ASSUMPTION-FREE SURVIVAL ANALYSIS UNDER LOCAL SMOOTHNESS PRIOR

Anonymous authors
Paper under double-blind review

ABSTRACT

Survival analysis appears in various fields such as medicine, economics, engineering, and business. Due to the difficulty of integration that naturally arises in continuous-time modeling, previous works either made a strong assumption or discretized the time domain, thus limits their practical usages. In this paper, we propose assumption-free survival analysis, which models continuous-time survival function without any assumption. Our model obtains an *assumption-free* survival function by integrating an *assumption-free* hazard function using Neural Ordinary Differential Equations. Inspired by smoothness prior from semi-supervised learning literature, we further propose a regularizer that encourages the survival function to be locally smooth by minimizing the variation of the survival function in the covariate space. We found this regularizer increases the predictive power of the survival function as it propagates high-quality local information to the neighborhoods of data points. Experimental results on three public benchmarks show that our approach has better predictive power and is well-calibrated compared to strong baselines. Moreover, the proposed regularizer is superior to global regularizers and insensitive to hyperparameters.

1 INTRODUCTION

Survival analysis (a.k.a time-to-event modeling) is a branch of statistics that predicts the duration of time until an event occurs (Kleinbaum & Klein, 2012). Survival analysis appears in various fields such as time-to-death modeling in medicine (Cox, 1972; Ishwaran et al., 2008; Katzman et al., 2018), unemployment duration prediction in economics (Meyer, 1988), machine failure analysis in engineering (O'Connor & Kleyner, 2011), and churn prediction in business (Jing & Smola, 2017; Li et al., 2021). As time is essentially continuous, survival analysis models should be able to model continuous time. However, continuous-time modeling is challenging due to the integration operation that naturally arises in survival analysis. Previous works proposed methods that detour the difficulty.

The Cox proportional hazards model (Cox, 1972; Katzman et al., 2018), which is a widely used model in survival analysis, makes a strong assumption that if the survival probability of a person A is higher than that of another person B at a certain time, A's survival probability is always higher than that of B. A mixture of experts is employed to model the density function (Nagpal et al., 2021a). Each primitive distribution should have support only in the space of positive reals as time in survival analysis is inherently positive. Also, each primitive distribution is assumed to be integrated into closed-form cumulative density function for ease of survival probability calculation. Another class of models discretized the time domain to detour the difficulty of continuous-time modeling (Lee et al., 2018; Ren et al., 2019; Xue et al., 2020).

In this paper, we model the continuous-time survival function using Neural Ordinary Differential Equations (Neural ODEs) (Chen et al., 2018; Kidger et al., 2021). The relationship between the log survival function and the hazard function (a.k.a conditional failure rate) is naturally defined as an ODE. By using Neural ODEs, we can solve the ODE numerically and make the continuous-time survival function fully learnable. Unlike previous works, the proposed method's hazard function and survival function are free from any assumption.

An *assumption-free* survival function is prone to be a wiggly function. To compensate for this, we propose a regularizer that enhances local smoothness. The regularizer minimizes the variation of the survival function in the covariate space so that the survival function becomes locally smooth. Under the smoothness prior, which is a classical inductive bias in semi-supervised learning (Chapelle et al., 2006), the proposed regularizer increases the predictive power of the survival function as it propagates high-quality local information to the neighborhoods of data points.

To demonstrate the superiority of our approach, we conduct experiments on three public benchmarks: SUPPORT, METABRIC, and GBSG. The experimental results show that our approach outperforms state-of-the-art baselines. The main contributions of this paper are summarized as follows:

- Assumption-free continuous-time survival functions based on Neural ODEs.
- A regularizer that encourages the survival function to be locally smooth and increases the predictive power of the survival function.
- The possibility of applying the developments in semi-supervised learning to survival analysis by experimentally showing the smoothness prior holds in survival analysis.
- The state-of-the-art predictive power, calibration performance on three public benchmarks.

2 Preliminaries

2.1 Survival Analysis

Survival analysis data comprises an observed covariate $\boldsymbol{x}$, a failure event time t, and an event indicator e. If an event is observed, t corresponds to the duration time from the beginning of the follow-up of an individual until the event occurs. In this case, event indicator $e = 1$. If an event is unobserved, t corresponds to the duration time from the beginning of follow-up of an individual until the last follow-up. In this case, we cannot know the exact time of the event occur and event indicator $e = 0$. An individual is said to be *right-censored* if $e = 0$. The presence of *right-censored* data differentiates survival analysis from regression problems. In this paper, we only focus on the single-risk problem where event e is a binary-valued variable.

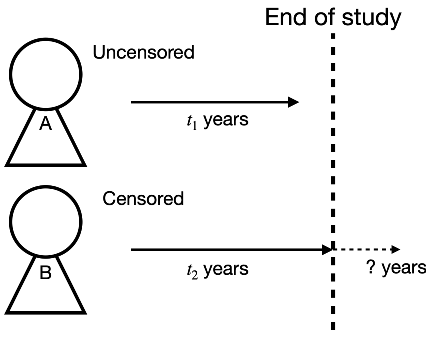


Figure 1: The difference between *uncensored* data and *censored* data. Individual B did not fail until the time of end of study t_2 while A failed at t_1 which is before the end time of study. All we can know about B is that B did not fail until t_2.

Given a set of triplet $\mathcal{D} = \{(\boldsymbol{x}_i, t_i, e_i)\}_{i=1}^N$, the goal of survival analysis is to predict the likelihood of an event occur $p(t \mid \boldsymbol{x})$ or the survival probability $S(t \mid \boldsymbol{x})$. The likelihood and the survival probability have the following relationship:

$$S(t \mid \boldsymbol{x}) = 1 - \int_0^t p(\tau \mid \boldsymbol{x}) d\tau \tag{1}$$

Modeling $p(t \mid \boldsymbol{x})$ or $S(t \mid \boldsymbol{x})$ directly is challenging as those have the following constraints:

$$\begin{gathered} p(t \mid \boldsymbol{x}) > 0, \quad \int_0^\infty p(\tau \mid \boldsymbol{x}) d\tau = 1 \\ S(0) = 1, \lim_{t \to \infty} S(t) = 0, \quad S(t) \text{ non-increasing} \end{gathered}$$

Many previous works instead modeled the hazard function (a.k.a conditional failure rate) $h(t \mid \boldsymbol{x})$.

$$h(t \mid \boldsymbol{x}) := \lim_{\Delta t \to 0} \frac{P(t \leq T < t + \Delta t \mid T \geq t, \boldsymbol{x})}{\Delta t} = \frac{p(t \mid \boldsymbol{x})}{S(t \mid \boldsymbol{x})} \tag{2}$$

As the hazard function is a probability per unit time, it is unbounded upwards. Hence, the only constraint of the hazard function is that the function is non-negative: $h(t \mid \boldsymbol{x}) \geq 0$

