DNNSurv: Deep Neural Networks for Survival Analysis Using Pseudo Values

A Preprint

Lili Zhao
Department of Biostatistics
University of Michigan
Ann Arbor, MI 48109
zhaolili@umich.edu

Dai Feng
Biometrics Research Department
Merck Research Laboratories
Kenilworth, NJ 07033 USA
fengdai73@gmail.com

August 8, 2019

Abstract

There has been increasing interest in modelling survival data using deep learning methods in medical research. Current approaches have focused on designing special cost functions to handle censored survival data. We propose a very different method with two steps. In the first step, we transform each subject's survival time into a series of *jackknife* pseudo conditional survival probabilities and then use these pseudo probabilities as a quantitative response variable in the deep neural network model. By using the pseudo values, we reduce a complex survival analysis to a standard regression problem, which greatly simplifies the neural network construction. Our two-step approach is simple, yet very flexible in making risk predictions for survival data, which is very appealing from the practice point of view. The source code is freely available at `http://github.com/lilizhaoUM/DNNSurv`.

***K*eywords** Deep learning · Neural network · Pseudo probability · IPCW · Risk prediction · Survival outcome

1 Introduction

Recently, using deep neural networks to predict when an event of interest will happen has gained considerable attention. These studies are often characterized by incomplete observations, in particular right-censored data; e.g. patients may be lost to follow-up without experiencing the event. Thus, handling the censored data becomes the crucial aspect of these analyses.

[1] developed a neural network assuming that survival time has a Weibull distribution. This parametric assumption is too restrictive in the analysis of large datasets. The semi-parametric Cox proportional hazards regression is the most widely used method for analyzing censored survival data. It studies the effects of the covariate variables on survival by estimating hazard functions, and it assumes that the hazard ratio for any two subjects is constant over time; that is,

the proportional hazards (PH) assumption. Several authors have adapted the Cox model to the deep neural networks [2, 3, 4]. They trained the deep neural network models by minimizing the negative partial likelihood defined under the PH assumption. However, this assumption is often questionable when the number of covariates is large, as every covariate needs to satisfy the PH assumption.

Another school of deep neural network modelling for survival data utilized the discrete-time survival model, including recent papers by [5, 6, 7, 8, 9]. In this modelling framework, the follow-up time is divided into a set of fixed intervals. For each time interval, the conditional hazard/probability is estimated: the probability of failure in the interval, given that the individual has survived up to the beginning of the interval. Compared to the Cox model, the discrete-time survival model is more flexible as it does not rely on the PH assumption. Furthermore, it directly estimates the survival probability, which is often of direct interest to patients and physicians than the hazard function in the Cox model. However, the censored data in each interval are handled either by an oversimplified method that assumes individuals with a censoring time in the second half of an interval survived that interval [5], or by an overcomplicated method that uses a ranking loss function [7] or a combination of a ranking loss and an isotonic regression [8].

All existing deep neural network models referenced above require the development of a special cost function to handle the censored data, sometimes with a sophisticated network structure, such as a convolutional or recurrent neural network [4, 9]. In this article, we develop a very different approach. To circumvent the complexity introduced by censored data, we substitute the observed survival times by *jackknife* pseudo observations, and then use these pseudo observations as a quantitative response variable in a regression analysis powered by the deep neural networks.

The remainder of this article is organized as follows. In section 2, we propose our two-step methods and address the covariate-dependent censoring problem. In section 3, we conduct simulation studies to evaluate our methods. In section 4, we apply our methods to two prospective, multicenter cohort studies for cardiovascular disease. We conclude this paper with a brief discussion in section 5.

2 METHOD

Notation. Let X_i be the survival time for subject i, C_i be the censoring time, $T_i = \min(X_i, C_i)$ be the observed survival time, and $\delta_i = I(X_i \leq C_i)$ be the censoring indicator. We assume that survival times and censoring times are independent. Let $\mathbf{Z_i} = (z_{i1}, \cdots, z_{ip})$ denote the p-dimensional covariates. We first assume that the censoring distribution does not depend on the covariates and then address the covariate-dependent problem in section 2.5.

2.1 Review the current pseudo-observations approach.

The pseudo-observations approach [10, 11, 12, 13] provides an efficient and straightforward way to study the association between the covariates and survival outcome in the presence of censoring.

If the outcome of interest is the survival probability a specific time t, without incomplete data, we could directly model $I(T_i > t)$ on $\mathbf{Z}_i$ $(i = 1, \cdots, n)$ using a generalized linear model (GLM) with a *logit* link, for a binary outcome variable.

In the presence of censoring, $I(T_i > t)$ is not observed for all subjects. In this case the Kaplan-Meier (KM) estimator can be used to estimate the survival probability at any given time point. The KM estimator is approximately unbiased under independent censoring [14], which is a requirement for the validity of the pseudo-observations approach. Based on the *jackknife* idea, a pseudo survival probability is computed for each (censored and uncensored) subject. For the i^{th} subject, the pseudo survival probability is computed by

$$\widehat{S}_i(t) = n\widehat{S}(t) - (n-1)\widehat{S}^{-i}(t), \tag{1}$$

where $\widehat{S}(t)$ is the KM estimator of $S(t)$ using all n subjects and $\widehat{S}^{-i}(t)$ is the KM estimator using sample size of $n-1$ by eliminating the i^{th} subject. Then $\widehat{S}_i(t)$ $(i = 1, \cdots, n)$ are used as a numeric response variable in the standard regression analysis, which is similar to model fit to $I(T_i > \tau)$, $(i = 1, \cdots, n)$, if these values were observed.

