning for survival analysis via efficient l2, 1-norm regularized cox regression, in: Data Mining (ICDM), in: 2016 IEEE 16th International Conference on, IEEE, 2016, pp. 231–240.
[14] Y. Li, J. Wang, J. Ye, C.K. Reddy, A multi-task learning formulation for survival analysis, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2016, pp. 1715–1724.
[15] L. Wang, Y. Li, J. Zhou, D. Zhu, J. Ye, Multi-task survival analysis, 2017 IEEE International Conference on Data Mining (ICDM), IEEE, 2017, pp. 485–494.
[16] R. Ranganath, A. Perotte, N. Elhadad, D. Blei, Deep survival analysis, arXiv preprint arXiv:1608.02158.
[17] P. Chapfuwa, C. Tao, C. Li, C. Page, B. Goldstein, L. Carin, R. Henao, Adversarial time-to-event modeling, arXiv preprint arXiv:1804.03184.
[18] D.R. Cox, Regression models and life-tables, Breakthroughs in Statistics, Springer, 1992, pp. 527–541.
[19] J. Fan, J. Jiang, Non-and semi-parametric modeling in survival analysis, New Developments in Biostatistics and Bioinformatics, World Scientific, 2009, pp. 3–33.
[20] R. Tibshirani, The lasso method for variable selection in the cox model, Stat. Med. 16 (4) (1997) 385–395.
[21] N. Simon, J. Friedman, T. Hastie, R. Tibshirani, Regularization paths for cox's

proportional hazards model via coordinate descent, J. Stat. Software 39 (5) (2011) 1.
[22] E.T. Lee, J. Wang, Statistical Methods for Survival Data Analysis Vol. 476 John Wiley & Sons, 2003.
[23] J. Tobin, Estimation of relationships for limited dependent variables, Econ.: J. Econ. Soc. (1958) 24–36.
[24] J. Buckley, I. James, Linear regression with censored data, Biometrika 66 (3) (1979) 429–436.
[25] E.L. Kaplan, P. Meier, Nonparametric estimation from incomplete observations, J. Am. Stat. Assoc. 53 (282) (1958) 457–481.
[26] S. Wang, B. Nan, J. Zhu, D.G. Beer, Doubly penalized Buckley–James method for survival data with high-dimensional covariates, Biometrics 64 (1) (2008) 132–140.
[27] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT press, 2016.
[28] N. Parikh, S. Boyd, Proximal algorithms, Found. Trends Optim. 1 (3) (2013) 123–231.
[29] L.J. Van't Veer, H. Dai, M.J. Van De Vijver, Y.D. He, A.A. Hart, M. Mao, H.L. Peterse, K. Van Der Kooy, M.J. Marton, A.T. Witteveen, et al., Gene expression profiling predicts clinical outcome of breast cancer, Nature 415 (6871) (2002) 530.
[30] D.G. Beer, S.L. Kardia, C.-C. Huang, T.J. Giordano, A.M. Levin, D.E. Misek, L. Lin, G. Chen, T.G. Gharib, D.G. Thomas, et al., Gene-expression profiles predict survival of patients with lung adenocarcinoma, Nat. Med. 8 (8) (2002) 816.
[31] A. Rosenwald, G. Wright, A. Wiestner, W.C. Chan, J.M. Connors, E. Campo, R.D. Gascoyne, T.M. Grogan, H.K. Muller-Hermelink, E.B. Smeland, et al., The proliferation gene expression signature is a quantitative integrator of oncogenic events that predicts survival in mantle cell lymphoma, Cancer Cell 3 (2) (2003) 185–197.
[32] T. Sørlie, R. Tibshirani, J. Parker, T. Hastie, J.S. Marron, A. Nobel, S. Deng, H. Johnsen, R. Pesich, S. Geisler, et al., Repeated observation of breast tumor subtypes in independent gene expression data sets, Proc. Natl. Acad. Sci. 100 (14) (2003) 8418–8423.
[33] L. Bullinger, K. Döhner, E. Bair, S. Fröhling, R.F. Schlenk, R. Tibshirani, H. Döhner, J.R. Pollack, Use of gene-expression profiling to identify prognostic subclasses in adult acute myeloid leukemia, N. Engl. J. Med. 350 (16) (2004) 1605–1616.
[34] I.S. Lossos, Diffuse large b cell lymphoma: from gene expression profiling to prediction of outcome, Biol. Blood Marrow Transplant. 14 (1) (2008) 108–111.
[35] H.C. van Houwelingen, T. Bruinsma, A.A. Hart, L.J. van't Veer, L.F. Wessels, Cross-validated cox regression on microarray gene expression data, Stat. Med. 25 (18) (2006) 3201–3216.
[36] A. Mayr, M. Schmid, Boosting the concordance index for survival data–a unified framework to derive and evaluate biomarker combinations, PloS One 9 (1) (2014) e84483.
[37] S. Pölsterl, N. Navab, A. Katouzian, Fast training of support vector machines for survival analysis, Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, 2015, pp. 243–259.
[38] S. Pölsterl, N. Navab, A. Katouzian, An efficient training algorithm for kernel survival support vector machines, arXiv preprint arXiv:1611.07054.
[39] P.J. Heagerty, Y. Zheng, Survival model predictive accuracy and roc curves, Biometrics 61 (1) (2005) 92–105.
[40] H. Steck, B. Krishnapuram, C. Dehing-oberije, P. Lambin, V.C. Raykar, On ranking in survival analysis: Bounds on the concordance index, in: Advances in Neural Information Processing Systems, 2008, pp. 1209–1216.
[41] D.J. Sheskin, Handbook of Parametric and Nonparametric Statistical Procedures, Chapman and Hall/CRC, 2003.
[42] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (Jan) (2006) 1–30.
[43] L.v.d. Maaten, G. Hinton, Visualizing data using t-sne, J. Mach. Learn. Res. 9 (Nov) (2008) 2579–2605.