Machine Learning with Applications 16 (2024) 100535

Contents lists available at ScienceDirect

Machine Learning with Applications

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

Case-Base Neural Network: Survival analysis with time-varying, higher-order interactions

Jesse Islam [a,*], Maxime Turgeon [b], Robert Sladek [a,c], Sahir Bhatnagar [d]

[a] McGill University Department of Quantitative Life Sciences, 805 rue Sherbrooke O, Montréal, H3A 0B9, Quebec, Canada
[b] University of Manitoba Department of Statistics, 50 Sifton Rd, Winnipeg, R3T2N2, Manitoba, Canada
[c] McGill University Department of Human Genetics, 805 rue Sherbrooke O, Montréal, H3A 0B9, Quebec, Canada
[d] McGill University Department of Biostatistics, 805 rue Sherbrooke O, Montréal, H3A 0B9, Quebec, Canada

ARTICLE INFO

Dataset link: https://github.com/Jesse-Islam/cbnnManuscript, https://github.com/Jesse-Islam/cbnn

Keywords:
Survival analysis
Machine learning
Case-base
Neural network

ABSTRACT

In the context of survival analysis, data-driven neural network-based methods have been developed to model complex covariate effects. While these methods may provide better predictive performance than regression-based approaches, not all can model time-varying interactions and complex baseline hazards. To address this, we propose Case-Base Neural Networks (CBNNs) as a new approach that combines the case-base sampling framework with flexible neural network architectures. Using a novel sampling scheme and data augmentation to naturally account for censoring, we construct a feed-forward neural network that includes time as an input. CBNNs predict the probability of an event occurring at a given moment to estimate the full hazard function. We compare the performance of CBNNs to regression and neural network-based survival methods in a simulation and three case studies using two time-dependent metrics. First, we examine performance on a simulation involving a complex baseline hazard and time-varying interactions to assess all methods, with CBNN outperforming competitors. Then, we apply all methods to three real data applications, with CBNNs outperforming the competing models in two studies and showing similar performance in the third. Our results highlight the benefit of combining case-base sampling with deep learning to provide a simple and flexible framework for data-driven modeling of single event survival outcomes that estimates time-varying effects and a complex baseline hazard by design. An R package is available at https://github.com/Jesse-Islam/cbnn.

1. Introduction

A common assumption in survival analysis is that the risk ratio of the event of interest does not vary with time which has led to the dominance of proportional hazard models (Cox, 1972; Hanley & Miettinen, 2009). This simplifying assumption explains the popularity of Cox proportional hazards models over smooth-in-time, accelerated failure time (AFT) models and results in analyses based on hazard ratios and relative risks rather than on survival curves and absolute risks (Hanley & Miettinen, 2009). The proportional hazards assumption in a Cox model may be incorrect in studies where the disease pathogenesis may change over time. For example, age, chronic obstructive pulmonary disease, diabetes mellitus, existing heart disease, prior heart surgery, intra-aortic balloon pump, urgent and emergent surgical priority are all preoperative factors that have demonstrated time-varying associations with long-term mortality following coronary artery bypass graft surgery (Gao et al., 2006). A time-varying effect of body mass index has been associated with all-cause mortality in hypertensive patients (Zhu et al., 2022) and the effect of long-term statin use on type 2 diabetes onset varies with time (Na, Cho, Kim, Choi, & Han, 2020). The time-varying effect of tumor size has been associated with the risk of cancer recurrence (Coradini et al., 2000). Time-varying interactions were identified in appraisals of 28 of 40 targeted and immuno-oncology therapies reported by the National Institute for Health and Care Excellence, highlighting the importance of accounting for time-varying features in cancer treatment studies (Salmon & Melendez-Torres, 2023). These time-varying covariate effects can be incorporated easily into AFT models; however, this requires prior knowledge of potential time-varying interactions and their quantitative effects (Royston & Parmar,

The code (and data) in this article has been certified as Reproducible by Code Ocean: (https://codeocean.com/). More information on the Reproducibility Badge Initiative is available at https://www.elsevier.com/physical-sciences-and-engineering/computer-science/journals.