[11, 12, 15] proposed computing a vector of pseudo survival probabilities at a finite number of time points (equally spread on the event time scale) for each subject, and then modelling these pseudo survival probabilities as a function of the covariates by the generalized estimating equation (GEE), with the *complementary log-log* link, which is equivalent to fitting a Cox model to the survival data. The regression coefficient estimates from this pseudo-based GEE is approximately consistent when the censoring distribution is independent of the covariates and of the survival times [10].

2.2 Compute pseudo conditional probabilities.

In this article, we adapt the pseudo-observations approach in a discrete-time survival framework. We first divide the follow-up time into J intervals: $(0, t_1]$,$(t_1, t_2]$,$\cdots$, $(t_{J-1}, t_J]$. For each interval, we compute the pseudo conditional survival probability: the probability of surviving the interval, given that the subject has survived the previous interval. For a given interval $(t_j, t_{j+1}]$, all subjects who are still at risk is denoted by R_j, and the pseudo conditional probability of surviving t_{j+1} is computed as

$$\hat{S}_{ij}(t_{j+1}|R_j) = R_j\widehat{S}(t_{j+1}|R_j) - (R_j - 1)\hat{S}^{-i}(t_{j+1}|R_j), \tag{2}$$

where $\widehat{S}(t_{j+1}|R_j)$ is the KM estimator constructed using the remaining survival times for all patients still at-risk at time t_j, and $\widehat{S}^{-i}(t_{j+1}|R_j)$ is the KM estimator for all patients at-risk but the i^{th} subject. For the first interval $(0, t_1]$, all subjects are at risk (i.e., $R_0 = n$). If there is no censored data, the pseudo probability in each interval is either 0 or 1. With censoring, the pseudo probability is a real value; that is, it can be above 1 or below 0; see properties of the pseudo observations discussed in [13].

By using the discrete-time survival framework, we transform each subject's observed survival time (censored or uncensored) into a series of pseudo conditional survival probabilities. Unlike the pseudo-based GEE, in which the pseudo probabilities are computed from the marginal survival function (i.e., defined from time zero), the pseudo

Table 1: An example output from *getPseudoConditional*. PseudoProb denotes the pseudo conditional survival probabilities for a subject at a particular time point. In this example, there is only one covariate, z_1; additional columns could be added for more covariates.

ID	Time point	PseudoProb	z_1
1	0	1.03926	3.2
1	t_1	1.01747	3.2
1	t_2	-0.0141	3.2
2	0	1.03926	5.8
2	t_1	1.01747	5.8
2	t_2	1.02501	5.8
2	t_3	0.72582	5.8
3	0	1.03926	1.5
3	t_1	1.01747	1.5
3	t_2	1.02501	1.5
3	t_3	1.08026	1.5
3	t_4	1.18059	1.5

probabilities in our approach are computed from the conditional survival function and they are independent. Thus, we avoid the complexity of considering the within-subject correlation as in the pseudo-based GEE.

Choosing the time intervals needs to be done carefully. It has been shown that as few as five time intervals equally spread on the event time scale worked quite well in most cases [12]. One could also divide the time based on time points of clinical relevance; for example, patients' prognosis at 5 and 10 years might be of particular interest to the investigators. In either case, the interval can not be too small or too large. If the interval is too large, data information is lost. Conversely, if the interval is too small, no or a few events would occur in the interval, leading to an inefficient analysis. As fewer subjects remain in the study at later follow-up time, the upper bound of the last interval, t_J, should be reasonably smaller than the maximum follow-up time, so that we have sufficient information in the last interval. In this article, we set t_J at the 50^{th} or the 60^{th} percentile of the empirical overall survival distribution.

We created a R function called *getPseudoConditional* to compute the pseudo conditional probabilities described above. The R code is available in the GitHub repository `http://github.com/lilizhaoUM/DNNSurv/blob/master/getPseudoConditional.R`. Table 1 is an example output for the first three subjects. Each subject has multiple pseudo probabilities and one for each time point, but the number of pseudo probabilities may vary as different subjects might have different follow-up times. The covariate, z_1, has the same value for the same subject. The outputs from *getPseudoConditional* are then used as inputs in a deep neural network model.

2.3 Implement a deep neural network model

By using pseudo observations, the complex survival data analysis is reduced to a standard regression analysis with a single quantitative response variable. Thus, we directly employed the conventional cost function of the sum of squared errors for a regression problem in the deep neural network model. The input layer has $p+1$ nodes, including p covariates and one variable for the time points (i.e., the second column in Table 1). The output of the network is a single node, which estimates the survival probabilities at pre-specified time points. To constrain the probability between 0 and 1, we used the sigmoid activation function. We call this pseudo-based deep neural network model as DNNSurv and implemented it in the Keras library [16] in R with Tensorflow [17] backend (code at `http://github.com/lilizhaoUM/DNNSurv`).

To determine the hyperparameters in DNNSurv, we used the Cardiovascular Health Study (CHS) data, which contains 5380 subjects (see details in the REAL APPLICATION section). We determined the network structure and hyperparameters by using 5-fold cross-validation (CV) on the 75% randomly selected training set and evaluated the performance on the remaining 25% holdout test set.