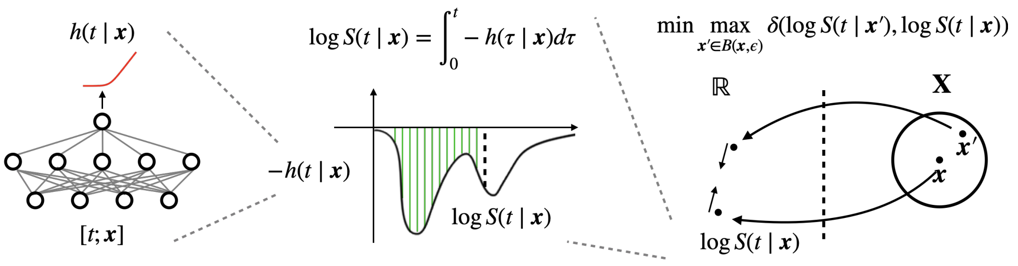

$h(t \mid x)$
$[t; x]$
$\log S(t \mid x) = \int_0^t -h(\tau \mid x) d\tau$
$-h(t \mid x)$
$\log S(t \mid x)$
$\min \max_{x' \in B(x,\epsilon)} \delta(\log S(t \mid x'), \log S(t \mid x))$
$\mathbb{R}$
$\mathbf{X}$
x'
x
$\log S(t \mid x)$

Figure 2: The overview of Assumption-Free Survival Analysis (AFreeSurv) and Local Survival Smoothing (LS2) regularizer. **Left)** We model the hazard function with a neural network followed by the softplus function to ensure that the output of the hazard function is positive. By modeling the hazard function with a neural network, we do not employ any assumption. **Middle)** We define log survival function as the solution of the ODE initial value problem where the ODE is defined as equation 3 and the initial value $\log S(0 \mid \boldsymbol{x}) = 0$. We calculate the survival function by integrating *assumption-free* hazard function. This, in turn, returns the *assumption-free* survival function. **Right)** The LS2 regularizer encourages the survival function to be locally smooth. At first LS2 finds the neighbor covariate $\boldsymbol{x}'$ of a covariate $\boldsymbol{x}$ whose log survival probability at time t is most different from that of $\boldsymbol{x}$. After then it minimize the difference between two log survival probability: $\log S(t \mid \boldsymbol{x}')$ and $\log S(t \mid \boldsymbol{x})$

2.2 Neural ODEs

Neural ODEs are a family of neural network models that define the continuous dynamics of variables (Chen et al., 2018). Starting from $\boldsymbol{z}(0)$, we can define the output $\boldsymbol{z}(T)$ to be the solution of the following ordinary differential equation (ODE) initial value problem.

$$\frac{d\boldsymbol{z}(t)}{dt} = f(\boldsymbol{z}(t), t, \theta)$$
$$\boldsymbol{z}(T) = \boldsymbol{z}(0) + \int_0^T f(\boldsymbol{z}(t), t, \theta) dt$$

Naively applying an ODE solver to solve an ODE initial value problem leads to a practical difficulty. An ODE solver builds a big computation graph which gives rise to high memory cost and additional numerical error in backpropagation steps.

Chen et al. (2018) showed that we can calculate the gradients of a scalar-valued loss w.r.t all inputs of any ODE solver with constant memory cost. We can calculate the gradients without backpropagating through the operations of the solver but with another call to an ODE solver.

3 Methods

In this section, we describe **A**ssumption-**Free Surv**ival Analysis (AFreeSurv) and **L**ocal **S**urvival **S**moothing (LS2) regularizer. Figure 2 illustrates an overview of AFreeSurv and LS2.

3.1 Assumption-Free Survival Analysis

We can obtain an ODE which explains the relationship between the hazard function and the survival function by putting derivative of equation 1 into equation 2 (Kleinbaum & Klein, 2012).

$$h(t \mid \boldsymbol{x}) = \frac{p(t \mid \boldsymbol{x})}{S(t \mid \boldsymbol{x})} = \frac{1}{S(t \mid \boldsymbol{x})} \left(-\frac{dS(t \mid \boldsymbol{x})}{dt} \right) = -\frac{d \log S(t \mid \boldsymbol{x})}{dt} \tag{3}$$

Starting from initial value $\log S(0 \mid \boldsymbol{x}) = 0$, we can define $\log S(t \mid \boldsymbol{x})$ as the solution of the ODE initial value problem where the ODE is defined as equation 3. We can acquire an *assumption-free* hazard function and an *assumption-free* survival function with constant memory cost by modeling $h(t \mid \boldsymbol{x})$ using a neural network followed by the softplus activation function and modeling $\log S(t \mid \boldsymbol{x})$ using Neural ODEs.

$$\log S(t \mid \boldsymbol{x}) = \log S(0 \mid \boldsymbol{x}) + \int_0^t -h(\tau \mid \boldsymbol{x}) d\tau = \int_0^t -h(\tau \mid \boldsymbol{x}) d\tau$$

From equation 2, the density function can be expressed as a product of the hazard function and the survival function.

$$p(t \mid \boldsymbol{x}) = h(t \mid \boldsymbol{x}) \cdot S(t \mid \boldsymbol{x})$$

We train the hazard function by minimizing the negative event-time log-likelihood

$$\begin{aligned}\mathcal{L} &= -\mathbb{E}_{(\boldsymbol{x},t,e)\sim\mathcal{D}}[\log p(t \mid \boldsymbol{x})^e \cdot S(t \mid \boldsymbol{x})^{1-e}] \\ &= -\mathbb{E}_{(\boldsymbol{x},t,e)\sim\mathcal{D}}[e \log p(t \mid \boldsymbol{x}) + (1-e)\log S(t \mid \boldsymbol{x})] \qquad (4) \\ &= -\mathbb{E}_{(\boldsymbol{x},t,e)\sim\mathcal{D}}[e \log h(t \mid \boldsymbol{x}) \cdot S(t \mid \boldsymbol{x}) + (1-e) S(t \mid \boldsymbol{x})] \\ &= -\mathbb{E}_{(\boldsymbol{x},t,e)\sim\mathcal{D}}\left[e\left(\log h(t \mid \boldsymbol{x}) + \int_0^t -h(\tau \mid \boldsymbol{x})d\tau\right) + (1-e)\int_0^t -h(\tau \mid \boldsymbol{x})\right] \qquad (5)\end{aligned}$$

Applying Neural ODEs to the ODE defined in equation 3 is simple yet effective. However, to our surprise, our work is the first to apply Neural ODEs in this way.

3.2 Locally Smoothing the Survival Function

As the survival function we propose is free from any assumption, it is prone to become a wiggly function. However, it is highly unlikely that a function in the real world has extreme oscillations (Murphy, 2012). *Smoothness prior*, which is a common inductive bias in semi-supervised learning literature, state that if the inputs in a high-density region are close, then so should be the corresponding outputs (Chapelle et al., 2006). Inspired by the smoothness prior, we propose a regularizer that minimizes the variation of the survival in the covariate space. Formally speaking, $S(t \mid \boldsymbol{x}) \approx S(t \mid \boldsymbol{x} + \epsilon \boldsymbol{u})$ where ϵ is a small positive number and $\boldsymbol{u}$ is a unit vector.