* Corresponding author.
E-mail addresses: jesse.islam@mail.mcgill.ca (J. Islam), turgeon.maxime@gmail.com (M. Turgeon), rob.sladek@mcgill.ca (R. Sladek), sahir.bhatnagar@5primesciences.com (S. Bhatnagar).

https://doi.org/10.1016/j.mlwa.2024.100535
Received 4 July 2023; Received in revised form 15 February 2024; Accepted 17 February 2024
Available online 20 February 2024

2002). In this study, we develop a method that accounts for these time-varying effects without user specification, improving risk prediction models.

We propose Case-Base Neural Networks (CBNNs) as a data-driven method for single event survival analysis. CBNNs use the case-base sampling technique, which allows probabilistic models to predict survival outcomes by adjusting for a sampling-specific bias with an offset term (Hanley & Miettinen, 2009). After case-base sampling, we implement a model using common neural network components that use time as a feature to estimate time-varying interactions and a flexible baseline hazard.

In this paper, we first review state-of-the-art deep learning methods for survival analysis to position CBNN relative to related works. Second, we describe the case-base sampling procedure and compare its properties to neural network models, along with our hyperparameter selection criteria, metrics, and software implementation. Third, we compare the performance of CBNN to neural network and regression-based survival methods on simulated data and describe their performance in three case studies. Finally, we explore the implications of our results and contextualize them within neural network survival analysis in a single event setting.

2. Related works

Deep learning approaches for survival analysis can be grouped into four broad categories: models that ignore censoring; Cox partial log-likelihood; discrete survival time; and fully parametric. We wish to distinguish models, where the innovation is provided by the neural network architecture, from frameworks, which alter how the survival outcome is interpreted. Based on this distinction, any framework can use any neural network architecture.

2.1. Survival models that ignore censoring

Survival analysis methods handle censored data, where the event time of interest may not be known. Several methods have been developed to ignore these censored individuals during the fitting process, simplifying the survival problem. For example, Zadeh Shirazi, Fornaciari, Bagherian, Ebert, Koszyca, and Gomez (2020) model survival data in a deep learning context, where the model predicts survival rate categories. In contrast, two other studies predict the probability of a bad prognosis by integrating imaging and gene expression data, using either a bilinear network (Wang, Li, Wang, & Li, 2021) or an attention layer with transfer learning for the imaging features and ensemble forests for gene expression (Jia et al., 2023). Though these methods predict a simple binary outcome, they are unable to estimate the hazard function. These architectures can be inserted into the CBNN framework which provides the added flexibility of estimating complex baseline hazards and time-varying effects. Rather than converting time-to-event outcomes into a risk-of-prognosis score, CBNNs can model the full hazard function.

2.2. Cox partial log-likelihood

Several architectures using the Cox partial log-likelihood have been proposed for conventional tabular datasets. DeepSurv is a neural network-based proportional hazards model (Katzman et al., 2018), which has been used to develop prediction models for several clinical scenarios (Bice et al., 2020; Kim et al., 2019; She et al., 2020; Yu, Huang, Feng, & Lyu, 2022). Ching, Zhu, and Garmire (2018) use gene expression data as input to a single hidden layer neural network to model cancer survival; an approach which is outperformed by an autoencoder with Cox regression model (Yin, Chen, Wu, & Wei, 2022) and a convolutional model with residual networks (Huang et al., 2020). In parallel, Hao, Kim, Kwon, and Ha (2021) assess simple Cox neural networks in a high dimensional setting against random forest and penalized regression approaches. As a more complex model for gene expression, Meng et al. (2022) propose to first fit a generative adversarial network (GAN) to the expression data before transferring the weights to a Cox neural network. These methods provide data-agnostic architectures in clinical settings that do not take advantage of known relationships between features.

