Survival Analysis of Heart Failure Patients

Souradeep Sen
Department of Computer Science,
University of Exeter
(Dated: July, 2023)

The study aims to compare the performance of Deep Learning (DL) architectures for predicting mortality in heart failure (HF) patients, against traditional survival analysis techniques. The aim is to see if deep learning can help estimate survival probabilities based on contextual and historic features from clinical data, and how these predictions perform against traditional techniques. By leveraging longitudinal patient data from Electronic Health Records (EHR), an examination of the performance of machine learning risk prediction models is conducted against conventional survival analysis in HF patients. The findings may have important implications for clinical practice, healthcare resource allocation, and future research in risk prediction modeling for HF patients.

1. INTRODUCTION

Heart failure is a clinical syndrome interfering with the heart's ability to pump blood, leading to a reduction in systemic circulation performance. This condition is widespread globally - as of 2017, approximately 64 million people worldwide are estimated to be affected by heart failure [1]. Hence, accurately predicting risk is crucial for improving patient outcomes.

In traditional survival analysis, accounting for time-variant patient characteristics may require experimentation and extensive domain knowledge. It can also be limited in its ability to handle complex non-linear dependencies. Deep learning is an exciting tool in this aspect due to its innate ability to handle complex non-linear relationships [2]. As deep learning is generally data-hungry, the advent of EHR data seems promising to be used in conjunction with this framework. This study attempts to examine if deep learning outperforms traditional survival analysis in terms of prediction accuracy using real-world EHR data.

The paper is organized as follows. Section 2 contains a brief review of related work. Section 3 presents a more rigorous understanding of survival analysis, before building up to how discrete-time survival likelihood may be expressed in terms of hazard rates [3] and can be parameterized by a neural network. A suitable loss function is chosen from the available literature and is derived as per [4]. Section 4 delves into Uncertainty Quantification, by way of Monte Carlo (MC) dropout [5] and looks at explainability of the model(s) built through the use of SHAP values [6]. Section 5 discusses the data used for the paper. Section 6 looks at the results by means of experiments over real and synthetic data and compares the architechture to existing solutions from deep learning and traditional survival analysis. Section 7 is reserved for proposed extensions to the model and further work that could be done.

2. RELATED WORK

Deep learning methods in conjunction with classical machine learning, have recently started gaining traction in clinical settings - see [7], [8], [9], [10] and [11]. They are starting to be adopted in the field of survival analysis as well. A deep neural network model with learned medical feature embedding is proposed in [12] to address high dimensionality and temporality in electronic health record (EHR) data. Here, a convolutional neural network is used to capture non-linear longitudinal evolution of EHRs and local temporal dependency for risk prediction, and embed medical features to account for high dimensionality. Experiments show promising results in predicting risks for congestive heart failure.

Personalized predictive modeling is investigated in [13], which aims to build specific models for individual patients using similar patient cohorts to capture their specific characteristics. According to this study, although CNNs have shown promise on measuring patient similarity, one disadvantage is that they could not utilize temporal and contextual information of EHRs. To measure patient similarity using EHRs, the authors proposed a time-fusion CNN framework. A vector representation was generated for each patient, which was then utilized for measuring patient similarity and personalized disease prediction. Dynamic updates to a CNN model are explored in [14] as more data is gathered over time - this architecture lends itself well to real-time mortality risk prediction. Maintaining interpretability across deep learning models is explored in [15].

Extensions to the classical Cox proportional hazards model were first proposed in [16] and later in [17]. More complex architectures like RNNs were applied to ingest time-varying patient data anad generate risk scores in [18]. Discrete time survival predictions were addressed in [19], [20] and [21]. This study aims to consider incidences as time-to-event to enable probabilistic risk prediction for mortality. The use of EHR such as those available in MIMIC-IV, allows access to comprehensive longitudinal

data, which captures the entire cycle of a host of medical factors such as patients' diagnosis and treatment.

3. SURVIVAL ANALYSIS

A. Basics

Survival analysis deals with the estimation of a survival distribution representing the probability of an event of interest, typically a failure, to occur beyond a certain time in the future [20]. One way to specify the survival distribution is through the survival function. The survival function defines the probability of surviving till point t [22].

$$S(t) = P(T > t),\ 0 < t < \infty \tag{1}$$

It can be thought of as the complement of the cumulative distribution function $F(t)$.

$$S(t) = P(T > t) = 1 - P(T \leq t) = 1 - F(t),\ 0 < t < \infty \tag{2}$$

Another way to specify the survival distribution is through the hazard function, which denotes the instantaneous rate of failure.

$$h(t) = \lim_{\delta \to 0} \frac{P(t < T < t + \delta | T > t)}{\delta} \tag{3}$$

For the continuous-time scenario, the hazard function and survival function are related as follows.

$$\begin{aligned} f(t) &= \frac{d}{dt}F(t) = -\frac{d}{dt}S(t) \\ h(t) &= \frac{f(t)}{S(t)} \end{aligned} \tag{4}$$

where $f(t)$ is the probability mass function (PMF). This says that the hazard is the probability of the subject experiencing an event at time t, provided that the subject is alive till time t. It can be further simplified as

$$\begin{aligned} h(t) &= -\frac{d}{dt}S(t)\frac{1}{S(t)} \\ \implies S(t) &= exp\left(-\int_0^t h(u)du\right) \end{aligned} \tag{5}$$

This relationship produces the survival function from a corresponding hazard function [22].

$$H(u) = \int_0^t h(u)du$$

is known as the cumulative hazard, which is also extensively studied throughout the literature.

B. A Brief Review of Traditional Fitters

In the context of continuous time survival models, the Cox Proportional Hazards model has long been the 'gold-standard' for survival analysis [23]. As seen in [17], extensions have been made for Cox models to learn non-linear hazards (although the proportional hazards assumption remains). The assumption of a proportional hazards model is that the covariates have a multiplicative effect on the hazard.

$$h(t|x) = h_0(t)e^{w^T x} \tag{6}$$

where h_0 is called as the baseline hazard.

Proportional hazards models are learned by optimizing Cox's partial likelihood in classical survival analysis [24]. A general formulation is derived as follows [22]. Consider failure time t_i. The set of all subjects in the trial "at risk" for failure at this time is denoted by $j : Y_j > Y_i$, where Y denotes the observed survival time. The probability of patient i failing at this time is

$$L_i = \frac{h(t_i|x_i)}{\sum_{j:Y_j > Y_i} h(t_i|x_j)} \tag{7}$$

where $h(t|x)$ denotes the hazard for patient with covariates x at time t. The likelihood of the entire cohort is thus,

$$L = \prod_{i:C_i=1} L_i = \prod_{i:C_i=1} \frac{h(t_i|x_i)}{\sum_{j:Y_j > Y_i} h(t_i|x_j)} \tag{8}$$

where $C_i = 1$ denotes an observed (uncensored) event. The assumption of a proportional hazards model reduces this to

$$L = \prod_{C_i=1} \frac{e^{w^T x_i}}{\sum_{j:Y_j > Y_i} e^{w^T x_j}} \tag{9}$$

It is noted that the baseline hazard h_0 is canceled from both the numerator and the denominator [24]. The log likelihood therefore becomes

$$\ell = \sum_{i:C_i=1} \left(w^T x_i - log \sum_{j:Y_j > Y_i} e^{w^T x_j} \right) \tag{10}$$

Interestingly, this negative of this log-likelihood is used as the loss function for DeepSurv and the Faraggi-Simon net.