We first experimented networks with different numbers of hidden layers, activation functions, optimization algorithms and regularization approaches available in Keras. Specifically, we compared network structures with one hidden layer and two hidden layers (4, 8, 16, 32, 64, or 128 nodes in each layer), under both the dropout regularization [18] (a dropout rate from 0.1 to 0.8 in each layer) and the ridge regularization [3] (a regularization penalty of 0.0001, 0.001, or 0.01 in each layer). For the activation function, we compared relu and tanh [16]. For the optimization algorithm, we compared Adaptive Moment Estimation (Adam) and rmsprop optimizer [16], with a learning rate of 0.001, 0.005 or 0.01 in both optimizers. Using 5-fold CV with the c-index as the performance metric, we found that a network with one or two hidden layers had similar performances; the dropout and ridge regularization had similar performances; tanh function had better performance than relu activation function, and Adam had better performance than rmsprop optimizer. Therefore, we decided on a network structure with two hidden layers, the tanh activation function in each hidden layer, the ridge regularizer and the Adam optimizer.

Next, we used a random Cartesian grid search [19] by a 5-fold CV to further tune the above hyperparameters, including number of nodes in each hidden layer, the regularization parameter in the ridge regularizer and the learning rate of the Adam optimizer. Finally, we determined a set of default values for the hyperparamters (see details in Table S1), which worked well in all simulation studies (see SIMULATIONS section). We recommend that users start with the default hyperparamters and adjust them if necessary.

2.4 Compute survival probabilities from DNNSurv

Given a set of $p + 1$ input values, DNNSurv is able to predict the conditional survival probability in each interval, that is, $P(T > t_j | T > t_{j-1})$ for $j = 1, \cdots, J$. The marginal survival probability, $P(T > t_j)$, is calculated by multiplying the conditional survival probabilities up to the j^{th} interval:

$$P(T > t_j) = \prod_{k=1}^{j} P(T > t_k | T > t_{k-1}).$$

Thus, the discrete-time survival framework allows us to predict both the marginal and the conditional survival probability, or the complementary risks. Both the marginal and conditional estimates have important clinical implications. For example, a patient who is diagnosed with lung cancer might be interested in the probability of surviving one year (i.e., the marginal survival probability). If the patient has survived the first year, then he/she might be interested in the probability of surviving another year (i.e., the conditional survival probability).

2.5 Handling covariate dependent censoring

KM estimates used in creating the pseudo values are subject to covariate-dependent censoring bias. In this case, we propose to use the inverse of probability of censoring weighted (IPCW) estimator for the survival function, denoted by $\hat{S}^W(t)$, which has been successful at reducing the bias [20, 21]). We replace $\hat{S}(t)$ by $\hat{S}^W(t)$ in (2) to compute the IPCW pseudo conditional survival probabilities by

$$\hat{S}^W_{ij}(t_{j+1}|R_j) = R_j\hat{S}^W(t_{j+1}|R_j) - (R_j - 1)\hat{S}^{W^{-i}}(t_{j+1}|R_j), \tag{3}$$

where $\hat{S}^W(t) = \exp\{-\hat{\Lambda}^W(t)\}$, and $\hat{\Lambda}^W(t)$ is the IPCW Nelson-Aalen estimator for the the cumulative hazard function and is estimated by

$$\hat{\Lambda}^W(t) = \sum_{i=1}^{n}\int_0^t \frac{dN_i(u)\hat{W}_i(u)}{\sum_{j=1}^{n} Y_j(u)\hat{W}_j(u)},$$

where $N_i(u) = I(T_i \leq u, \delta_i = 1)$ is the observable counting process for subject i, $Y_j(u) = 1(T_j \geq u)$ is the at risk process for subject j, and $\hat{W}_i(u)$ is the inverse of probability of censoring for subject i at time u. By assigning different weights for subjects based on their covariate values, $\hat{S}^W(t)$ is approximately unbiased if the censoring distribution is correctly specified [20]. Calculation of the IPCW pseudo conditional survival probabilities is also implemented in the *getPseudoConditional* function.

To model the censoring time distribution, we consider the Cox model, but other models, such as accelerated failure time model (AFT) [22] or deep neural networks, could work as well. In calculating the pseudo probabilities, we fit the censoring model once and then use the same censoring model for all subjects; that is, $\hat{W}_i(u)$ $(i = 1, \cdots, n)$ remain the same in computing the pseudo probabilities in (3). Given these IPCW pseudo probabilities, we apply the same deep neural network model as described in section 2.3 for the survival probability estimation. We refer this model as DDNSurv_ipcw in this article.

3 SIMULATIONS

We conducted simulation studies to evaluate our models in the case of independent and covariate-dependent censoring, respectively. We compared our models to Cox-nnet [3], which is a neural network trained by minimizing the negative partial likelihood defined under the PH assumption; see comparisons of the hyperparameters between DNNSurv and Cox-nnet in Table S1. We chose Cox-nnet for comparison as it has undergone peer review and has well documented source code, which can be found at `https://github.com/lanagarmire/cox-nnet`.

To evaluate the model performance, we considered two metrics commonly used in survival analysis. The first metric is the time-dependent concordance index (c-index) [23], which measures of how well a model predicts the ordering of sample event times. The other metric is the Brier score [24], which evaluates the accuracy of a predicted survival probability at a given time point by measuring the average squared distances between the observed survival status and the predicted survival probability (the smaller the better). Cox-nnet does not provide an estimate of the survival probability as DNNSurv; instead, it outputs a log hazard ratio ($\hat{\theta}_i$) for each subject. To compute the survival probability at a time point, we first computed the Nelson-Aalen estimate of the baseline survival function, $\hat{S}_0(t)$, and then computed the survival probability for subject i by $\hat{S}_i(t) = \hat{S}_0(t)^{\exp(\hat{\theta}_i)}$.