To minimize the variation of the survival in the covariate space, we minimize the following regularizer.

$$\mathcal{R} = \mathbb{E}_{(\boldsymbol{x},t)\sim\mathcal{D}}\left[\max_{\boldsymbol{x}' \in B(\boldsymbol{x},\epsilon)} \delta(\log S(t \mid \boldsymbol{x}'), \log S(t \mid \boldsymbol{x}))\right] \qquad (6)$$

where $B(\boldsymbol{x}, \epsilon)$ is a epsilon ball with center $\boldsymbol{x}$ and δ is a distance function. Intuitively speaking, we find $\boldsymbol{x}^*$ whose log survival probability at time t is the most different from that of $\boldsymbol{x}$ near $\boldsymbol{x}$. We then minimize the difference between the two log survival probability: $\log S(t \mid \boldsymbol{x}')$ and $\log S(t \mid \boldsymbol{x})$. Obviously, the key to calculating the value of the regularizer is to find

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}' \in B(\boldsymbol{x},\epsilon)}{\arg\max}\,[\delta(\log S(t \mid \boldsymbol{x}'), \log S(t \mid \boldsymbol{x}))]$$

Instead of finding the exact $\boldsymbol{x}^*$, we find the approximate $\hat{\boldsymbol{x}} \approx \boldsymbol{x}^*$. The first order Taylor approximation of $\log S(t \mid \boldsymbol{x}')$ is as follows:

$$\log S(t \mid \boldsymbol{x}') \approx \log S(t \mid \boldsymbol{x}) + (\boldsymbol{x}' - \boldsymbol{x})^T \frac{d \log S(t \mid \boldsymbol{x})}{d\boldsymbol{x}} \qquad (7)$$

Say, we set $\delta(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|_2$. Putting equation 7 into the inside of the brackets of equation 6 yields the following equation.

$$\max_{\boldsymbol{x}' \in B(\boldsymbol{x},\epsilon)} \left\| (\boldsymbol{x}' - \boldsymbol{x})^T \frac{d \log S(t \mid \boldsymbol{x})}{d\boldsymbol{x}} \right\|_2$$

The maximum of $\left\| (\boldsymbol{x}' - \boldsymbol{x})^T \frac{d \log S(t|\boldsymbol{x})}{d\boldsymbol{x}} \right\|_2$ under the constraint $\boldsymbol{x}' \in B(\boldsymbol{x},\epsilon)$ is obtained when $\boldsymbol{x}' - \boldsymbol{x} = \epsilon \cdot normalize\left(\frac{d \log S(t|\boldsymbol{x})}{d\boldsymbol{x}}\right)$ where $normalize(\boldsymbol{z}) = \frac{\boldsymbol{z}}{\|\boldsymbol{z}\|_2}$. Hence, the approximate $\hat{\boldsymbol{x}}$ is expressed as

$$\hat{\boldsymbol{x}} = \boldsymbol{x} + \epsilon \cdot normalize\left(\frac{d \log S(t \mid \boldsymbol{x})}{d\boldsymbol{x}}\right) \tag{8}$$

Now, we can train an *assumption-free* survival function with the local survival smoothing regularizer by combining equation 4 and equation 6.

$$\begin{aligned} \mathcal{L}_{all} = \mathcal{L} + \lambda\mathcal{R} &\approx \mathcal{L} + \lambda\mathbb{E}_{(\boldsymbol{x},t)\sim\mathcal{D}}\left[\|\log S(t \mid \hat{\boldsymbol{x}}) - \log S(t \mid \boldsymbol{x})\|_2\right] \\ &= -\mathbb{E}_{(\boldsymbol{x},t,e)\sim\mathcal{D}}[e \log p(t \mid \boldsymbol{x}) + (1-e)\log S(t \mid \boldsymbol{x})] \\ &\quad + \lambda\mathbb{E}_{(\boldsymbol{x},t)\sim\mathcal{D}}[\|\log S(t \mid \hat{\boldsymbol{x}}) - \log S(t \mid \boldsymbol{x})\|_2] \end{aligned} \tag{9}$$

where λ is a coefficient that controls the magnitude of the local survival smoothing regurlarizer.

4 Related Work

Previous works proposed methods to model censored data in survival analysis. Cox (1972) proposed Cox proportional hazards (CoxPH) which assumes constant proportional hazard. Every individual has an extended or shrunk form of hazard function in the same shape. This is a strong assumption in that if the survival probability of a person A is higher than that of another person B at a certain time, A's survival probability is higher than that of B all the time. Faraggi & Simon (1995); Katzman et al. (2018) replaced the linear model of CoxPH with a neural network.

A mixture of experts was used for better expressivity of the density function (Nagpal et al., 2021a;b). Though a mixture of experts is an effective method, the method assumes that the density function is a combination of primitive distributions. For ease of integration, the density function is further restricted to a mixture of primitive distribution with a closed-form cumulative density function (Nagpal et al., 2021a).

A line of works employed Gaussian Process (GP) for survival analysis (Fernandez et al., 2016; Alaa & van der Schaar, 2017). Employing GP can make a more expressive hazard function compared to CoxPH. However, the methods need expert knowledge for selecting Gaussian process prior and kernel functions. Chapfuwa et al. (2018) modeled the density function using Generative Adversarial Networks (GANs). Though the adversarial approach does not have an assumption, it cannot access exact likelihood as GANs are implicit models.

The work of Groha et al. (2020) is similar to ours in that Neural ODEs are applied to survival analysis. However, we apply Neural ODEs to the relationship between the hazard function and the survival function while Groha et al. (2020) applied Neural ODEs to Markov jump processes for multi-state modeling. In addition, we explore the applicability of semi-supervision through the LS2 regularizer while Groha et al. (2020) did not.

Minimax optimization can be found in diverse fields of Machine Learning including GANs (Goodfellow et al., 2014), adversarial training (Madry et al., 2018), semi-supervised learning (Miyato et al., 2018b), and optimization procedure (Foret et al., 2021). Among those works, the most related to ours is virtual adversarial training (Miyato et al., 2018b). Ours and Miyato et al. (2018b)'s work are similar in that both find the neighbor with the most variation in the output space and minimize the variation. However, ours focus on survival analysis while Miyato et al. (2018b)'s work focused on the classification problem. To the best of our knowledge, our work is the first to explore how helpful a smoothness prior is in survival analysis. Through the exploration, we open the new chances of applicability of advances in semi-supervised learning to survival analysis.

Table 1: Summary statistics of the datasets used in our experiments.

Dataset	N	d	Censoring (%)	Durations		Event Quantiles		
				# unique	domain	$t = 25\%$	$t = 50\%$	$t = 75\%$
SUPPORT	9105	24	31.89%	1724	$\mathbb{N}^+$	14	58	252
METABRIC	1904	9	42.06%	1686	$\mathbb{R}^+$	42.68	85.86	145.33
GBSG	2232	7	43.23%	1230	$\mathbb{R}^+$	13.61	24.01	40.32

5 Experiments

In this section, we evaluate our model against competitive baselines, on three publicly available real-world datasets: Study to Understand Prognoses Preferences Outcomes and Risks of Treatment (**SUPPORT**), Molecular Taxonomy of Breast Cancer International Consortium (**METABRIC**), and Rotterdam & German Breast Cancer Study Group (**GBSG**). Table 1 summarizes the details of the datasets.