Other approaches use the prior knowledge of existing structure between features to improve predictive performance. For example, Chen et al. (2023) transfer the embeddings of pre-trained convolutional layers on histopathological images to an autoencoder, the central layer of which is passed into a linear Cox regression to predict cervical cancer survival. Histopathological images were also used to predict lung cancer survival, using either convolutional layers (Zhu, Yao, Zhu, & Huang, 2017) or a convolutional transformer and Siamese network model with a modified Cox partial log-likelihood loss function that accounts for aleatoric uncertainty (Tang et al., 2023). Kaynar et al. (2023) propose a pathway-informed model integrating metabolomic data and known metabolic pathways to predict survival in patients with glioma. To assess cardiovascular risk, Barbieri et al. (2022) propose a sex-specific neural network model using bidirectional gated recurrent units to represent recent clinical history. These architectures focus on structure within unimodal datasets and extend the concepts proposed by data-agnostic architectures.

Other approaches integrate prior knowledge across multiple modalities to improve predictive performance. Hao, Jing, and Sun (2022) use sample similarity and correlations across three genomic data modalities, then integrate them in a graph neural network to model cancer outcomes. To integrate genomic and imaging data, Mobadersany et al. (2018) make use of convolutional layers and Li, Wu, Li, and Wang (2022) propose a factorized bilinear network as an improvement over an unfactorized network. Finally, Chen et al. (2020) integrate spatial transcriptomics as well, using both convolutional layers and graphical neural networks. Whether data-agnostic, data-specific or multi-modal, all these models make use of the Cox partial log-likelihood and assume time-invariant effects.

2.3. Discrete-time

Discrete-time survival models make survival predictions for specified intervals of follow-up time. For example, DeepHit directly estimates survival probabilities and assumes an inverse Gaussian distribution as the baseline hazard (Lee, Zame, Yoon, & Schaar, 2018). The same goal is accomplished by Gensheimer and Narasimhan (2019) using basic hidden layers and Giunchiglia, Nemchenko, and van der Schaar (2018) using a recurrent neural network architecture. To estimate a piecewise hazard, Kopper, Wiegrebe, Bischl, Bender, and Rügamer (2022) convert survival data into a piecewise exponential data format and propose a computationally efficient neural network. To predict cancer survival, Wulczyn et al. (2020) incorporate prior knowledge of image motifs using convolutional layers. Discrete-time neural network models have also been used with multi-modal data. To predict long term cancer survival, Vale-Silva and Rohr (2021) propose a model integrating images along with transcriptomic, genomic and epigenomic data. All these methods provide a discrete estimate of survival outcomes dependent on the follow-up times of interest.

2.4. Fully parametric framework

Fully parametric methods provide an estimate of the full hazard function. For example, Deep Survival Machines (DSM), uses neural networks with a mixture of distributions to fit a flexible baseline hazard (Nagpal, Li, & Dubrawski, 2021). This method extends to settings where the covariates are measured multiple times (i.e. time-series) (Nagpal, Jeanselme, & Dubrawski, 2021). However, the initial reports did not assess performance when time-varying effects of baseline measurements are present (Nagpal, Jeanselme, & Dubrawski, 2021; Nagpal et al., 2021).

clinical datasets often show significant time-varying effects that are relevant to disease-specific outcomes (Coradini et al., 2000; Gao et al., 2006; Na et al., 2020; Salmon & Melendez-Torres, 2023; Zhu et al., 2022). Not limited to these diseases, we expect all models that predict a survival outcome may gain improved performance by accounting for non-proportional hazards. Cox partial log-likelihood based models assume the effects are time invariant (Cox, 1972). CBNNs accounts for the time-varying effects by design. For all the discrete-time frameworks and architectures, a selection of smaller intervals of time or more survival times of interest may result in a few or no events per interval, which potentially leads to instability in the optimization function (Bhatnagar*, Turgeon*, Islam, Hanley, & Saarela, 2022). In contrast, selecting larger intervals may mask nonlinear, time-varying effects in the hazard function (Bhatnagar* et al., 2022). CBNNs can model time-varying interactions and a complex baseline hazard function without the trade-off between long or short interval lengths. CBNNs also estimate a full hazard function for all of follow-up time, rather than discrete intervals of follow-up-time. Compared to fully parametric approaches like DSM, CBNN explicitly incorporates time into the model, providing a simple way to account for time-varying effects that improve survival prediction.