Accelerated failure time models are a popular parametric alternative to the Cox PH model [25]. They are especially handy in discriminating between groups where the survival times of one can be uniformly shifted backward to forward to get the survival times of another. Unlike Cox PH model, where the baseline hazard is unspecified, AFT models directly model the survival times [26].

Random Survival Forest models have managed to achieve state-of-the-art performance. At the cost of higher fitting times, they deliver high discriminative power with impressive calibration [27]. Building on Breiman's approach in [28], this method uses the log-rank test between survival cohorts to determine the best splits. RSF have a selection bias towards covariates with many possible splits or missing values. Improvements towards mitigating this have been made in the form of Conditional Inference Forests which use more nuanced criteria to split nodes, via unbiased recursive partitioning [29].

C. Data Setup

Before moving to the discrete setting, some formal notation for the data is introduced. The data is assumed to be right-censored (starting times for all subjects are known, but ending times are not). Hence, the data, $\mathcal{D}$ can be represented as a set of tuples $\{(x_i, t_i, d_i)\}_{i=1}^{N}$ [20]. Here, $x_i \in \mathbb{R}^d$ are covariates for patient i. t_i is the time of an event or censoring such that $t_i = min(T_i, C_i)$, where T_i and C_i respectively denote the times of event and censoring. A subject is assumed to have either experienced an event or have been censored, but not both. d_i is an indicator that signifies whether t_i is event time or censoring time. $d = 1$ for a subject that experiences the event (uncensored) while $d = 0$ for a subject that is censored before experiencing the event. More formally, $d = \mathbb{1}\{T_i \leq C_i\}$. Later in the paper, experiments with time-variant covariates will necessitate the use of a null masking matrix, $\mathcal{M}$ [21].

D. Discrete-Time Survival Analysis

For hazard and survival calculation in a discrete-time setting, the following formulation from [4] is presented. Let $\mathbb{T} = \{\tau_1, \tau_2, \dots\}$ denote the timestamps, i.e. the indices of the discrete times corresponding to different subjects in the data. The event timestamp is $T* \in \mathbb{T}$. The definitions of PMF and survival function follow as

$$\begin{aligned} f(\tau_j) &= P(T* = \tau_j), \\ S(\tau_j) &= P(T* > \tau_j) = \sum_{k>j} f(\tau_k) \end{aligned} \tag{11}$$

It can be seen that the hazard at time τ_j $h(\tau_j)$, is just the probability of an event happening at time τ_j, given the subject has survived till the previous time step τ_{j-1}.

$$h(\tau_j) = P(T* = \tau_j | T* > \tau_{j-1}) = \frac{f(\tau_j)}{S(\tau_{j-1})} \tag{12}$$

$$\begin{aligned} \implies h(\tau_j) &= \frac{S(\tau_{j-1}) - S(\tau_j)}{S(\tau_{j-1})} \\ \implies S(\tau_j) &= (1 - h(\tau_j))S(\tau_{j-1}) \end{aligned}$$

Recursively, the survival function can be parameterized wholly in terms of the hazard function as

$$S(\tau_j) = \prod_{k=1}^{j} (1 - h(\tau_k)) \tag{13}$$

Equation 13 will be useful in later derivations. Now, coming to likelihoods, if there were no censoring involved, the likelihood of observing n failures (events) is of the form

$$L(t_1, t_2, \dots, t_n) = f(t_1)f(t_2)\dots f(t_n) = \prod_{i=1}^{n} f(t_i) \tag{14}$$

As per [22], for an observed event, the pdf is retained. But for a right-censored observation, it is replaced by the survival function, as that subject is known only to have survived past that time. The likelihood then becomes

$$L(t_1, t_2, \dots, t_n) = \prod_{i=1}^{n} f(t_i)^{\delta_i} S(t_i)^{1-\delta_i} = \prod_{i=1}^{n} h(t_i)^{\delta_i} S(t_i) \tag{15}$$

E. Loss Function(s)

The architecture incorporates two loss functions - one being the negative log likelihood of the data [4], another being a lower bound on the concordance index [24]. For the derivations, see Appendix A 4

These two loss functions are linearly combined based on a hyperparameter $\alpha \in [0, 1]$ to make the total loss

$$\mathcal{L}_{total} = \alpha \mathcal{L}_{nll} + (1 - \alpha)\mathcal{L}_{lbo} \tag{16}$$

F. Neural Network Parameterization

As hazards are conditional probabilities (see Eq. 12), they must lie within $[0, 1]$. A handy function for this is the sigmoid non-linearity

$$g(x) = \frac{1}{1 + e^{-x}} \tag{17}$$

For a neural network taking x_i (the covariates of patient i) as input and producing m outputs denoting m discrete timesteps in the patient's journey, applying the sigmoid function effectively transforms the outputs into valid hazards.

$$h(\tau_j | x_i) = g(\phi_j(x_i)) = \frac{1}{1 + e^{-\phi_j(x_i)}} \tag{18}$$

where $\phi_j(x_i)$ is the output of the neural network at node j corresponding to the hazard function over the time period $[\tau_{j-1}, \tau_j)$.

Returning to the problem at hand, for such a parameterization, the continuous time scale $\mathbb{T}$ needs to be discretized into m pieces. A simple way to do this is to equally divide up the set $\mathbb{T}$ into m equal parts. The hazards across these m timesteps can be cumulatively multiplied to find the survival function as seen in Eq. 13.

4. UNCERTAINTY QUANTIFICATION AND EXPLAINABILITY

An addition this work attempts to make to the above is by introducing a layer of uncertainty quantification via Monte-Carlo dropout [5]. Dropout was originally introduced in 2014 [30] as a method to intorduce regularization into neural networks by randomly dropp units along with their connections. This prevents units from co-adapting too much, thereby forcing their neighbours to 'learn' the signal in a more 'robust' fashion. Dropout is generally reserved for training as a means to enforce regularization on the network. Libraries like PyTorch allow users to prime networks for training or evaluation. When applied during the testing or prediction phase as well, a probability distribution of the output is created, allowing for probabilistic inference instead of point inference as present in regular neural networks. Over many iterations, this essentially means that the network used for prediction is slightly different each time, resulting in slightly different outputs - thereby allowing users to generate a probability distribution of outputs.

Being uncertainty-aware ties in with the idea of interpretability of modern deep learning methods, which is often a barrier towards the adoption of such methods into highly regulated industries such as finance or medicine. SHAP (SHapley Additive exPlanations) values [6], extended from the field of cooperative game theory provide a unified, model-agnostic method to explain the contributions of covariates to a model's output.