3.1 Simulations with independent censoring

Survival data were generated from an AFT model. Since the deep neural network model is able to approximate complex nonlinear functions, we considered the flexible random function generator in [25] for the mean function in the AFT model [26]. For all of the studies presented in this subsection, the number of variables is 20 (i.e., $\mathbf{Z} = (z_1, \dots, z_{20})$), and their joint distribution follows a standard multivariate normal distribution. The nonlinear mean function in the AFT model takes the form

$$\mu(\mathbf{Z}) = \sum_{l=1}^{10} a_l g_l(\mathbf{Z}_{(1)}), \tag{4}$$

where coefficients a_l's were generated from a uniform distribution $a_l \sim U[-1, 1]$. Each g_l is a Gaussian function of a randomly selected variables, $\mathbf{Z}_{(1)}$, of the 20 variables; see details in [25] on the selection of these variables and the Gaussian function construction. The expected number of variables for each g_l function was 4. By using a linear combination of 10 g_l functions, the mean in the AFT model is a function of all, or nearly all, of the 20 variables with different strength of association with the survival outcome, and it also involves higher-order interactions between some of the variables. Finally, we generated the residuals in the AFT model from a gamma distribution with a shape parameter of 2 and a rate parameter of 1, resulting in a signal-to-noise ratio of 3.

We simulated 100 datasets and each dataset has a different mean function, $\mu(\mathbf{Z})$, generated from (4). The censoring times were independently generated from an exponential distribution. A different rate parameter was chosen to obtain a censoring rate of 0.2, 0.4, or 0.6, which corresponds to light, moderate and heavy censoring, respectively. We generated 5000 observations, 75% of which were randomly chosen as training data and the remaining 25% were test data. We divided the time into six 10% intervals from approximately the 10^{th} to the 60^{th} percentile of the empirical overall survival distribution.

As shown in Figure 1, DNNSurv outperformed the Cox-nnet in the majority of the simulated datasets, as evidenced by the higher C-index and lower Brier scores in DNNSurv compared to Cox-nnet. It is worth noting that the performance of DNNSurv is even better when the censoring rate is high.

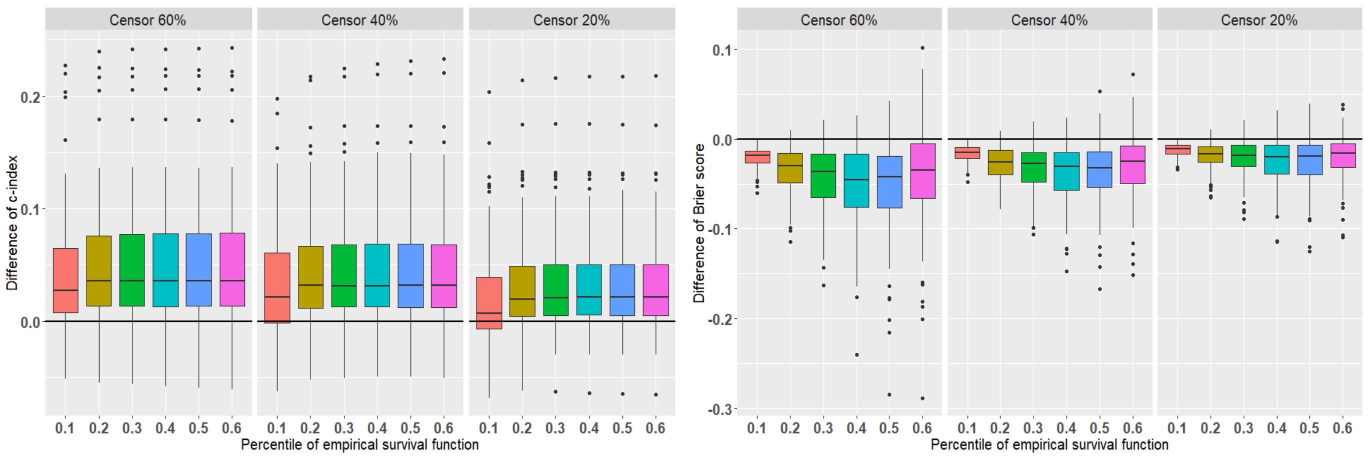

Figure 1: Boxplots of the differences in c-index (left) and Brier score (right) between the DNNSurv and Cox-nnet (Cox-nnet subtracted from DNNSurv) over 100 simulations generated from the AFT model with Friedman's random function generator

We performed a sensitivity analysis by reducing the 6 intervals into 3 intervals of 20% at the 20^{th}, 40^{th}, and 60^{th} percentiles of the empirical overall survival distribution. We found that the c-index and Brier scores remained almost the same (boxplots are shown in Supplementary Figure 1S).

3.2 Simulations with covariate dependent censoring

First, we used the same model in section3.1 to generate survival times, but we generated covariate-dependent censoring times. Specifically, the censoring times were generated from a Cox model using the five most frequently selected variables in the g_l functions in (4). In each simulated dataset, we controlled the censoring rate at approximately $40 \sim 50\%$.