5.1 Methods Compared

Cox Proportional Hazards (CoxPH) is the standard semi-parametric model which assumes constant proportional hazard, i.e., $h(t \mid \boldsymbol{x}) = h_0(t)e^{f(\boldsymbol{x})}$ (Cox, 1972). The function $f(\boldsymbol{x})$ is linear. The model is semi-parametric in that the baseline hazard $h_0(t)$ is an unspecified function.

DeepSurv is an extension of CoxPH where the linear function in the exponent of e is replaced with a deep neural network (Katzman et al., 2018).

DeepHit models the distribution of first hitting time $p(t \mid \boldsymbol{x})$. To satisfy the positivity constraint $p(t \mid \boldsymbol{x}) > 0$ and the integrate-to-one constraint $\int_0^\infty p(t \mid \boldsymbol{x})dt = 1$, Lee et al. (2018) discretize the time domain and employ a neural network followed by a single softmax layer.

Deep Recurrent Survival Analysis (DRSA) employs RNNs to model survival probability over the discretized time domain. The hazard function at time k is defined as $h(t = k \mid \boldsymbol{x}) = f_\theta(\boldsymbol{r}_k)$ where $\boldsymbol{r}_k$ is the kth hidden vector of the RNN and $f_\theta(\cdot)$ is a neural network.

Random Survival Forests (RSF) is an extension of the Random Forest model to right-censored data (Ishwaran et al., 2008). RSF splits the tree so that each node maximizes survival difference.

Deep Survival Machines (DSM) is a fully parametric model which models the density function $p(t \mid \boldsymbol{x})$ as a mixture of K primitive distributions. Nagpal et al. (2021a) choose the Weibull and the Log-Normal distribution for the primitive distributions as a) they have support only in the positive reals and b) closed-from solution CDF.

5.2 Evaluation Metrics

Throughout this subsection, we denote $\hat{S}(t \mid \boldsymbol{x})$ as the estimate of $S(t \mid \boldsymbol{x})$, $I(\cdot)$ as the indicator function, $(\boldsymbol{x}_i, T_i, e_i)$ as the ith covariate, time, event indicator of the dataset, $\hat{G}(t)$ as the Kaplan-Meier estimator for censoring distribution (Kaplan & Meier, 1958), and ω_i as $\frac{1}{\hat{G}(T_i)}$.

Time Dependent Concordance Index (C^{td}) The concordance index, or C-Index is defined as the proportion of correctly ordered pairs among all comparable pairs. We use time dependent variant of C-Index that truncates pairs within the prespecified time point (Uno et al., 2011).

$$C^{td}(t) = \frac{\sum_{i=1}^N \sum_{j=1}^N e_i\{\hat{G}(T_i)\}^{-2} I(T_i < T_j, T_i < t) I(\hat{S}(t \mid \boldsymbol{x}_i) < \hat{S}(t \mid \boldsymbol{x}_j))}{\sum_{i=1}^N \sum_{j=1}^N e_i\{\hat{G}(T_i)\}^{-2} I(T_i < T_j, T_i < t)}$$

Time Dependent Area Under Curve (AUC) is an extension of the ROC-AUC to survival data (Hung & Chiang, 2010). It measures how well a model can distinguish individuals who fail before the given time ($T_i < t$) and who fail after the given time ($T_j > t$).

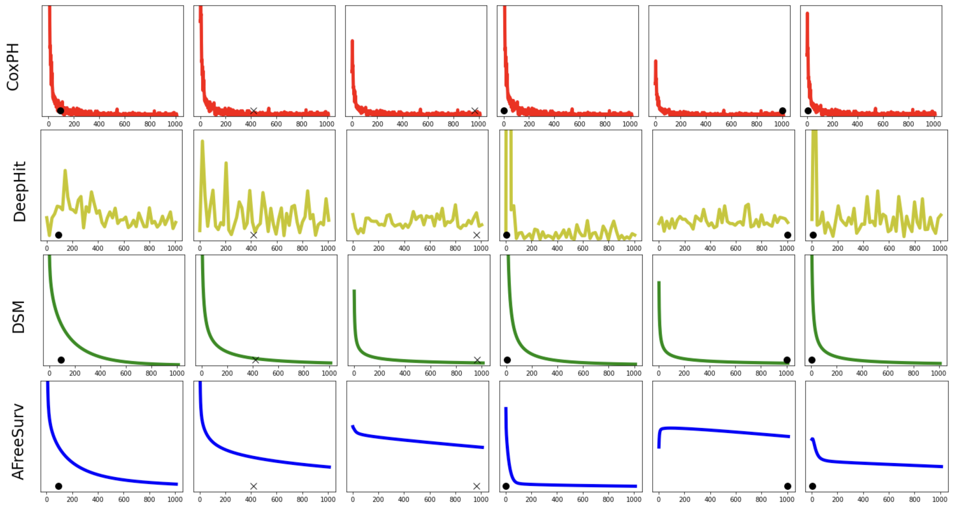


Figure 3: Density plots of various models on SUPPORT dataset. Uncensored data (●) and censored data (✕) are drawn at the corresponding time. **CoxPH** outputs density plots of almost similar shapes across all samples. **DeepHit** outputs density plots of relatively various shapes compared to CoxPH or DSM yet the model outputs wiggly density plots as it models discrete-time domain using softmax. **DSM** sometimes outputs density plots with a gentle decrease. However, we can observe a sharp decrease across all density plots. **AFreeSurv** outputs density plots of various shapes across various samples. We can even observe a slightly increasing density plot in the initial time (5th sample).

$$AUC(t) = \frac{\sum_{i=1}^{N}\sum_{j=1}^{N} I(T_j > t) I(T_i \leq t)\omega_i I(\hat{S}(t \mid \boldsymbol{x}_i) \leq \hat{S}(t \mid \boldsymbol{x}_j))}{(\sum_{i=1}^{N} I(T_i > t))(\sum_{i=1}^{N} I(T_i \leq t)\omega_i)}$$

Brier Score is a measure of calibration in survival analysis. The Brier score is a weighted mean squared error that measures the difference between 0 (or 1) and the survival probability if the event did not happen until t (if the event happened before t) (Graf et al., 1999).

$$BS(t) = \frac{1}{N}\sum_{i=1}^{N} I(T_i \leq t, e_i = 1)\frac{\left(0 - \hat{S}(t \mid \boldsymbol{x}_i)\right)^2}{\hat{G}(T_i)} + I(T_i > t)\frac{\left(1 - \hat{S}(t \mid \boldsymbol{x}_i)\right)^2}{\hat{G}(t)}$$

5.3 Results and Analyses

We report the C^{td} score in Table 2a, the time-dependent AUC in Table 2b, and the Brier score in Table 2c. We measure all the scores at $t = 25\%, 50\%, 75\%$ event quantiles. The values of event quantiles of different datasets are in the rightmost columns of Table 1. From now on, we will denote Assumption-Free Survival Analysis as **AFreeSurv** and Assumption-Free Survival Analysis with Local Survival Smoothing regularizer as **AFreeSurv+**. Our approach beats the state-of-the-art baselines by comfortable margins on almost all metrics across all datasets. Figure 3 demonstrates the qualitative results.