5. MODEL ARCHITECTURE

This study attempts to model survival functions from both a time-invariant perspective (using means of covariates) and from a time-variant perspective (using historical data for covariates). The latter takes some more preprocessing and requires a slghtly more complex architecture.

A. Time-Invariant

The simpler time-invariant version of the architecture is a multi-layer perceptron (MLP), otherwise referred to as a fully-connected neural network. Consider a batch of input covariates $x \in \mathbb{R}^{n \times d}$, where n is the batch size and d is the number of input covariates. The basic architecture is as outlined in Table I.

Layer Type	Shape/Rate
Dense (nn.Linear)	(d, h)
ReLU (nn.ReLU)	-
Batch Normalization (nn.BatchNorm1d)	h
Dropout (nn.Dropout)	0.5
Dense (nn.Linear)	(h, h)
ReLU (nn.ReLU)	-
Batch Normalization (nn.BatchNorm1d)	h
Dropout (nn.Dropout)	0.5
Dense (nn.Linear)	(h, m)

TABLE I. General Architecture - Time-Invariant

Here, m is the output size, denoting the number of discretized time indices across time $max(\mathbb{T})$, while h denotes the number of hidden nodes in the linear layers.

B. Time-Varying

First, a patient image is built for each subject in the dataset. Consider a batch of input covariates $x \in \mathbb{R}^{n \times t_s \times d}$, where t_s is the number of time-steps along the subjects' journey(s). n and d retain their meaning from earlier.

A convolutional neural network is used to identify patterns in the patient-image, whose outputs are then fed into a fully-connected network. The architecture is elaborated in Table II.

Layer Type	Shape/Rate
Convolutional (nn.Conv2d)	$(1, 14)$
ReLU (nn.ReLU)	-
Convolutional (nn.Conv2d)	$(14, 7)$
ReLU (nn.ReLU)	-
Flatten (nn.Flatten)	-
Dense (nn.Linear)	(fc^1, h)
ReLU (nn.ReLU)	-
Batch Normalization (nn.BatchNorm1d)	h
Dropout (nn.Dropout)	0.5
Dense (nn.Linear)	(h, h)
ReLU (nn.ReLU)	-
Batch Normalization (nn.BatchNorm1d)	h
Dropout (nn.Dropout)	0.5
Dense (nn.Linear)	(h, m)

TABLE II. General Architecture - Time-Variant

Here, kernel_size = 3, stride = 1 and padding = 1.

6. DATASET

The study uses the large publicly available database MIMIC-IV [31], which consists of critical care data from

hospital and ICU admissions for almost 300,000 patients admitted to intensive care units at the Beth Israel Deaconess Medical Center (BIDMC). For a more thorough treatment of the data, see MIMIC-IV website.

A. Preprocess

To prepare the data, all admissions associated with a Heart Failure ICD-10 code are selected - this formsa the base pool of patients. Admission and discharge times are collected along with static covariates such as patients' gender, age and date of death. Patients' ethnicity is collected and grouped into six broad categories - Native, Asian, Black, Hispanic, White and Other [2].

Time-variant records such as BMI, Weight and Height are collected from Online Medical Records (OMR). Such records have an associated chart-time or chart-date, denoting when they were collected. Lab tests corresponding to cholesterol, sodium intake, lymphocyte count and hemoglobin levels were considered. Medication administered from the classes of angiotensin-converting enzyme blockers, angiotensin receptor blockers, calcium channel blockers and beta blockers are also taken as features. Vital signs such as temperature, heartrate, respiratory rate, O_2 saturation, systolic and diastolic blood pressure are also taken. Patients who have at least one record for all of the above datasets (OMR, lab test, medication and vital signs) are retained. For the time-varying version, the time difference between each subsequent observation is noted. Only patients who have at least 10 distinct time steps are retained. This of course, brings down the number of patients slightly as compared to the time-invariant version.

B. Censoring

Patients' earliest admission time is considered as their 'start' date. If their death date is not captured in the data [3], their last known discharge time is known as their 'end' date - these patients are considered to be censored. Otherwise, their date of death is taken to be the 'end' date - these are the uncensored patients. The distribution of event/censoring times is quite similar across both censored and uncensored cohorts - see Figure 1.

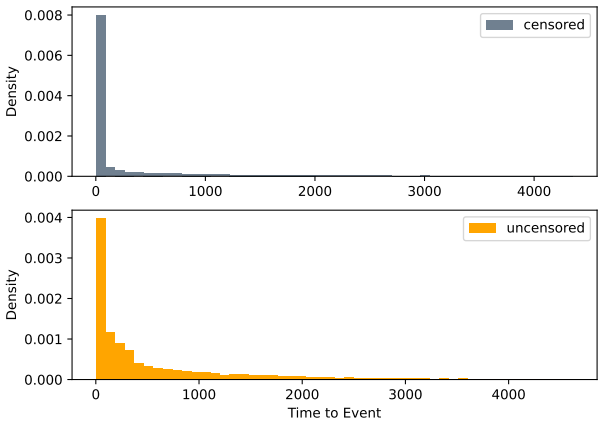

FIG. 1: Cohort-wise Histogram showing Censored vs. Uncensored subjects

C. Data Standardization

Patients are randomly distributed into train, test and validation pools. Columns with zero variance in the training data are discarded from all three datasets - train, test and validation. Across each pool, their survival time is discretized into $q = 15$ buckets or time intervals. For example, a patient whose survival time belongs to the j^{th} bucket has experienced the event (or been censored) between the time interval $[q_j, q_{j+1})\ \forall j \in [1, 2, \cdots, m]$.

For the time-invariant version, simple pipelines for imputation and scaling are carried out. For the time-varying version, the process is slightly more involved. First, a patient-level cumulative time difference column is introduced, to keep track of when medications or test were administered - for every patient, this column starts with 0.

Next, a null mask $\mathcal{M}$ is created, as in [21]. As neural networks cannot natively handle missing data, the matrices need to be imputed (strategies will be discussed later). However, it can be reductive to ignore the fact that some data is missing in the original dataset. Therefore, a compromise is reached by creating a null mask before imputing the missing values. This null mask is then appended to the dataset before being fed to the neural network. The expectation is for the network to consider imputed values as missing when their corresponding mask element $\mathcal{M}_{idj} = 0$, denoting the j^{th} observation of covariate d for patient i is missing.

As convolutional neural networks expect input of the same shape, the patient images need to be quantized to

[2] To avoid multi-collinearity issues arising later, the 'OTHER' variable is dropped after one-hot encoding. Similary, 'F' variable corresponding to gender (female) is also dropped after one-hot encoding.

[3] Out-of-hospital mortality is captured from state records. If the individual survived for at least one year after their last hospital discharge, then the death date will have a NULL value.

have the same width [4] and the same height. The latter is achieved by sampling the multivariate time series data at set points during the patients history. For this study, a patient's history is divided into 10 equal parts, and all their covariates are interpolated over these divisions. Finally each column is normalized as per the L2 norm. Finally, they are reshaped to be fed into a PyTorch CNN model.