As shown in Figure 2, both DNNSurv and DDNSurv_ipcw performed significantly better than Cox-nnet, and DDNSurv_ipcw was slightly better than DDNSurv, as demonstrated by the larger c-index in DDNSurv_ipcw.

To further investigate the IPCW estimator, we set up a simple simulation study. In this study, survival data were generated from a Cox model with one covariate: $0.1\exp(\beta z)$, and $\beta = 1$ and z was simulated from a standard normal distribution. The censoring model followed the same Cox model as the survival times, resulting in approximately 50% censoring rate in each simulated dataset. We applied five models to each dataset: DDNSurv, DDNSurv_ipcw, Cox-nnet, pseudo-based GEE model as described in 2.1 and GEE_ipcw (i.e., the GEE using the IPCW pseudo values). In each simulation, we randomly selected 2000 subjects to train the model and predict survival probabilities for a separate set of 2000 subjects at 10^{th}, 20^{th}, 30^{th}, 40^{th}, and 50^{th} percentiles of the overall survival distribution.

Based on 100 simulations, we found that GEE_ipcw reduced the bias in the estimation of the regression coefficient in presence of covariate-dependent censoring. The estimate for β, averaged over 100 simulated datasets, was 0.783 with a mean squared error (MSE) of 0.05 in GEE, while the β estimate was 1.002 with a MSE of 0.0026 in GEE_ipcw (remember the true $\beta = 1$). Therefore, in correcting the bias in the GEE model, we further validated the IPCW method.

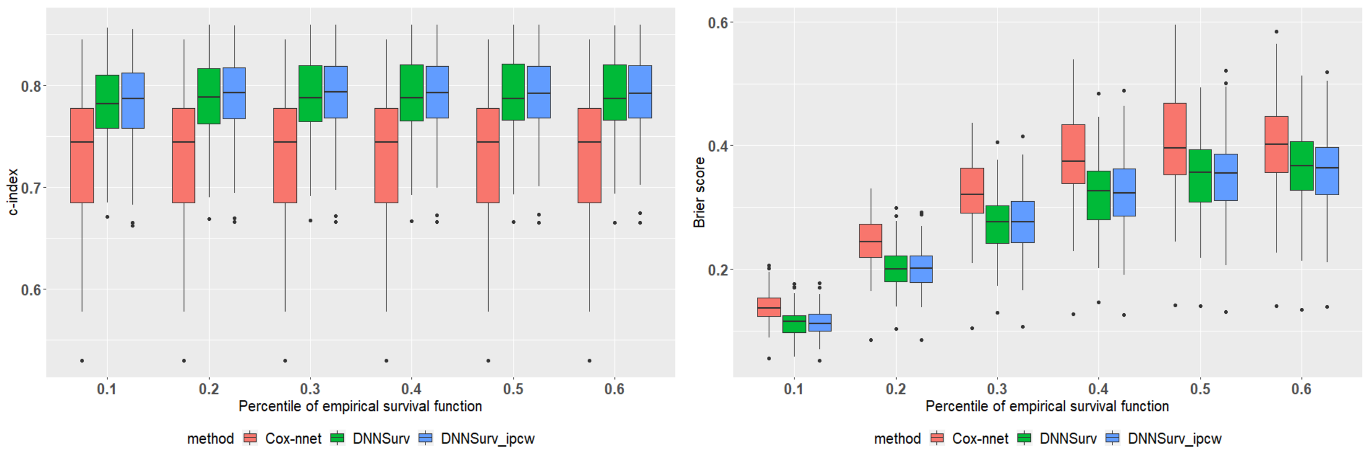

Figure 2: Box plots of c-index (left) and brier score (right) for the 100 simulated datasets generated from the AFT model with Friedman's random function generator in the case of independent censoring.

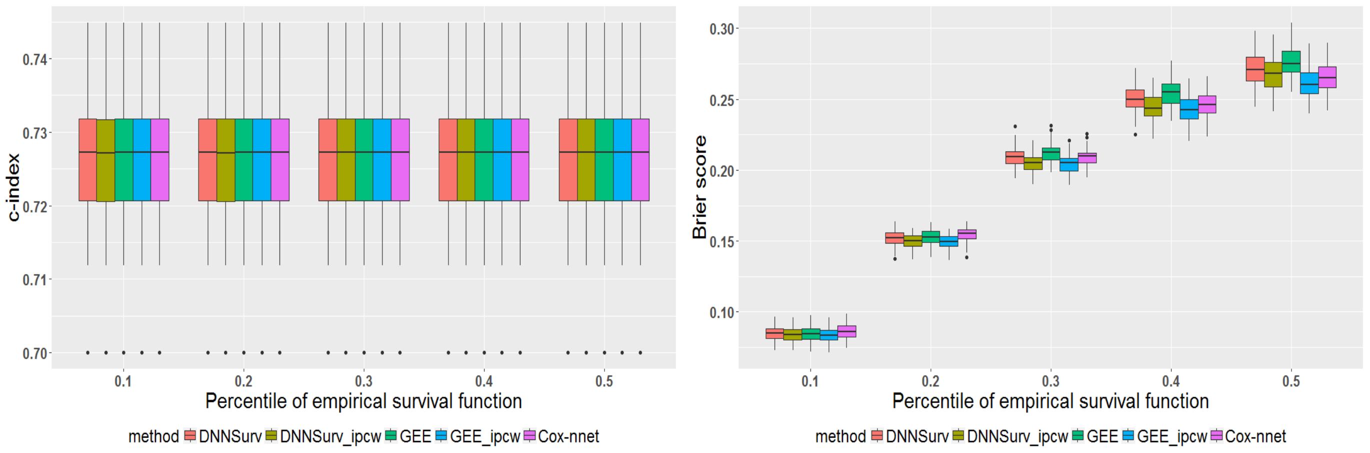

Figure 3: Box plots of c-index (left) and Brier score (right) for the 100 simulated datasets generated from the simple Cox model in the case of covariate-dependent censoring.