3. Case-base neural network, metrics, hyperparameters and software

Case-base sampling is an alternate framework for survival analysis (Hanley & Miettinen, 2009), which converts the total survival time into discrete person-time coordinates (person-moments). In this section, we detail how CBNNs explicitly incorporate time as a feature while adjusting for the sampling bias in case-base sampled data; describe the metrics; and describe the hyperparameter selection procedure used to compare CBNN with other methods. An R package to use CBNN is available at https://github.com/Jesse-Islam/cbnn. The entire code base to reproduce the figures and empirical results in this paper is available at https://github.com/Jesse-Islam/cbnnManuscript.

3.1. Case-base sampling

To implement case-base sampling, we divide the total survival time for each individual into discrete person-moments and treat each person-moment as a sample. This creates a *base series* of *person-moments* where an event does not occur. This *base series* complements the *case series*, which contains all person-moments at which the event of interest occurs.

For each person-moment sampled, let X_i be the corresponding covariate profile $(x_{i1}, x_{i2}, \dots, x_{ip})$, T_i be the time of the person-moment and Y_i be the indicator variable for whether the event of interest occurred at time T_i. We estimate the hazard function $h(t \mid X_i)$ using the sampled person-moments. Recall that $h(t \mid X_i)$ is the instantaneous risk of experiencing the event at time t for a given set of covariates X_i, assuming $T_i \geq t$.

Now, let b be the (user-defined) size of the *base series* and let B be the sum of all follow-up times for the individuals in the study. Let c be the number of events in the *case series*. A reasonable concern is determining how large b should be relative to c, since the size of b influences how much information is lost in the sampling process (Hanley & Miettinen, 2009). The relative information when comparing two averages is measured by $\frac{cb}{c+b} = \left[\left(\frac{1}{c} + \frac{1}{b}\right)\right]$, where b is the sample size of the *base series* and c is the sample size of the *case series* (Hanley & Miettinen, 2009; Mantel, 1973). If $b = 100c$, then we expect learned weights to be at most one percent higher than if the entire study base B was used, as they are proportional to $\frac{1}{c} + \frac{1}{100c}$ rather than $\frac{1}{c} + \frac{1}{\infty c}$ (Hanley & Miettinen, 2009; Mantel, 1973).

If we sample the *base series* uniformly across the study base, then the hazard function of the sampling process is equal to b/B. Therefore, we have the following equality (Saarela & Hanley, 2015) [For a rigorous treatment, see Saarela & Hanley (2015) section 3]:

$$\frac{P\left(Y_i = 1 \mid X_i, T_i\right)}{P\left(Y_i = 0 \mid X_i, T_i\right)} = \frac{h\left(T_i \mid X_i\right)}{b/B}. \tag{1}$$

The odds of a person-moment being in the *case series* is the ratio of the hazard $h(T_i \mid X_i)$ and the uniform rate b/B. Using (1), we can see how the log-hazard function can be estimated from the log-odds arising from case-base sampling:

$$\log\left(h\left(t \mid X_i\right)\right) = \log\left(\frac{P\left(Y_i = 1 \mid X_i, t\right)}{P\left(Y_i = 0 \mid X_i, t\right)}\right) + \log\left(\frac{b}{B}\right). \tag{2}$$

To estimate the correct hazard function, we adjust for the bias introduced when sampling a fraction of the study base ($\frac{b}{B}$) by adding the term $\log\left(\frac{B}{b}\right)$ to offset $\log\left(\frac{b}{B}\right)$ during the fitting process. See Algorithm 1 for a pseudocode implementation of the case-base sampling procedure. For a discussion of the connection between our objective function and the full data likelihood, see Saarela (2016).