Performance Evaluation on Real Data

We can observe intriguing results by looking at C^{td} and AUC, which measure the predictive power. The results on AUC in Table 2b shows that AFreeSurv outperforms other methods except that our method is on par with DeepHit at $t = 25\%, 50\%$ on GBSG. However, DeepHit and RSF beat our approach on METABRIC and GBSG on C^{td}. The observation means that if the failure times of the two individuals are before the metric measurement time, AFreeSurv does not distinguish the survival probabilities of those two individuals well. The indistinguishability of AFreeSurv implies that the survival probability after the failure does not properly decrease which, in turn, can be the

Table 2: Experimental results of various models on three public benchmarks.

(a) C^{td}

MODEL	SUPPORT			METABRIC			GBSG		
	25%	50%	75%	25%	50%	75%	25%	50%	75%
CoxPH	0.6888	0.6776	0.6720	0.6598	0.6381	0.6354	0.7351	0.7085	0.6972
DeepSurv	0.7046	0.6897	0.6745	0.6539	0.6455	0.6336	0.7168	0.7008	0.6883
DeepHit	0.7454	0.7174	0.6843	0.7501	0.6748	0.6227	0.7407	0.7196	0.6961
DRSA	0.7131	0.6906	0.6676	0.7197	0.6682	0.6516	0.7316	0.7072	0.6946
RSF	0.7611	0.7111	0.6613	0.7748	0.6750	0.6384	0.7372	0.7135	0.6984
DSM	0.7502	0.7157	0.6793	0.7540	0.6580	0.6190	0.7320	0.7168	0.7054
AFreeSurv	0.7588	0.7252	0.6882	0.7494	0.6734	0.6405	0.7373	0.7193	0.6987
AFreeSurv+	**0.7694**	**0.7304**	**0.6912**	**0.7837**	**0.6960**	**0.6530**	**0.7422**	**0.7212**	**0.7063**

(b) AUC

MODEL	SUPPORT			METABRIC			GBSG		
	25%	50%	75%	25%	50%	75%	25%	50%	75%
CoxPH	0.6940	0.7099	0.7228	0.6742	0.6599	0.6782	0.7522	0.7384	0.7401
DeepSurv	0.7055	0.7205	0.7131	0.6681	0.6676	0.6579	0.7329	0.7279	0.7308
DeepHit	0.7581	0.7495	0.7265	0.7644	0.6958	0.6750	**0.7609**	0.7519	0.7319
DRSA	0.7206	0.7220	0.7192	0.7378	0.6887	0.6755	0.7482	0.7370	0.7357
RSF	0.6507	0.6742	0.7006	0.6754	0.6507	0.6579	0.7431	0.7348	0.7357
DSM	0.7618	0.7506	0.7284	0.7684	0.6831	0.6551	0.7487	0.7480	0.7479
AFreeSurv	0.7763	0.7581	0.7364	0.7687	0.6959	0.6918	0.7538	0.7518	0.7366
AFreeSurv+	**0.7844**	**0.7597**	**0.7363**	**0.8017**	**0.7149**	**0.7010**	0.7597	**0.7527**	**0.7489**

(c) Brier score

MODEL	SUPPORT			METABRIC			GBSG		
	25%	50%	75%	25%	50%	75%	25%	50%	75%
CoxPH	0.1388	0.1982	0.2136	0.1103	0.1963	0.2225	0.1160	0.1826	0.2093
DeepSurv	0.1328	0.1928	0.2182	0.1134	0.1994	0.2413	**0.1091**	0.1802	0.2062
DeepHit	0.1323	0.1874	0.2133	0.1071	0.1973	0.2250	0.1103	0.1753	0.2068
DRSA	0.2611	0.2550	0.2388	0.1962	0.2456	0.2563	0.2140	0.2315	0.2397
RSF	0.1316	0.1968	0.2199	0.1030	0.1934	0.2269	0.1125	0.1784	0.2066
DSM	0.1327	0.1938	0.2110	0.1049	0.1969	0.2323	0.1098	0.1747	**0.1998**
AFreeSurv	0.1319	**0.1851**	**0.2095**	**0.1025**	0.1905	**0.2181**	0.1100	**0.1725**	0.2038
AFreeSurv+	**0.1289**	0.1855	0.2099	0.1026	**0.1893**	0.2210	0.1116	0.1747	0.2047

proof of the survival function's wiggliness. Note that this may be due to the lack of data given that the dataset size of METABRIC or GBSG is smaller than that of SUPPORT.

Unlike AFeeSurv, AFreeSurv+ consistently outperforms strong baselines across all datasets on C^{td} and AUC except that it is on par with DeepHit at $t = 25\%$ on AUC score on GBSG. Especially, AFreeSurv+ outperforms compared methods by a large margin on SUPPORT and METABRIC datasets. The results show that the *smoothness prior*, which is widely accepted in semi-supervised learning, is also an acceptable inductive bias in survival analysis. In other words, the proposed LS2 regularizer diminishes the wiggliness of the survival function and endows better predictive power.

The results in Table 2c shows that AFreeSurv is well-calibrated compared to competitive baselines. Our method outperforms other deep learning-based methods by a large margin on SUPPORT and METABRIC. The result is natural given that our AFreeSurv does not have any assumption while other methods have an assumption in any form. Comparing the results of AFreeSurv and AFreeSurv+, we can see that the proposed LS2 regularizer does not increase the Brier score. We speculate that what LS2 does is propagating high-quality labels to neighborhoods in covariate space, so they cannot increase the calibration power of AFreeSurv as AFreeSurv has already been trained with high-quality labels.

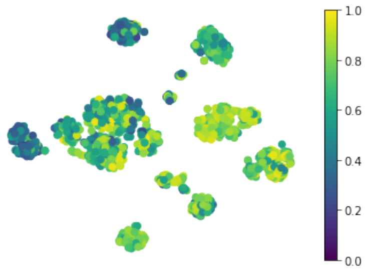
Superiority of Local Regularizer over Global Regularizer

To argue the necessity of *local* smoothness constraint, we compare the LS2 regularizer with global regularizers: weight decay and spectral normalization (Miyato et al., 2018a). See Table 3 for the results. Overall, the proposed LS2 regularizer outperforms two counterparts on the C^{td} and AUC scores and the LS2 is on par with two counterparts on the Brier score.