7. EVALUATION METRICS

As survival analysis differs from ordinary linear regression in the aspect that not all the survival times are known (right-censoring), there is merit in reviewing the evaluation metrics for this task. The two metrics used in the experimental setup are the time-dependent concordance index (in favour of the regular concordance index) and the Brier score (along with its aggregated version, the integrated Brier score).

A. Time-Dependent Concordance Index (*td*-concordance index)

The C-index, often referred to as 'Harrell's C-Index' or simply as the C-statistic is a measure of the discriminative capacity of a model. In essence, it is the generalization of the area under the ROC (Receiver Operator Characteristic) curve in a survival analysis setting [32] [33]. The formula for the c-index is given as

$$c = \frac{1}{|\mathcal{E}|} \sum_{i:C_i=1} \sum_{T_j>T_i} \mathbb{1}_{\eta_j>\eta_i} \tag{19}$$

where η_i denotes the predicted survival time for subject i. As c-index is rank-based index, it can be substituted with the survival probability of subject i as well. $|\mathcal{E}|$ is the number of pairs that *can* be compared.

It can be interpreted as the fraction of all pairs of subjects whose predicted survival times are correctly ordered among all subjects that can actually be ordered [34]. A pair (i, j) is considered 'comparable' if the one with the lower observed time is uncensored, that is, when $T_i < T_j$, then $d_i = 1$. A pair is considered 'concordant' when the model predicts a higher risk for the patient with a lower survival time. Thus, it follows that

$$c - index = \begin{cases} 1.0, & \text{perfect concordance,} \\ 0.5, & \text{equivalent to random classification,} \\ 0.0, & \text{perfect anti-concordance.} \end{cases}$$

[4] This is already done as all the patients have the same number of covariates.

There are limitations with the traditional c-index as highlighted by [35], an important one being the assumption that the risk scores do not change over time. [36] propose a time-dependent concordance index which essentially takes a weighted average of time-specific C-index values across the entire time scale.

B. Brier Score

Contrasting the discriminatory power of the C-index, the Brier score provides a measure of how well the model is calibrated. It represents the distance between observed and predicted survival probability - hence 0 is the most desirable value. Given the data, $\mathcal{D}$, a set of tuples $\{(x_i, t_i, d_i)\}_{i=1}^{N}$ and the predicted survival function $\hat{S}(t, x_i),\ \forall t \in \mathbb{R}^+$, the Brier score (without right-censored observations) assumes a form similar to the mean-squared-error.

$$BS(t) = \frac{1}{N} \sum_{i=1}^{N} (\mathbb{I}_{T_i>t} - \hat{S}(t, x_i))^2 \tag{20}$$

With the occurence of right-censored data, the formula needs to be adjusted by the inverse probability of censoring weights method [37]. Let $\hat{G}(t) = P[C > t]$ be the estimator of the conditional survival function of the censoring times calculated using the Kaplan-Meier method, where C is the censoring time.

$$BS(t) = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{(0 - \hat{S}(t, x_i))^2 \cdot \mathbb{I}_{T_i \le t, d_i=1}}{\hat{G}(T_i)} + \frac{(1 - \hat{S}(t, x_i))^2 \cdot \mathbb{I}_{T_i>t}}{\hat{G}(t)} \right) \tag{21}$$

An aggregated version of this score provides a simpler summary of the calibration of a model.

$$IBS(t_{max}) = \frac{1}{t_{max}} \int_0^{t_{max}} BS(t) dt \tag{22}$$

8. EXPERIMENTS AND FITS

A series of experiments were conducted on this data from both a time-invariant and a time-varying perspective. The survival distributions were first examined with a non-parametric fitter, the Kaplan-Meier fitter (see: Figure 2). A short write-up on these methods can be found in the Appendix A. All relevant code for this study can be found on github.

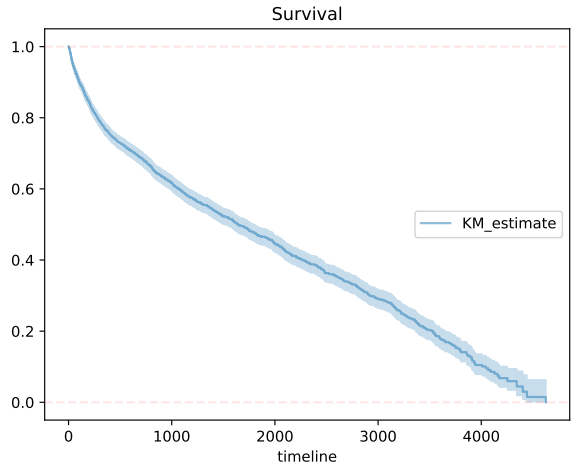

FIG. 2: Kaplan-Meier curve of Survival for entire cohort

Next, a range of parametric, semi-parametric, tree-based and neural network-based methods were applied to the data and their evaluation criteria noted. To establish confidence intervals on said criteria, fits were run multiple times - all time-invariant models apart from Cox Proportional Hazards model and Weibull Accelerated Failure Time model were run 20 times. The model allowing for time-varying data was run 10 times. Thus, distributions of their cindex and integrated Brier scores were generated.

One of the earlier aims of this study was to see if hybrid deep learning, that is the supplementation of deep learning with classical machine learning techniques would yield better performance compared to traditional techniques. In that light, although methods such as clustering and PCA are introduced into the pipeline, the highest discriminatory power (comparable to traditional methods) is generally observed if the data is directly fed into the neural network(s). Due to this shortcoming, the time-variant version has not been equipped with any clustering utility as opposed to its time-invariant counterpart. Some tests were carried out to see the effect of having multiple clusters and preprocessing the data with PCA. Details are in Table III.
The details of experiments comparing the two architectures with other off-the-shelf methods are listed in Table V. It is noted that although the upper bound for the time-invariant method is above that of the traditional statistical fitters (CPH and AFT), its mean is below theirs. The time-variant counterpart seems to comfortably beat all other methods in terms of c-index (discriminative power) but both methods have quite a large Brier score (calibration) in comparison to off-the-shelf methods. The Figures 3 and 4 showcase this situation.

This implementation is equipped with a wrapper for the Python library `shap` which allows explanation of particular observations. SHAP allows the introduction of a notion of an 'expected' [5] survival curve, which can be compared against baseline survival of traditional fitters and the empirical survival distribution from non-parametric fitters such as the Kaplan-Meier fitter. Figure 5 shows this.