Algorithm 1: Pseudocode algorithm of the case-base sampling procedure.

```
function sampleCaseBase(X);
Input  : X = matrix of covariate profiles.
         X_T = Vector of event times for each profile.
         X_Y = Vector of event indicators for each profile.
Output: Case-base sampled data matrix.
n = length(X_T) ;                    /* Number of subjects */
B = Sum(X_T) ;    /* Total person-time in study base */
c = Sum(X_Y) ;                       /* Number of cases */
b = 100 * c ;                  /* Number of base samples */
for i = 1, ..., n do
    Set p_i = X_T[i]/B;
end
while j < b do
    Sample ℓ from {1, ..., n} with probabilities p_1, ..., p_n
    Set baseSeries[j, ] = X[ℓ, ]
    Sample baseSeries_T[j] from Uniform(0, X_T[ℓ])
end
Set caseSeries = X[X_Y == 1, ].
return(concatenate(baseSeries,caseSeries))
```

3.2. Neural networks to model the hazard function

After case-base sampling, we pass all features, including time, into any user-defined feed-forward component (Fig. 1). Then, we add an offset term and pass the output through a sigmoid activation function (Fig. 1). Since we are interested in predicting the odds of an event occurring, the sigmoid activation function is ideal as it is the inverse of the odds and can be used to calculate the hazard. The general form for any feed-forward neural network architecture using CBNN is:

$$P\left(Y = 1 | X, T\right) = \text{sigmoid}\left(f_\theta(X, T) + \log\left(\frac{B}{b}\right)\right),$$

where T is a random variable representing the event time, X is the random variable for a covariate profile, $f_\theta(X, T)$ represents any feed-forward neural network architecture, $\log\left(\frac{B}{b}\right)$ is the offset term to adjust for the bias $\left(\log\left(\frac{b}{B}\right)\right)$ set by case-base sampling, θ is the set of parameters learned by the neural network and $\text{sigmoid}(x) = \frac{1}{1+e^{-x}}$. By approximating a higher-order polynomial of time using a neural network, the baseline hazard specification is now data-driven, while user-defined hyperparameters such as regularization, number of layers and nodes control the flexibility of the hazard function.

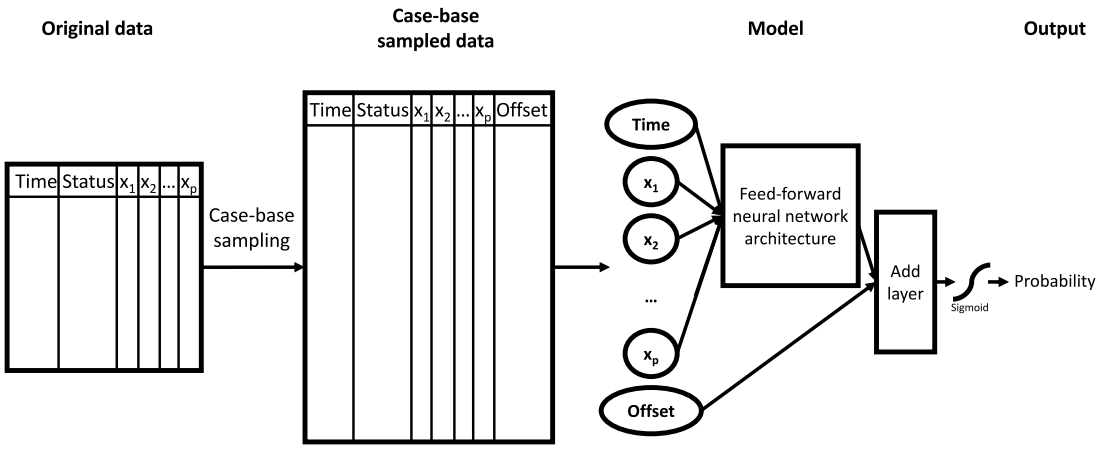

Fig. 1. Methodological steps involved in CBNN. The first step, case-base sampling, is completed before training begins. Then, we pass this sampled data through a feed-forward neural network, add an offset to adjust for the bias inherent in case-base sampling and apply a sigmoid activation function to estimate a probability. Once the neural network model completes its training, we can convert the probability to a hazard for the survival outcome of interest.