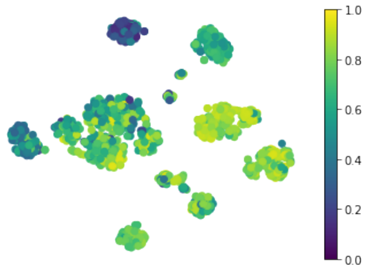
We run experiments of weight decay by setting the coefficient of the regularizer $\lambda =$ 1e-6, 1e-4, 1e-2. Across all datasets and metrics, there is little difference in performance. Though there is little difference in performance across all datasets and metrics, the performance is rather poor when $\lambda =$ 1e-2 so we report the result of $\lambda =$ 1e-4 for weight decay experiments.

We can observe that the spectral normalization deteriorates the performance of the model on SUPPORT and GBSG. The spectral normalization limits the Lipschitz constant of the network to a fixed constant by dividing each layer by the largest singular value of the layer. We conjecture that the global regularization of the function variation strongly restricts the expressivity of the function.

The *low-density separation*, which is widely accepted prior in semi-supervised learning (Chapelle et al., 2006), states that the decision boundary should lie in a low-density region. Though the prior is common in classification problems, it can also be applied to survival analysis. There may be cases where we need to model the differences in survival probabilities between different clusters, which leads to the necessity of modeling low function variation in high-density regions and high function variation in low-density regions.



(a) Without LS2



(b) With LS2

Figure 4: t-SNE plots (Van der Maaten & Hinton, 2008) of SUPPORT test data. Colors denotes the survival probability at $t = 50\%$ event quantile. It is easy to see that the LS2 regularizer makes the survival function locally smooth.

Also, we experimentally found that the LS2 regularizer is insensitive to hyperparameters: the coefficient λ and the neighborhood size ϵ. The hyperparameter insensitivity implies that the regularizer is easy for practitioners to use. See Table 4 and Table 5 for details.

6 Conclusion

We proposed assumption-free continuous-time survival function modeling, AFreeSurv, based on Neural ODEs. We also proposed the LS2 regularizer which minimizes the variation of the survival function in the covariate space. AFreeSurv outperforms state-of-the-art deep learning baselines on public datasets by a large margin as AFreeSurv does not make any assumptions. However, AFreeSurv is prone to become a wiggly function due to the constraint-free property. The proposed LS2 regularizer, which is inspired by the smoothness prior from semi-supervised learning, encourages the survival function to be locally smooth to increase the predictive power of the survival function. The combination of AFreeSurv and LS2 has better predictive power and is well-calibrated compared to strong baselines.

The performance gain thanks to the LS2 regularizer opens the possibility that developments in semi-supervised learning (Zhang et al., 2018; Yun et al., 2019; Kurakin et al., 2020) can also be applied to the survival analysis domain. Furthermore, we hope that recent developments in representation learning will be applied to the field of survival analysis.

REFERENCES

Ahmed M Alaa and Mihaela van der Schaar. Deep multi-task gaussian processes for survival analysis with competing risks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 2326–2334. Curran Associates Inc., 2017.

Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016. URL `http://arxiv.org/abs/1607.06450`.

Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien (eds.). *Semi-Supervised Learning*. The MIT Press, 2006. ISBN 9780262033589.

Paidamoyo Chapfuwa, Chenyang Tao, Chunyuan Li, Courtney Page, Benjamin Goldstein, Lawrence Carin Duke, and Ricardo Henao. Adversarial time-to-event modeling. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 735–744. PMLR, 10–15 Jul 2018. URL `https://proceedings.mlr.press/v80/chapfuwa18a.html`.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL `https://proceedings.neurips.cc/paper/2018/file/69386f6bb1dfed68692a24c8686939b9-Paper.pdf`.

David R Cox. Regression models and life-tables. *Journal of the Royal Statistical Society: Series B (Methodological)*, 34(2):187–202, 1972.

William Falcon et al. Pytorch lightning. *GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning*, 3:6, 2019.

David Faraggi and Richard Simon. A neural network model for survival data. *Statistics in medicine*, 14(1):73–82, 1995.

Tamara Fernandez, Nicolas Rivera, and Yee Whye Teh. Gaussian processes for survival analysis. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL `https://proceedings.neurips.cc/paper/2016/file/ef1e491a766ce3127556063d49bc2f98-Paper.pdf`.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=6Tm1mposlrM`.

Stephane Fotso et al. PySurvival: Open source package for survival analysis modeling, 2019–. URL `https://www.pysurvival.io/`.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL `https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf`.

Erika Graf, Claudia Schmoor, Willi Sauerbrei, and Martin Schumacher. Assessment and comparison of prognostic classification schemes for survival data. *Statistics in medicine*, 18(17-18):2529–2545, 1999.

Stefan Groha, Sebastian M Schmon, and Alexander Gusev. Neural odes for multi-state survival analysis. *https://arxiv.org/abs/2006.04893*, 2020.

Hung Hung and Chin-Tsang Chiang. Estimation methods for time-dependent auc models with survival data. *The Canadian Journal of Statistics / La Revue Canadienne de Statistique*, 38(1): 8–26, 2010. ISSN 03195724. URL `http://www.jstor.org/stable/27805213`.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 448–456, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/ioffe15.html.

Hemant Ishwaran, Udaya B Kogalur, Eugene H Blackstone, Michael S Lauer, et al. Random survival forests. *The annals of applied statistics*, 2(3):841–860, 2008.

How Jing and Alexander J. Smola. Neural survival recommender. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, WSDM '17, pp. 515–524, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450346757. doi: 10.1145/3018661.3018719. URL https://doi.org/10.1145/3018661.3018719.

Edward L Kaplan and Paul Meier. Nonparametric estimation from incomplete observations. *Journal of the American statistical association*, 53(282):457–481, 1958.

Jared L Katzman, Uri Shaham, Alexander Cloninger, Jonathan Bates, Tingting Jiang, and Yuval Kluger. Deepsurv: personalized treatment recommender system using a cox proportional hazards deep neural network. *BMC medical research methodology*, 18(1):24, 2018.

Patrick Kidger, Ricky T. Q. Chen, and Terry J Lyons. "hey, that's not an ode": Faster ode adjoints via seminorms. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 5443–5452. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/kidger21a.html.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun (eds.), *International Conference on Learning Representations*, 2015. URL http://arxiv.org/abs/1412.6980.

David G. Kleinbaum and Mitchel Klein. *Survival Analysis: A Self-Learning Text*. Springer-Verlag New York, 2012. ISBN 9781441966469. doi: 10.1007/978-1-4419-6646-9.

Alex Kurakin, Chun-Liang Li, Colin Raffel, David Berthelot, Ekin Dogus Cubuk, Han Zhang, Kihyuk Sohn, Nicholas Carlini, and Zizhao Zhang. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *NeurIPS*, 2020.

Changhee Lee, William R Zame, Jinsung Yoon, and Mihaela van der Schaar. Deephit: A deep learning approach to survival analysis with competing risks. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Jiayu Li, Hongyu Lu, Chenyang Wang, Weizhi Ma, Min Zhang, Xiangyu Zhao, Wei Qi, Yiqun Liu, and Shaoping Ma. A difficulty-aware framework for churn prediction and intervention in games. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, KDD '21, pp. 943–952, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383325. doi: 10.1145/3447548.3467277. URL https://doi.org/10.1145/3447548.3467277.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJzIBfZAb.