Figure 3 shows box plots of c-index over 100 simulations. It is interesting to note that all the studied methods have the same c-index. This could due to the fact that the rank-based c-index measure is not sensitive to the bias when the covariate effect is monotonic on survival.

The accuracy measure, Brier scores in Figure 3, shows more interesting findings: 1) GEE had the worst performance due to the bias, whereas GEE_ipcw had the best performance as it addressed the bias correctly, and the model matched the true data generating model; 2) DNNSurv_ipcw improved the prediction accuracy over the DNNSurv, and it was reasonably close to GEE_ipcw; 3) compared to GEE, DNNSurv was less sensitive to the violation of the independent censoring assumption; and 4) it is surprising to see that DNNSurv_ipcw performed better than Cox-nnet as the data were generated under the PH assumption.

4 REAL APPLICATION

We applied our DNNSurv to two prospective, multicenter cohort studies for coronary heart disease and stroke. Both studies are sponsored by the National Heart Lung and Blood Institute of the National Institutes of Health.

The Cardiovascular Health Study (CHS) was initiated in 1987 to determine the risk factors for development and progression of cardiovascular disease (CVD) in older adults, with an emphasis on subclinical measures. Detailed description of the study can be found in [27]. The event of interest was time to CVD. The study has collected a large number of variables at baseline, including demographics (e.g., age, gender and race), family history of CVD, lab results and medication information, with the goal to identity important risk factors for the CVD event. We selected 29 predictor variables to make predictions of the CVD event; see a complete description of the variables in Table S2.

After excluding subjects with missing data in any of the selected predictor variables, we had $5,380$ subjects and 65.2% of which had CVD during the study period. We randomly selected 75% subjects as training data and the other 25% as test data. We then trained the data using DNNSurv and Cox-nnet. In DNNSurv, we used a model-averaging idea to obtain the risk predictions. Specifically, after the grid search, rather than using a single best set of hyperparameters to train the model, we used five best sets of hyperparameters to train the model five times, resulting in five models. Each model was then applied to the test data to predict the survival probability for each subject at each time point. The final estimated survival probability is the averaged survival probabilities from the five models. We found that by averaging five models, the prediction performance is more stable compared to using a single best model; for example, the prediction was less sensitive to initial values.

We predicted the survival probability in every year up to 15 years. In CHS data, an interval of one year is not very small, as the event rate (65.2%) is large and reasonable numbers of the event were observed in each interval. DNNSurv and Cox-nnet had the same model performance. The c-index is 0.705 in both methods and Brier scores are almost the same (see Table S3). Figure 4 shows the survival functions for some predictor variables, which were known to be important predictor variables for the CVD event. The estimated survival probabilities from DNNSurv are very similar to those from the Cox-nnet, which indicates that the proportional hazard assumption is reasonable in this dataset.

The second CVD study we considered is the Multi-Ethnic Study of Atherosclerosis (MESA) study, which enrolled subjects who were free from clinical cardiovascular disease from six communities in the United States in $2000-2002$. Participants were followed for identification and characterization of cardiovascular disease events. Detailed description of the study can be found in [28]. Similar to the CHS study, the event of interest was time to CVD. We have selected 30 variables to make predictions of the CVD event; see a complete description of the predictor variables in Table S4. After excluding subjects with missing data in any of the selected predictor variables, we had $6,547$ subjects and 5.4% of which had CVD during the study period. The event rate is very small, an interval of one year would be too small, as it is likely no event in some intervals. Thus, we predicted the survival probability only at 3 and 5 years, which are clinically meaningful landmark time points.