The following derivation shows how our probability estimate is converted to odds:

$$\begin{aligned}\log\left(h(t \mid X)\right) &= \log\left(\frac{\text{sigmoid}\left(f_\theta(X,T)+\log\left(\frac{B}{b}\right)\right)}{1-\text{sigmoid}\left(f_\theta(X,T)+\log\left(\frac{B}{b}\right)\right)}\right)+\log\left(\frac{b}{B}\right)\\ &= \log\left(\frac{\frac{\exp\left(f_\theta(X,T)+\log\left(\frac{B}{b}\right)\right)}{\exp\left(f_\theta(X,T)+\log\left(\frac{B}{b}\right)\right)+1}}{1-\frac{\exp\left(f_\theta(X,T)+\log\left(\frac{B}{b}\right)\right)}{\exp\left(f_\theta(X,T)+\log\left(\frac{B}{b}\right)\right)+1}}\right)+\log\left(\frac{b}{B}\right)\\ &= \log\left(\exp\left(f_\theta(X,T)+\log\left(\frac{B}{b}\right)\right)\right)+\log\left(\frac{b}{B}\right)\\ &= f_\theta(X,T)+\log\left(\frac{B}{b}\right)+\log\left(\frac{b}{B}\right)\\ &= f_\theta(X,T).\end{aligned}$$

We use binary cross-entropy as our loss function (Gulli & Pal, 2017), from which we calculate:

$$L(\theta) = -\frac{1}{N}\left(\sum_{i=1}^{N} y_i \cdot \log(\hat{f}_\theta(x_i,t_i)) + (1-y_i)\cdot\log(1-\hat{f}_\theta(x_i,t_i))\right),$$

where $\hat{f}_\theta(x_i,t_i)$ is our estimate for a given covariate profile and time, y_i is our target value specifying whether an event occurred, and N represents the number of individuals in our training set.

Backpropagation with an appropriate minimization algorithm (such as Adam, RMSPropagation, stochastic gradient descent) is used to optimize the parameters in the model (Gulli & Pal, 2017). For our analysis, we use Adam (Kingma & Ba, 2014). While the size of the *case series* is fixed as the number of events; the size of the *base series* is not restricted. We use a ratio of 100:1 *base series* to *case series* (Hanley & Miettinen, 2009). After fitting our model, we convert the output to a hazard. To use CBNN for predictions, we manually set the offset term $\left(\log\left(\frac{B}{b}\right)\right)$ to 0 in the new data as we already account for the sampling bias during the fitting process.

Since we are directly modeling the hazard, we can readily estimate the risk function (F) at time t for a covariate profile X,

$$F(t \mid X) = 1-\exp\left(-\int_0^t h(u|X)\,\mathrm{d}u\right). \tag{3}$$

We use a finite Riemann sum (Hughes-Hallett, Gleason, & McCallum, 2020) to approximate the integral in (3).

3.3. Performance metrics

To choose our method-specific hyperparameters, we use the Integrated Brier Score (IBS) (Graf, Schmoor, Sauerbrei, & Schumacher, 1999), which is based on the Brier Score (BS) and provides a summarized assessment of performance for each model. We assess the performance of models on a held-out dataset using two metrics: (1) the Index of Prediction Accuracy (IPA) (Kattan & Gerds, 2018); and (2) the Inverse probability censoring weights-adjusted time-dependent area under the receiver operating characteristic curve (AUC_{IPCW}) (Blanche et al., 2015). The IPA score is a metric for both discrimination and calibration, while the AUC_{IPCW} provides a metric for discrimination only.

3.3.1. Brier score (BS)

The BS (Graf et al., 1999) is defined as

$$BS(t) = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{\left(1-\hat{F}(t|X_i)\right)^2\cdot I(T_i\le t,\delta_i=1)}{\hat{G}(T_i)} + \frac{\hat{F}(t|X_i)^2\cdot I(T_i>t)}{\hat{G}(t)}\right), \tag{4}$$

where $\delta_i = 1$ shows individuals who have experienced the event, N represents the number of samples in our dataset over which we calculate $BS(t)$, T_i is the survival or censoring time of an individual, $\hat{G}(t) = P[c > t]$ is a non-parametric estimate of the censoring distribution and c is censoring time. The BS provides a score that accounts for the information loss because of censoring. Once we fix our t of interest, the individuals in the dataset can be divided into three groups. Individuals who experienced the event before t are present in the first term of the equation. The second term of the equation includes individuals who experience the event or are censored after t. Those censored before t (the remaining individuals) are accounted for by the IPCW adjustment ($G(\cdot)$) present on both terms.