Extension	**Hyperparameters**	**C-Index**	**IBS**
Clustering	clusters = 1	0.6957	0.3995
Clustering	clusters = 2	0.6857	0.4075
Clustering	clusters = 3	0.6786	0.3915
Clustering	clusters = 4	0.6724	0.4108
Clustering	clusters = 5	0.6703	0.4141
PCA	n_components = 1	0.4891	0.4147
PCA	n_components = 2	0.5071	0.4127
PCA	n_components = 5	0.4927	0.4133
PCA	n_components = 10	0.4826	0.4049
Clustering & PCA	clusters = 2, n_components = 2	0.5103	0.4131

TABLE III. Evaluation metrics - Degradation with extensions. Note these metrics were collected on one specific run. Reruns may yield slightly different results

p-values	**CPH**	**KM**	**NN**
CPH	-	0.0019	0.1877
KM	-	-	0.0461

TABLE IV. Results of KS test

This leads to the question of statistical similarity of the curves thus generated. The Kolmogorv-Smirnov test [38] is used to test for this, with the null hypothesis being that the curves are 'similar', i.e. the generating process for both the curves is the same. As standard, two critical values of 0.01 and 0.05 are chosen. It is seen that the p-value for the test between CPH and the neural network is above 0.05; therefore the null hypothesis cannot be rejected and it follows that the baseline survival from a traditional fitter such as Cox Proportional Hazards is statistically similar to the 'expected' survival generated from the NN[6]. The case for comparison with the empirical distribution is more of an edge case, allowing rejection of the null hypothesis at critical value of 0.05 but not at 0.01.

From here, it may be natural to want to investigate the factors that cause anindividual subject's survival to deviate (at any particular timestep) from the 'expected' survival at that time. Figure 6 illustrates how a particular subject's survival curve is shifted from the expected curve. Figure 7 shows the breakdown for one patient at

[5] In the literature, expected survival generally refers to the mean or median survival time and not a time-varying metric.
[6] Refers to the expected survival curve generated from the time-invariant neural network in this study

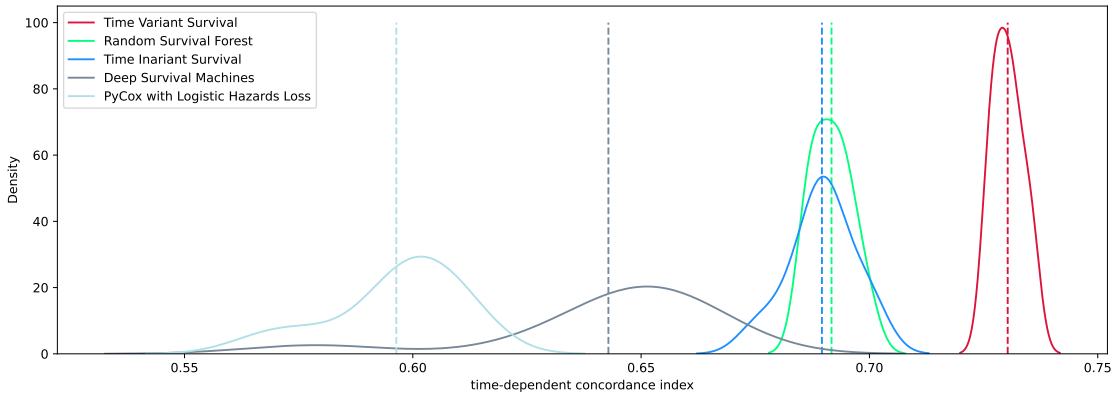

FIG. 3: Distribution of C-index - Time-Variant version run for 10 iterations. All others run for 20 iterations.

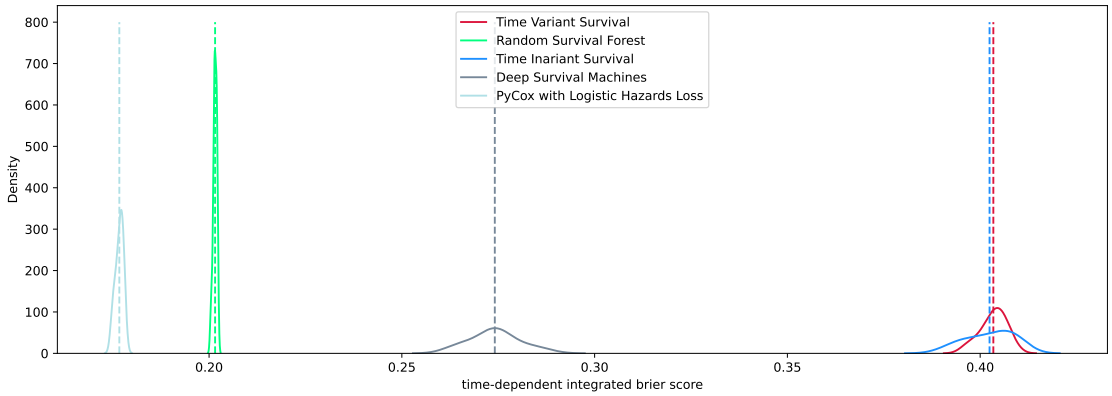

FIG. 4: Distribution of IBS - Time-Variant version run for 10 iterations. All others run for 20 iterations.

a particular discrete timestep.

Furthermore, this implementation generates confidence intervals by applying Monte-Carlo dropout. For any particular subject, the returned survival curve natively has confidence bands associated with it. They can be seen again in Figure 6.

9. FURTHER WORK

Neural network methods capable of temporal learning (forms of RNN) can be applied to this problem to see if the discriminative power can be increased. This study has not applied several modern neural network optimizations like Nesterov momentum and learning rate scheduling - further work can focus on these aspects. Deeper architectures may be applied to the problem, however, longer training times may discourage this. This study has used evaluation metrics implemented in [4]. Currently the time-dependent concordance index calculation appears to take a good amount of time to generate results. This can be a point for further improvement. From a modularization perspective, classes for Time-Invariant and Time-Variant processes are completely separate at the moment. Perhaps later they can inherit common functionalities from a parent class.

In future, if the clustering utility is investigated in more detail, one might find that some parts of the code break due to the random initialization of clusters sometimes producing cohorts whose duration indices do not match the number of requested discretizations. This is a bug and may warrant investigation. For evaluation criteria, another popular metric in the survival analysis literature is cumulative dynamic AUC [39]. Future iterations can focus on this metric along with the two others mentioned in this study to see if it yields a different assessment of the models' suitability.

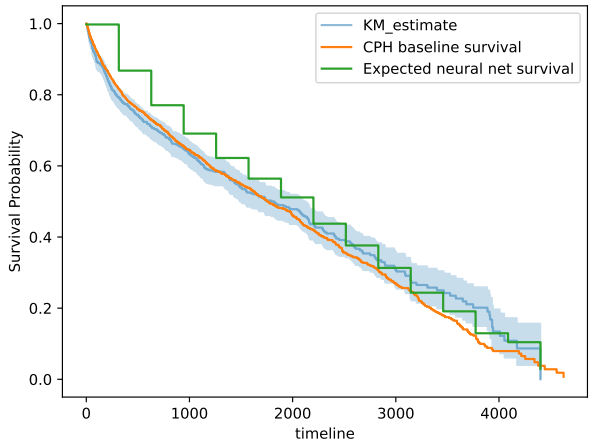

FIG. 5: Comparison of expected survival curve against baseline survival and empirical survival