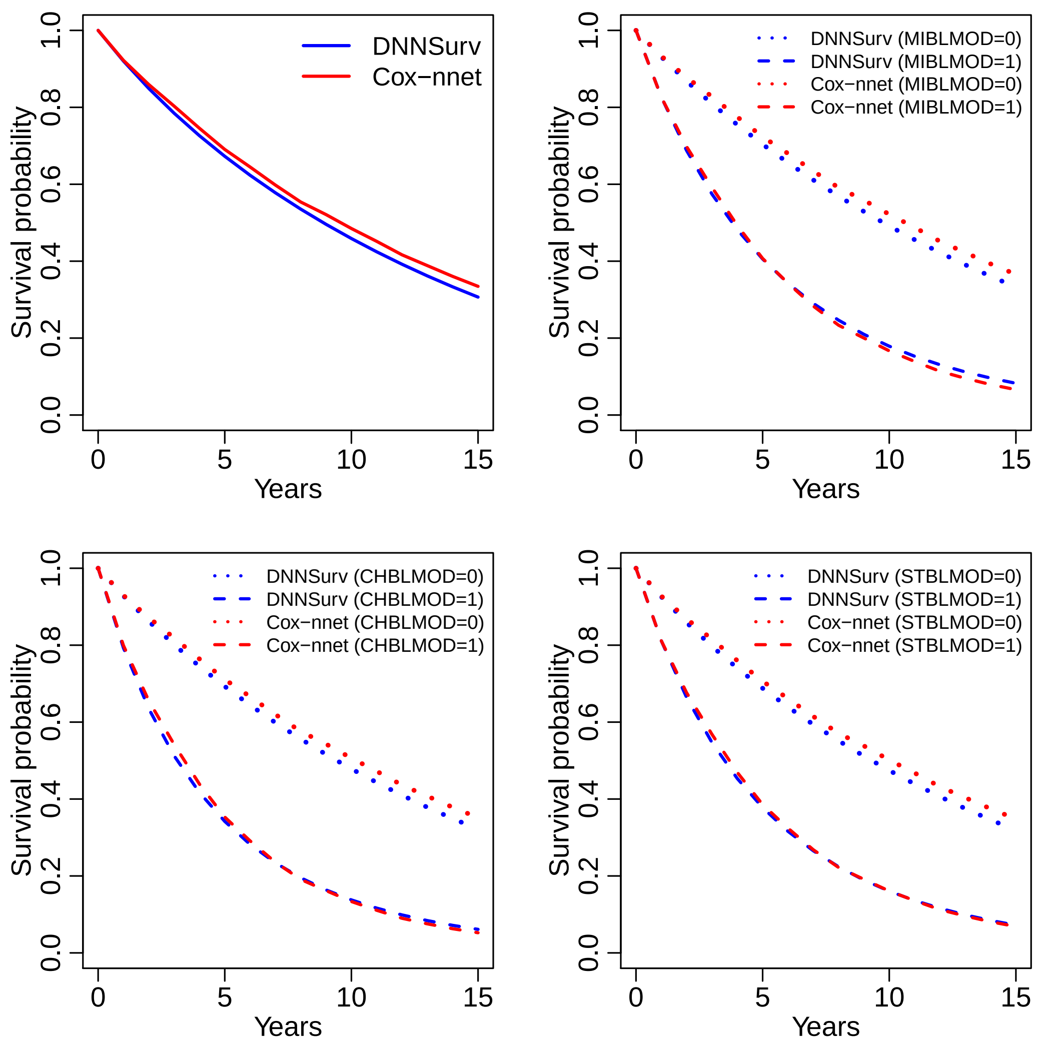

Figure 4: Survival functions for the CHS data estimated from both the Cox-nnet and DNNSurv. The first plot shows the overall survival functions using both Cox-nnet and DNNSurv. The remaining three plots present the survival functions stratified by the MI status (MIBLMOD), CLD status (CHBLMOD), or the stroke status (STBLMOD), respectively.

Table 2: The results for mesa data

	Cox-nnet		DNNSurv	
	year 3	year 5	year 3	year 5
c-index	0.74	0.74	0.75	0.75
Brier Score	0.03	0.04	0.03	0.04

We randomly selected 75% subjects as training data and the other 25% as test data. Table 2 shows that DNNSurv and Cox-nnet had similar prediction performance. We also tried DNNSurv_ipcw with weights calculated based on four variables that were significantly associated with the censoring time using the Cox model. Both c-index and Brier score remained the same as the DNNSurv.

5 CONCLUSION

In this article, we develop a two-step approach for making risk predictions in survival analysis using a deep neural network model. The first step is to compute the *jackknife* pseudo survival probabilities in the discrete-time survival framework, and substitute the survival times by these pseudo probabilities in the deep neural networks to make risk predictions. The IPCW pseudo probabilities are used in case of the covariate-dependent censoring. By using the pseudo values, the analysis for censored survival data is reduced to a regression problem with a quantitative response variable, which greatly facilitates the use of deep learning methods. Standard deep neural networks can be directly applied, which avoids the difficulty of designing a special cost function for the censored data, as in the current methods.

We demonstrated superior performance of DNNSurv to Cox-nnet in all simulation studies. In real data analyses, these two methods had very similar performance. This could be due to the validity of the PH assumption in the real datasets. From another perspective, DNNSurv is a competing alternative to Cox-nnet when the data satisfies the PH assumption. Compared to Cox-nnet, DNNSurv is much simpler to implement, and it outputs survival or risk probabilities, which are often of direct interest to patients and physicians than the hazards in Cox-nnet.

The pseudo-observations approach offers a great opportunity to study right-censored survival data using deep neural networks. It can be generalized to analyze survival data with competing risks [10, 11, 15] and the restricted mean survival time (RMST) analysis [29, 21, 30]. The proposed pseudo-value calculations in the discrete-time framework would allow risk predictions in the presence of competing risks. The RMST is often of great clinical interest in practice, especially in the presence of non-proportional hazard functions. However, the estimation procedure is often complicated as the RMST involves an integration of the survival function. Therefore, only a few parametric methods (such as Cox or GEE) are available for such analysis [29, 21, 30]. By using the pseudo observations, the deep neural network model is directly applicable in estimating the RMST.

6 Acknowledgements

This Manuscript was prepared using the CHS and MESA data obtained from the National Heart, Lung, and Blood Institute (NHLBI) Biologic Specimen and Data Repository Information Coordinating Center and does not necessarily reflect the opinions or views of the NHLBI.CHS and MESA data were obtained from the National Heart, Lung, and Blood Institute (NHLBI) Biologic Specimen and Data Repository Information Coordinating Center (`https://biolincc.nhlbi.nih.gov/`)

Additional Files

Additional Table S1 — Hyperparameters in Cox-nnet and DNNSurv

Additional Table S2 — Description of variables in CHS study

Additional Table S3 — Brier scores for DNNSurv in the CHS data

Additional Table S4 — Description of variables in MESA study

References

[1] E. Martinsson. WTTE-RNN: Weibull time to event recurrent neural network. Master's thesis, University of Gothenburg, Sweden, 2016.