3.3.2. Integrated brier score (IBS)

The Integrated Brier Score (IBS) is a function of the BS (4) (Graf et al., 1999), which is defined as

$$IBS(t) = \int_t^{t_{max}} BS(t)\,dw(t),$$

where $w(t) = \frac{t}{t_{max}}$ and t_{max} is the upper bound for the survival times of interest. As the IBS is calculated for a range of follow-up times, it is a useful metric to track overall performance across all of follow-up time during cross-validation. We use the IPA score to get an estimate of performance for a range of follow-up times in our studies.

3.3.3. Index of prediction accuracy (IPA)

The IPA is a function of time, based on the BS. The IPA score is given by

$$\text{IPA}(t) = 1-\frac{BS_{model}(t)}{BS_{null}(t)},$$

where $BS_{model}(t)$ represents the BS over time (t) for the model of interest and $(S_{null})(t)$ represents the BS if we use an unadjusted Kaplan–Meier (KM) curve as the prediction for all observations (Kattan & Gerds, 2018). The IPA score has an upper bound of one, where positive values show an increase in performance over the null model and negative values show that the null model performs better. As an extension of BS, the IPA score assesses both model calibration and discrimination and shows how performance changes over follow-up time (Graf et al., 1999; Kattan & Gerds, 2018).

3.3.4. Inverse probability censoring weights-adjusted time-dependent area under the receiver operating characteristic curve (AUC_{IPCW})

The IPCW-adjusted AUC (AUC_{IPCW}) is a time-dependent metric that considers censoring (Blanche et al., 2015). For a given follow-up time of interest,

$$AUC_{IPCW}(t) = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} I_{\hat{F}(t|X_i)>\hat{F}(t|X_j)} \cdot I_{T_i\le t,\delta_i=1} \cdot \left(1 - I_{T_j\le t,\delta_j=1}\right) \cdot W_i(t) \cdot W_j(t)}{\sum_{i=1}^{n}\sum_{j=1}^{n} I_{T_i\le t,\delta_i=1} \cdot \left(1 - I_{T_j\le t,\delta_j=1}\right) \cdot W_i(t) \cdot W_j(t)},$$

where

$$W_i(t) = \frac{I_{T_i\le t,\delta_i=1}}{\hat{G}(T_i)} + \frac{I_{T_i>t}}{\hat{G}(t)}.$$

The AUC_{IPCW} measures discrimination (Blanche et al., 2015). By examining both IPA and AUC_{IPCW}, we can better understand whether a model performs better in terms of calibration or discrimination.

3.4. Hyperparameter selection

Neural networks require external parameters that constrain the learning process (hyperparameters). We apply the following hyperparameter optimization procedure to each method for each study. First, we fix a test set with 15% of the data which we keep aside during hyperparameter selection. To determine the best hyperparameters for each neural network method, we use a three-fold cross-validated grid search on the remaining data (85% training, 15% validation) for the following range of hyperparameters:

$$\text{Search space}: \begin{cases} \text{Learning rate} \sim \{0.001, 0.01\} \\ \text{Dropout} \sim \{0.01, 0.05, 0.1\} \\ \text{First layer nodes} \sim \{50, 75, 100\} \\ \text{Second layer nodes} \sim \{10, 25, 50\} \\ \text{Number of batches} \sim \{100, 500\} \\ \text{Activation function} \sim \{\text{ReLU, Linear}\} \\ \alpha \sim \{0, 0.5, 1\} \text{ (DeepHit only)}. \end{cases}$$

For all methods, we use the training set to estimate the mean and standard deviation and scale each covariate of interest other than time, which is divided by the training set maximum. For CBNN, we perform case-base sampling independently for the training and validation sets, then use the estimated training offset as our validation offset. The linear models combine the training set and validation set for training. DeepHit uses α as a hyperparameter, while DeepSurv and CBNN do not. We track IBS on the validation set for each hyperparameter combination, choosing the combination with the lowest score for each method (Table 1).