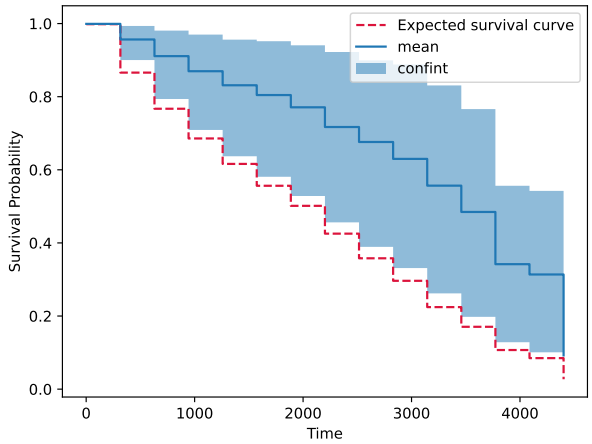

FIG. 6: An individual subject's survival curve differs from the expected survival curve

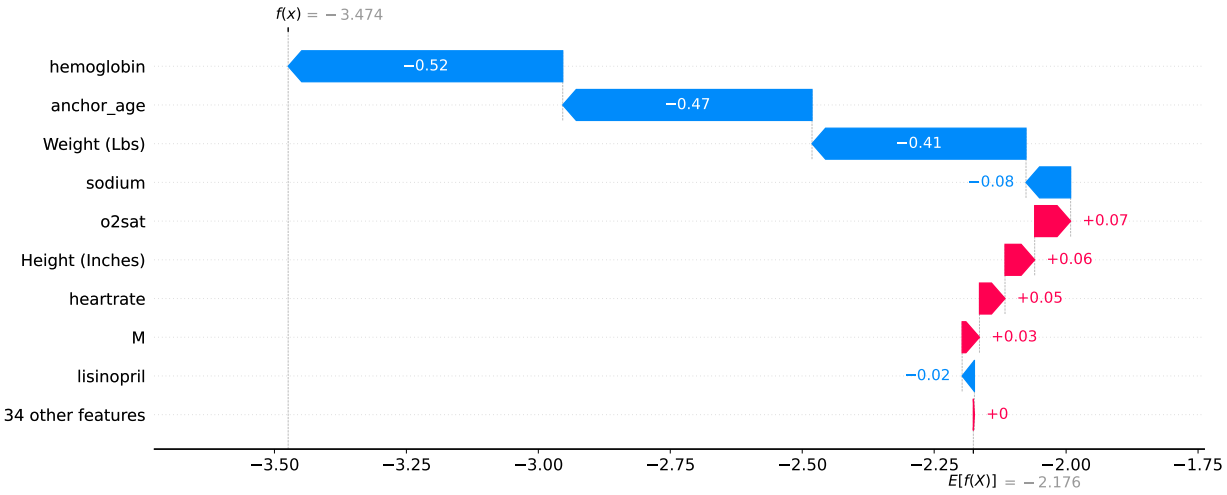

FIG. 7: Deviation from expected output for single subject shown on predictor scale. Passing the values through sigmoid function will convert them to response scale, i.e. discrete hazards

Model	Hyperparameters	C-Index	IBS
Cox Proportional Hazards (CPH)	penalizer = 0.1, step_size = 0.1	0.6953	**0.1731**
Weibull Accelerated Failure Times (AFT)	penalizer = 0.01	0.6953	0.1741
Random Survival Forest	n_estimators=1000, min_samples_split=10, min_samples_leaf=15, n_jobs=-1, random_state=1234, oob_score = True	0.6911 (0.6855, 0.6973)	0.2013 (0.1998, 0.2024)
PyCox (Logistic Hazard loss function [4])	discretization=15, layers=[256,256], batch_norm=True, dropout=0.5, batch_size=256, epochs=500	0.5984 (0.5723, 0.6104)	0.1768 (0.1751, 0.1778)
Deep Survival Machines	k=6, distribution=LogNormal, learning_rate=1e-3, layers=[100, 100]	0.6498 (0.6016, 0.6598)	0.2739 (0.2651, 0.2834)
Time-Invariant Survival	clusters=1, discretizations = 15, hidden_size = 25 , alpha = 0.05 , batch_size=256, num_epochs=400, learning_rate=0.001, patience=20, dropout_rate = 0.5, mc_iter = 100	0.6903 (0.6789, 0.6994)	0.4030 (0.3937, 0.4090)
Time-Variant Survival	discretizations = 15, hidden_size = 25 , alpha = 0.01 , batch_size=256, num_epochs=100, learning_rate=0.001, patience=20, dropout_rate = 0.5, mc_iter = 100	**0.7301** (0.7263 , 0.7352)	0.4039 (0.3981, 0.4072)

TABLE V. Evaluation metrics - Best metrics are written in **bold**. Both metrics are reported as `Mean (lower 90% confidence interval, upper 90% confidence interval)`.

[1] S. L. James, D. Abate, K. H. Abate, S. M. Abay, C. Abbafati, N. Abbasi, and H. Abbastabar, Global, regional, and national incidence, prevalence, and years lived with disability for 354 diseases and injuries for 195 countries and territories, 1990–2017: A systematic analysis for the global burden of disease study 2017., The Lancet https://doi.org/10.1016/S0140-6736(18)32279-7 (2018).

[2] G. Cybenko, Approximation by superpositions of a sigmoidal function, Mathematics of Control, Signals, and Systems **5**, 455–455 (1992).

[3] M. F. Gensheimer and B. Narasimhan, A scalable discrete-time survival model for neural networks, PeerJ **7**, 10.7717/peerj.6257 (2019).

[4] H. Kvamme and Ø. Borgan, Continuous and discrete-time survival prediction with neural networks (2019), arXiv:1910.06724 [cs, stat].

[5] Y. Gal and Z. Ghahramani, Dropout as a bayesian approximation: Representing model uncertainty in deep learning, in *Proceedings of The 33rd International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 48, edited by M. F. Balcan and K. Q. Weinberger (PMLR, New York, New York, USA, 2016) pp. 1050–1059.

[6] S. M. Lundberg and S.-I. Lee, A unified approach to interpreting model predictions (2017).

[7] M. Gjoreski, A. Gradisek, B. Budna, M. Gams, and G. Poglajen, Machine learning and end-to-end deep learning for the detection of chronic heart failure from heart sounds, IEEE Access **8**, 20313 (2020).

[8] J. J. Nirschl, A. Janowczyk, E. G. Peyster, R. Frank, K. B. Margulies, M. D. Feldman, and A. Madabhushi, A deep-learning classifier identifies patients with clinical heart failure using whole-slide images of h&e tissue, PloS one **13**, e0192726 (2018).

[9] L. Wang, L. Sha, J. R. Lakin, J. Bynum, D. W. Bates, P. Hong, and L. Zhou, Development and Validation of a Deep Learning Algorithm for Mortality Prediction in Selecting Patients With Dementia for Earlier Palliative Care Interventions, JAMA Network Open **2**, e196972 (2019).

[10] J. R. Ayala Solares, F. E. Diletta Raimondi, Y. Zhu, F. Rahimian, D. Canoy, J. Tran, A. C. Pinho Gomes, A. H. Payberah, M. Zottoli, M. Nazarzadeh, N. Conrad, K. Rahimi, and G. Salimi-Khorshidi, Deep learning for electronic health records: A comparative review of multiple deep neural architectures, Journal of Biomedical Informatics **101**, 103337 (2020).