Bruce D Meyer. Unemployment insurance and unemployment spells. Working Paper 2546, National Bureau of Economic Research, March 1988. URL http://www.nber.org/papers/w2546.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018a. URL https://openreview.net/forum?id=B1QRgziT-.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1979–1993, 2018b.

Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012. ISBN 0262018020.

Chirag Nagpal, Xinyu Li, and Artur Dubrawski. Deep survival machines: Fully parametric survival regression and representation learning for censored data with competing risks. *IEEE Journal of Biomedical and Health Informatics*, 2021a.

Chirag Nagpal, Steve Yadlowsky, Negar Rostamzadeh, and Katherine Heller. Deep cox mixtures for survival regression. *Machine Learning for Healthcare Conference*, 2021b.

Patrick D. T. O'Connor and Andre Kleyner. *Introduction to Reliability Engineering*, chapter 1, pp. 1–18. John Wiley & Sons, Ltd, 2011. ISBN 9781119961260. doi: 10.1002/9781119961260. URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119961260.ch1`.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019. URL `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Sebastian Pölsterl. scikit-survival: A library for time-to-event analysis built on top of scikit-learn. *Journal of Machine Learning Research*, 21(212):1–6, 2020. URL `http://jmlr.org/papers/v21/20-729.html`.

Kan Ren, Jiarui Qin, Lei Zheng, Zhengyu Yang, Weinan Zhang, Lin Qiu, and Yong Yu. Deep recurrent survival analysis. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33 (01):4798–4805, Jul. 2019. doi: 10.1609/aaai.v33i01.33014798. URL `https://ojs.aaai.org/index.php/AAAI/article/view/4407`.

Hajime Uno, Tianxi Cai, Michael J Pencina, Ralph B D'Agostino, and Lee-Jen Wei. On the c-statistics for evaluating overall adequacy of risk prediction procedures with censored survival data. *Statistics in medicine*, 30(10):1105–1117, 2011.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Yuan Xue, Denny Zhou, Nan Du, Andrew M. Dai, Zhen Xu, Kun Zhang, and Claire Cui. Deep state-space generative model for correlated time-to-event predictions. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, pp. 1552–1562, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450379984. doi: 10.1145/3394486.3403206. URL `https://doi.org/10.1145/3394486.3403206`.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *International Conference on Computer Vision (ICCV)*, 2019.

Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=r1Ddp1-Rb`.

A Reproducibility Statement

Throughout all experiments, metrics are calculated using `scikit-survival` (Pölsterl, 2020).

A.1 Datasets

Throughout all datasets, we split the train, validation, and test datasets into ratios of 0.7, 0.1, and 0.2. We split the datasets using `PyTorch`'s `random_split` function and set the seed to 42. All datasets were standardized using `StandardScaler` of `scikit-learn` (Pedregosa et al., 2011).

SUPPORT We used the same version of SUPPORT dataset that DSM used [1]. We used 24 features: `age, income, num.co, meanbp, wblc, hrt, resp, temp, pafi, alb, bili, crea, sod, ph, glucose, bun, urine, adlp, adls, sex, dzgroup, dzclass, race`, and `ca`. For `wblc, bili, bun`, we added 1 and took $\log$. We turned `income` feature into the ordinal feature. Missing values were imputed using the mean value for numerical features and the mode for categorical features.

METABRIC We downloaded METABRIC dataset from `PyCox` package [2]. Missing values were imputed using the mean value.

GBSG We downloaded GBSG dataset from `PyCox` package. For `x5, x6` feature, we added 1 and took $\log$. Missing values were imputed using the mean value.

A.2 Methods Compared

For CoxPH, DeepSurv, and RSF, we used `PySurvival`'s implementation (Fotso et al., 2019–). For DeepHit, we used `PyCox`'s implementation. For DSM, we used official implementation [3]. Across all deep learning-based models, we stacked two hidden layers with dimension size 64. We experimented both with and without batch normalization (Ioffe & Szegedy, 2015) and reported a better result. The experiments of RSF were conducted by selecting the number of trees between 10 and 20, and we reported the best results among them. We discretized the time domain into 100 bins for DeepHit experiments. The experiments of DSM were conducted by selecting the number of primitive distributions between 3, 4, and 6. We reported the best results among them.

A.3 AFreeSurv and LS2 Regularizer

We implemented our model using `PyTorch` (Paszke et al., 2019) and `PyTorch Lightning` (Falcon et al., 2019). We stacked two hidden layers with dimension size 64 with layer normalization (Ba et al., 2016). We normalized the time t by $\frac{t-q_2}{q_3-q_1}$ where q_1, q_2 and q_3 are 25%, 50%, and 75% event quantile each. We set `atol, rtol` as 1e-5 and used `dopri5` solver. We used L_1 distance instead of L_2 distance in equation 9 across all experiments as we could get better results empirically. We used Adam optimizer (Kingma & Ba, 2015).

B Further Experiments

B.1 Comparison of Different Regularization Techniques

We compare different regularization techniques: weight decay, spectral normalization, and our local survival smoothing regularizer. See Table 3 for details.

B.2 Hyperparameter Sensitivity of LS2 regularizer

We show experimental results related to the sensitivity of our model to the two hyperparameters: λ and ϵ in equation 9. See Table 4 and Table 5 for details.

[1]https://github.com/autonlab/DeepSurvivalMachines/blob/master/dsm/datasets/support2.csv
[2]https://github.com/havakv/pycox
[3]https://github.com/autonlab/DeepSurvivalMachines

Table 3: Result comparison of different regularization techniques

(a) SUPPORT

MODEL	C^{td}			Brier Score			AUC		
	25%	50%	75%	25%	50%	75%	25%	50%	75%
vanilla	0.7588	0.7252	0.6882	0.1319	**0.1851**	**0.2095**	0.7763	0.7581	0.7364
weight decay	0.7588	0.7252	0.6882	0.1319	**0.1851**	**0.2095**	0.7763	0.7581	0.7364
spectral norm	0.7580	0.7243	0.6829	0.1333	0.1854	0.2114	0.7739	0.7575	0.7303
LS2 (ours)	**0.7694**	**0.7304**	**0.6912**	**0.1289**	0.1855	0.2099	**0.7844**	**0.7597**	**0.7363**

(b) METABRIC

MODEL	C^{td}			Brier Score			AUC		
	25%	50%	75%	25%	50%	75%	25%	50%	75%
vanilla	0.7497	0.6734	0.6405	**0.1025**	0.1905	**0.2181**	0.7687	0.6959	0.6918
weight decay	0.7497	0.6734	0.6405	**0.1025**	0.1905	**0.2181**	0.7687	0.6959	0.6918
spectral norm	0.7633	0.6851	0.6519	0.1028	**0.1883**	0.2191	0.7806	0.7074	0.7003
LS2 (ours)	**0.7837**	**0.6960**	**0.6530**	0.1026	0.1893	0.2210	**0.8017**	**0.7149**	**0.7010**