[2] J. L. Katzman, U. Shaham, A. Cloninger, J. Bates, T. Jiang, and Y. Kluger. Deepsurv: personalized treatment recommender system using a cox proportional hazards deep neural network. *BMC medical research methodology*, 18:24, 2018.

[3] T. Ching, X. Zhu, and L. X. Garmire. Cox-nnet: An artificial neural network method for prognosis prediction of high-throughput omics data. *PLoS Comput. Biol.*, 14:e100607, 2018.

[4] Jiawen Yao Xinliang Zhu and Junzhou Huang. Deep convolutional neural network for survival analysis with pathological images. *IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 2016.

[5] Michael F. Gensheimer and Balasubramanian Narasimhan. A scalable discrete-time survival model for neural networks. *arXiv preprint arXiv:1805.00917*, 2018.

[6] Stephane Fotso. Deep neural networks for survival analysis based on a multi-task framework. *arXiv preprint arXiv:1801.05512*, 2018.

[7] C. Lee, W. R. Zame, J. Yoon, and M. van der Schaar. Deephit: A deep learning approach to survival analysis with competing risks. *AAAI*, 2018.

[8] M. Luck, T. Sylvain, H. Cardinal, A. Lodi, and Y. Bengio. Deep learning for patient-specific kidney graft survival analysis. *arXiv preprint arXiv:1705.10245*, 2017.

[9] Eleonora Giunchiglia, Anton Nemchenko, and Mihaela van der Schaar. RNN-SURV: A deep recurrent model for survival analysis. *International Conference on Artificial Neural Networks*, 90:15–27, 2018.

[10] P. K. Andersen, J. P. Klein, and S. Rosthøj. Generalised linear models for correlated pseudo-observations, with applications to multistate models. *Biometrika*, 90:15–27, 2003.

[11] J. P. Klein and P. K. Andersen. Regression modeling of competing risks data based on pseudovalues of the cumulative incidence function. *SIM*, 61:223–229, 2005.

[12] P. K Andersen and J. P. Klein. Regression analysis for multistate models based on a pseudo-value approach, with applications to bone marrow transplantation studies. *SIM*, 34:3–16, 2007.

[13] P. K Andersen and M. P. Perme. Pseudo-observations in survival analysis. *Statistical Methods in Medical Research*, 19:71–99, 2010.

[14] P. K. Andersen, ø. Borgan, R. D. Gill, and N. Keiding. *Statistical models based on counting processes*. Springer-Verlay, New York, 1993.

[15] John P Klein, Mette Gerster, P. K. Andersen, Sergey Tarima, and Maja Pohar Perme. SAS and R functions to compute pseudo-values for censored data regression. *Comput Methods Programs Biomed.*, 89:289–300, 2008.

[16] François Chollet et al. Keras. `https://keras.io`, 2015.

[17] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[18] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[19] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305, 2012.

[20] Nadine Binder, Thomas A. Gerds, and Per Kragh Andersen. Pseudo-observations for competing risks with covariate dependent censoring. *Lifetime Data Anal.*, 20:303–315, 2014.

[21] Fang Xiang and Susan Murray. Restricted mean models for transplant benefit and urgency. *Statistics in Medicine*, 6:561–576, 2012.

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

[23] Frank E. Harrell, Kerry L. Lee, and Daniel B. Mark. Tutorial in biostatistics: multivariable prognostic models: issues in developing models, evaluating assumptions and adequacy, and measuring and reducing errors. *Statistics in Medicine*, 15:361–387, 1996.

[24] P. K Andersen and M. P. Perme. Consistent estimation of the expected brier score in general survival models with right-censored event times. *Biometrical Journal*, 48:1029–1040, 2006.

[25] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[26] Nicholas C Henderson, Thomas A Louis, Gary L Rosner, and Ravi Varadhan. Individualized treatment effects with censored data via fully nonparametric bayesian accelerated failure time models. *arXiv preprint arXiv:1706.06611*, 2017.

[27] Grethe S.Tell DrPhilos, Linda P.Fried, BonnieHermanson, Teri A.Manolio, Anne B.Newman, and Nemat O.Borhani3. Recruitment of adults 65 years and older as participants in the cardiovascular health study. *Annals of Epidemiology*, 3:358–366, 1993.

[28] D. E. Bild, D. A. Bluemke, G. L. Burke, R. Detrano, A. V. Diez Roux, A. R. Folsom, P. Greenland, D. R. Jacob, R. Kronmal, K. Liu, J. C. Nelson, D. O'Leary, M. F. Saad, S. Shea, M. Szklo, and R. P. Tracy. Multi-ethnic study of atherosclerosis: objectives and design. *Am J Epidemiol.*, 156(9):871–881, 2002.

[29] Xin Wang and Douglas E. Schaubel. Modeling restricted mean survival time under general censoring mechanisms. *Lifetime Data Analysis*, 24:176–199, 2018.

[30] P. K. Andersen, M. G. Hansen, and J. P. Klein. Regression analysis of restricted mean survival time based on pseudo-observations. *Lifetime Data Anal*, 10:335–350, 2004.