[11] G. Lorenzoni, S. S. Sabato, C. Lanera, D. Bottigliengo, C. Minto, H. Ocagli, P. De Paolis, D. Gregori, S. Iliceto, F. Pisanò, and et al., Comparison of machine learning techniques for prediction of hospitalization in heart failure patients, Journal of Clinical Medicine **8**, 1298 (2019).

[12] Z. Che, Y. Cheng, Z. Sun, and Y. Liu, Exploiting convolutional neural network for risk prediction with medical feature embedding, arXiv preprint arXiv:1701.07474 (2017).

[13] Q. Suo, F. Ma, Y. Yuan, M. Huai, W. Zhong, A. Zhang, and J. Gao, Personalized disease prediction using a cnn-based similarity learning method, in *2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)* (IEEE, 2017) pp. 811–816.

[14] L. Brand, A. Patel, I. Singh, and C. Brand, Real time mortality risk prediction: A convolutional neural network approach., in *HEALTHINF* (2018) pp. 463–470.

[15] W. Caicedo-Torres and J. Gutierrez, Iseeu: Visually interpretable deep learning for mortality prediction inside the icu, Journal of biomedical informatics **98**, 103269 (2019).

[16] D. Faraggi and R. Simon, A neural network model for survival data, Statistics in Medicine

https://doi.org/10.1002/sim.4780140108 (1995).
[17] J. Katzman, U. Shaham, J. Bates, A. Cloninger, T. Jiang, and Y. Kluger, Deepsurv: Personalized treatment recommender system using a cox proportional hazards deep neural network., BMC Medical Research Methodology https://doi.org/10.1186/s12874-018-0482-1. (2018).
[18] E. Giunchiglia, A. Nemchenko, and M. van der Schaar, Rnn-surv: A deep recurrent model for survival analysis, Artificial Neural Networks and Machine Learning – ICANN https://doi.org/10.1109/TBME.2019.2909027 (2018).
[19] H. Kvamme, Ø. Borgan, and I. Scheel, Time-to-event prediction with neural networks and cox regression., Journal of Machine Learning Research (2019).
[20] C. Nagpal, X. R. Li, and A. Dubrawski, Deep survival machines: Fully parametric survival regression and representation learning for censored data with competing risks (2021), arXiv:2003.01176 [cs, stat].
[21] C. Lee, J. Yoon, and M. van der Schaar, Dynamic-deephit: A deep learning approach for dynamic survival analysis with competing risks based on longitudinal data, IEEE Transactions on Biomedical Engineering https://doi.org/10.1109/TBME.2019.2909027 (2020).
[22] D. F. Moore, Chapter 2 basic principles of survival analysis, in *Applied Survival Analysis Using R* (Springer, 2016).
[23] D. R. Cox, Regression models and life-tables., Journal of the Royal Statistical Society (1972).
[24] V. C. Raykar, H. Steck, B. Krishnapuram, C. Dehing-oberije, and L. Philippe, On ranking in survival analysis: Bounds on the concordance index, Advances in Neural Information Processing Systems (2007).
[25] L. J. Wei, The accelerated failure time model: a useful alternative to the cox regression model in survival analysis., Statistics in Medicine 10.1002/sim.4780111409 (1992).
[26] W. R. Swindell, Accelerated failure time models provide a useful statistical framework for aging research, Experimental Gerontology https://doi.org/10.1016/j.exger.2008.10.005 (2009).
[27] H. Ishwaran, U. B. Kogalur, E. H. Blackstone, and M. S. Lauer, Random survival forests, The Annals of Applied Statistics https://doi.org/10.1214/08-AOAS169 (2008).
[28] L. Breiman, Random survival, Machine Learning https://doi.org/10.1023/A:1010933404324. (2001).
[29] T. HOTHORN, K. HORNIK, and A. ZEILEIS, Unbiased split variable selection for random survival forests using maximally selected rank statistics, Statistics in Medicine https://doi.org/10.1002/sim.7212. (2006).
[30] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting., Journal of Machine Learning Research (2014).
[31] A. E. Johnson, L. Bulgarelli, L. Shen, A. Gayles, A. Shammout, S. Horng, T. J. Pollard, S. Hao, B. Moody, B. Gow, and et al., Mimic-iv, a freely accessible electronic health record dataset, Scientific Data **10**, 10.1038/s41597-022-01899-x (2023).
[32] H. Uno, T. Cai, M. J. Pencina, R. B. D'Agostino, and L. J. Wei, On the c-statistics for evaluating overall adequacy of risk prediction procedures with censored survival data, Statistics in Medicine **30**, 1105–1117 (2011).
[33] Pysurvival c-index, https://square.github.io /pysurvival/metrics/c_index.html, accessed: 2023-08-01.
[34] J. D. Pinto, A. M. Carvalho, and S. Vinga, Outlier detection in survival analysis based on the concordance c-index, Proceedings of the International Conference on Bioinformatics Models, Methods and Algorithms 10.5220/0005225300750082 (2015).
[35] N. Hartman, S. Kim, K. He, and J. D. Kalbfleisch, Pitfalls of the concordance index for survival outcomes, Statistics in Medicine **42**, 2179–2190 (2023).
[36] P. J. Heagerty and Y. Zheng, Survival model predictive accuracy and roc curves, Biometrics **61**, 92–105 (2005).
[37] E. Graf, C. Schmoor, W. Sauerbrei, and M. Schumacher, Assessment and comparison of prognostic classification schemes for survival data, Statistics in Medicine (1999).
[38] H. W. Lilliefors, On the kolmogorov-smirnov test for normality with mean and variance unknown., Journal of the American Statistical Association **62**, https://doi.org/10.1080/01621459.1967.10482916. (1967).
[39] H. Hung and C.-T. Chiang, Estimation methods for time-dependent auc models with survival data., The Canadian Journal of Statistics / La Revue Canadienne de Statistique **38** (2010).
[40] E. L. Kaplan and P. Meier, Nonparametric estimation from incomplete observations., Journal of the American Statistical Association **53**, https://doi.org/10.2307/2281868. (1958).
[41] R. F., The perceptron: a probabilistic model for information storage and organization in the brain., Psychology Review doi: 10.1037/h0042519. (1958).
[42] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, Learning representations by backpropagating errors., Nature (1986).
[43] S. M. Lundberg, B. Nair, M. S. Vavilala, M. Horibe, M. J. Eisses, T. Adams, and D. E. Liston, Explainable machine-learning predictions for the prevention of hypoxaemia during surgery, Nature Biomedical Engineering https://doi.org/10.1038/s41551-018-0304-0. (2018).
[44] C. C. Brown, On the use of indicator variables for studying the time-dependence of parameters in a response-time model., Biometrics https://doi.org/10.2307/2529811 (1975).

Appendix A: Supplementary Material

Some formal mathematical formulations of the methods used in this paper are elaborated in the following sections.