(c) GBSG

MODEL	C^{td}			Brier Score			AUC		
	25%	50%	75%	25%	50%	75%	25%	50%	75%
vanilla	0.7373	0.7193	0.6987	**0.1100**	**0.1725**	0.2038	0.7538	0.7518	0.7366
weight decay	0.7373	0.7193	0.6987	**0.1100**	**0.1725**	0.2038	0.7538	0.7518	0.7366
spectral norm	0.7350	0.7144	0.6986	0.1105	0.1731	**0.2011**	0.7507	0.7455	0.7390
LS2 (ours)	**0.7422**	**0.7212**	**0.7063**	0.1116	0.1747	0.2047	**0.7597**	**0.7527**	**0.7489**

Table 4: Coefficient λ sensitivity of LS2 regularizer when $\epsilon = 0.01$

(a) SUPPORT

MODEL	C^{td}			Brier Score			AUC		
	25%	50%	75%	25%	50%	75%	25%	50%	75%
$\lambda = 0$	0.7588	0.7252	0.6882	0.1319	0.1851	0.2095	0.7763	0.7581	0.7364
$\lambda = 10$	0.7620	0.7272	0.6908	0.1289	0.1857	0.2098	0.7769	0.7571	0.7357
$\lambda = 20$	0.7694	0.7304	0.6912	0.1289	0.1855	0.2099	0.7844	0.7597	0.7363
$\lambda = 40$	0.7803	0.7361	0.6810	0.1319	0.1891	0.2165	0.7949	0.7660	0.7308

(b) METABRIC

MODEL	C^{td}			Brier Score			AUC		
	25%	50%	75%	25%	50%	75%	25%	50%	75%
$\lambda = 0$	0.7494	0.6734	0.6405	0.1025	0.1905	0.2181	0.7687	0.6959	0.6918
$\lambda = 10$	0.7722	0.6895	0.6527	0.1018	0.1882	0.2200	0.7913	0.7114	0.7017
$\lambda = 20$	0.7837	0.6960	0.6530	0.1026	0.1893	0.2210	0.8017	0.7149	0.7010
$\lambda = 40$	0.7832	0.7047	0.6532	0.1042	0.1929	0.2255	0.8000	0.7197	0.6964

(c) GBSG

MODEL	C^{td}			Brier Score			AUC		
	25%	50%	75%	25%	50%	75%	25%	50%	75%
$\lambda = 0$	0.7373	0.7193	0.6987	0.1100	0.1725	0.2038	0.7538	0.7518	0.7366
$\lambda = 10$	0.7431	0.7232	0.7065	0.1106	0.1726	0.2026	0.7604	0.7558	0.7465
$\lambda = 20$	0.7422	0.7212	0.7063	0.1116	0.1747	0.2047	0.7597	0.7527	0.7489
$\lambda = 40$	0.7330	0.7124	0.7021	0.1132	0.1786	0.2110	0.7489	0.7426	0.7438

Table 5: Neighborhood size ϵ sensitivity of LS2 regularizer

(a) SUPPORT

MODEL	C^{td}			Brier Score			AUC		
	25%	50%	75%	25%	50%	75%	25%	50%	75%
$\epsilon = 0.1$	0.7680	0.7293	0.6906	0.1291	0.1858	0.2101	0.7830	0.7585	0.7359
$\epsilon = 0.01$	0.7694	0.7304	0.6912	0.1289	0.1855	0.2099	0.7844	0.7597	0.7363
$\epsilon = 0.001$	0.7695	0.7305	0.6913	0.1289	0.1855	0.2099	0.7846	0.7599	0.7363

(b) METABRIC

MODEL	C^{td}			Brier Score			AUC		
	25%	50%	75%	25%	50%	75%	25%	50%	75%
$\epsilon = 0.1$	0.7827	0.6961	0.6546	0.1025	0.1890	0.2206	0.8007		
$\epsilon = 0.01$	0.7837	0.6960	0.6530	0.1026	0.1893	0.2210	0.8017	0.7149	0.7010
$\epsilon = 0.001$	0.7837	0.6961	0.6530	0.1026	0.1893	0.2211	0.8016	0.7149	0.7009

(c) GBSG

MODEL	C^{td}			Brier Score			AUC		
	25%	50%	75%	25%	50%	75%	25%	50%	75%
$\epsilon = 0.1$	0.7417	0.7210	0.7061	0.1116	0.1746	0.2044	0.7593	0.7525	0.7487
$\epsilon = 0.01$	0.7422	0.7212	0.7063	0.1116	0.1747	0.2047	0.7597	0.7527	0.7489
$\epsilon = 0.001$	0.7423	0.7213	0.7062	0.1116	0.1747	0.2047	0.7598	0.7528	0.7485

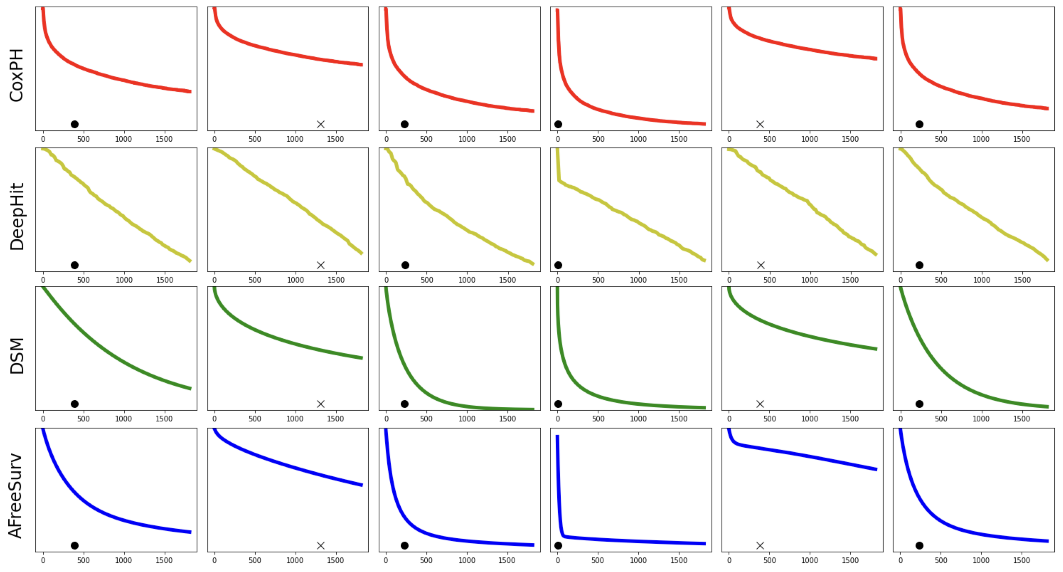


Figure 5: Survival probability plots of diverse samples. Uncensored data (●) and censored data (×) are drawn at the corresponding time.