1. The Kaplan Meier fitter

This estimator, first proposed in [40], is the product over the failure times of the conditional probabilities of surviving to the next failure time. It is given as

$$\hat{S}(t) = \prod_{t_i \leq t} (1 - \hat{q}_i) = \prod_{t_i \leq t} (1 - \frac{d_i}{n_i}) \tag{A1}$$

, where n_i is the number of subjects at risk at time t_i, and d_i is the number of individuals who fail at that time [22]. When plotted, it produces the standard step curve

that comes up in all introductory material to survival analysis.

2. Multi Layer Perceptron

The multi-layer perceptron, arose from the field of psychology [41] formed the foundation of neural networks, although later implementation gradually tore away from the biological foundations and now bear only a slight similarity to the functioning of a human brain. For ordinary regression tasks, an input vector $\mathbf{x}$, weight vector $\mathbf{w}$ and bias vector $\mathbf{b}$,

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \quad \mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

could be used to estimate a target variable $\mathbf{y}$ as

$$\mathbf{y} = \sigma(\mathbf{w}^T\mathbf{x} + \mathbf{b}) \tag{A2}$$

where σ is the sigmoid function for logistic regression and just the identity for linear regression. Now, a perceptron without any hidden layers (as was the original) is effectively similar to ordinary regression. With the addition of hidden layers and non-linear activation functions, multilayer perceptrons become much more powerful. For a network with $\mathbf{L}$ layers, the output of the i^{th} neuron in the l^{th} layer, where $l \in \mathbf{L}$ is given as

$$z_i^l = \mathbf{w}_i^T \mathbf{a}^{l-1} + \mathbf{b}_i \tag{A3}$$

$$\mathbf{a}_i^l = \sigma^l(z_i^l) \tag{A4}$$

where $\mathbf{a}_i^l$ is the activated (having passed through the activation function) output from the i^{th} neuron in the l^{th} layer. This network is trained iteratively by backpropagating [42] errors through the connections over multiple epochs until some reasonable notion of convergence is achieved.

3. SHAP

Explainability of results is sought through SHAP values. The concept of SHAP (SHapley Additive Explanations) comes from coalitional game theory and were recently incorporated into the field of machine learning [43] [6]. The idea is to justly allocate credit to each member of a group (or coalition) for achieving a common goal. An easy example to think of might be to think of a group of friends running up a restaurant bill. At the end, it would be fair for them to split the bill according to what they individually ate. It can be seen that this quite naturally translates to the world of model building, where it is important to know what the contribution of each feature was to the final model output. SHAP provides this unified framework.

4. Loss Function(s)

Negative Log-Likelihood - hazard function
The following derivation is taken from [4]. For notational convenience, let $\kappa(t) \in \{0, \dots, m\}$ define the index of the discrete time t, meaning $t = \tau_{\kappa(t)}$. Thus, the likelihood contribution for individual i is seen to be

$$\begin{aligned} L_i &= f(t_i)^{\delta_i} S(t_i)^{1-\delta_i} \\ &= [h(t_i)S(\tau_{\kappa(t_i)-1})]^{d_i}[(1-h(t_i))S(\tau_{\kappa(t_i)-1})]^{1-d_i} \\ &= h(t_i)^{d_i}[1-h(t_i)]^{1-d_i} S(\tau_{\kappa(t_i)-1})] \\ &= h(t_i)^{d_i}[1-h(t_i)]^{1-d_i} \prod_{j=1}^{\kappa_{t_i-1}} [1-h(\tau_j)] \end{aligned} \tag{A5}$$

For all the individuals, the combined log-likelihood is

$$L = \prod_{i=1}^{n} L_i = \prod_{i=1}^{n} \left(h(t_i|x_i)^{d_i}[1-h(t_i|x_i)]^{1-d_i} \prod_{j=1}^{\kappa_{t_i-1}} [1-h(\tau_j|x_i)] \right) \tag{A6}$$

From here, the loss function for a batch can be constructed as the negative log likelihood.

$$log(L) = \sum_{i=1}^{n} \left(d_i log[h(t_i|x_i)] + (1-d_i) log[1-h(t_i|x_i)] + \sum_{j=1}^{\kappa(t_i)-1} log[1-h(\tau_j|x_i)] \right) \tag{A7}$$

To adjust for varying batch sizes, the mean negative log likelihood is taken as the loss.

$$\begin{aligned} \mathcal{L}_{nll} &= -\frac{1}{n} \sum_{i=1}^{n} \left(d_i log[h(t_i|x_i)] + (1-d_i) log[1-h(t_i|x_i)] + \sum_{\kappa(t_i)-1}^{j=1} log[1-h(\tau_j|x_i)] \right) \\ &= \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{\kappa_{t_i}} (y_{ij} log[h(\tau_j|x_i)] + (1-y_{ij}) log[1-h(\tau_j|x_i)]) \end{aligned} \tag{A8}$$

This can now be minimized by gradient-based methods, thus making it a useful loss function for a neural network to work with. Here, y_{ij} is an indicator variable corresponding to 1 if and only if an event is experienced by the individual i at time t_j. Hence, y is a sparse matrix consisting of mostly 0 with 1 only present when the time t_j represents an observed event $d_i = 1$ for individual x_i.

The loss can be thought of as the negative log likelihood of Bernoulli data $\theta^p(1-\theta)^{1-p}$, where $\theta = h(\tau_j|x_i)$

and $p = y_{ij}$ [44], and can be computed using existing functions in the PyTorch library.

Lower-Bound on C-Index

The following loss function is taken from [24]. The sigmoid function defined as $\sigma = (1 + e^{-z})^{-1}$ is an approximation to the indicator function $\mathbb{1}_{z>0}$. However, it is not a lower bound. For this, the scaled version of its log is taken.

$$\begin{aligned} &\mathbb{1}_{z>0} \geq log\ [2\sigma(z)]/log\ 2 \\ \implies &\mathbb{1}_{z>0} \geq 1 + (log\ \sigma(z)/log\ 2) \end{aligned} \tag{A9}$$

For a more rigourous treatment of this inequality, see the original paper [24]. It follows that the lower bound on the concordance index then becomes

$$c = \frac{1}{|\mathcal{E}|} \sum_{i:C_i=1} \sum_{T_j>T_i} \mathbb{1}_{\eta_j>\eta_i}$$

$$c \geq \frac{1}{|\mathcal{E}|} \sum_{i:C_i=1} \sum_{T_j>T_i} 1 + (log\ \sigma(\eta_j - \eta_i)/log\ 2) \tag{A10}$$

The negative of this lower-bound is chosen to be the loss function, which can therefore be minimized using gradient-based methods (again, with the help of a library like PyTorch).

$$\mathcal{L}_{lbo} = -\frac{1}{|\mathcal{E}|} \sum_{i:C_i=1} \sum_{T_j>T_i} 1 + (log\ \sigma(\eta_j - \eta_i)/log\ 2) \tag{A11}$$

These two loss functions are linearly combined based on a hyperparameter $\alpha \in [0, 1]$ to make the total loss

$$\mathcal{L}_{total} = \alpha\mathcal{L}_{nll} + (1 - \alpha)\mathcal{L}_{lbo} \tag{A12}$$