3.5. Software implementation

R (R. Core Team, 2021) and Python (Van Rossum & Drake, 2009) are used to evaluate methods from both languages. We fit Cox models using the **survival** package (Therneau & Grambsch, 2000) and CBLR models using the **casebase** package (Bhatnagar, Turgeon, Islam, Hanley, & Saarela, 2020). Both DeepSurv and DeepHit are fit using **pyCox** (Lee et al., 2018). We made the components of CBNN using the **casebase** package (Bhatnagar et al., 2020) for the sampling step and the **keras** package (Allaire & Chollet, 2021) for our neural network architecture. We use the **simsurv** package (Brilleman, Wolfe, Moreno-Betancur, & Crowther, 2020) for our simulation studies and **flexsurv** (Jackson, 2016) to fit a flexible baseline hazard using splines for our complex simulation. The **riskRegression** package (Gerds & Kattan, 2021) is used to get the IPA and AUC_{IPCW}. We modify the **riskRegression** package to be used with any user supplied risk function F. We use the **reticulate** package (Ushey, Allaire, & Tang, 2021) to run both R and Python based methods on the same seed.

4. Simulation study

We simulate data to evaluate the performance of CBNN in comparison with existing regression (Cox, CBLR) and neural network (DeepHit, DeepSurv) methods. We specify a linear combination of each covariate as the linear predictor in the regression-based methods (Cox, CBLR), which contrasts with neural network approaches that allow for approximations of non-linear interactions. We simulate data based on a complex baseline hazard with time-varying interactions and 10% random censoring. We simulate three covariates for 5000 individuals:

$$z_1 \sim \text{Bernoulli}(0.5) \qquad z_2 \sim \begin{cases} N(0, 0.5) & \text{if } z_1 = 0 \\ N(1, 0.5) & \text{if } z_1 = 1 \end{cases} \qquad z_3 \sim N(1, 0.5).$$

In addition to the methods described above, we include the exact functional form of the covariates in a CBLR model (referred to as Optimal for simplicity) in the complex simulation. We obtain confidence intervals by conducting 100 bootstrap re-samples on the training data. We keep 15% of the data for testing before hyperparameter selection. We use 15% of the remaining data for validation and the rest is reserved for training. We do not perform case-base sampling on the testing set and we set the offset term to 0 during prediction because the bias from case-base sampling is appropriately adjusted for during the fitting process. We predict risk functions for individuals in the test set, which are used to calculate our IPA and AUC_{IPCW}.

4.1. Complex simulation: flexible baseline hazard, time-varying interactions

We use this simulation to assess performance on data with a complex baseline hazard and a time-varying interaction. We design the model

$$\log h(t \mid X_i) = \sum_{i=1}^{5}(\gamma_i \cdot \psi_i) + \beta_1(z_1) + \beta_2(z_2) + \beta_3(z_3) + \tau_1(z_1 \cdot t) + \tau_2(z_2 \cdot z_3),$$

where $\gamma_1 = 3.9, \gamma_2 = 3, \gamma_3 = -0.43, \gamma_4 = 1.33, \gamma_5 = -0.86, \beta_1 = -5, \beta_2 = -1, \beta_3 = 1, \tau_1 = 0.001, \tau_2 = -1$ and ψ_i are basis splines. The γ coefficients are obtained from an intercept-only cubic splines model with three knots using the *flexsurvspline* function from the **flexsurv** package (Jackson, 2016) on the German Breast Cancer Study Group dataset. These coefficients provide parameters for a complex baseline hazard from which we can simulate. The study comprised of 686 women with breast cancer followed between 1984 and 1989 (Royston & Parmar, 2002). These γ, β and τ coefficients are used as our baseline hazard parameters and are fixed for the analysis. The β coefficients represent direct effects, τ_2 represents an interaction and τ_1 is a time-varying interaction.

4.2. Performance comparison in complex simulation

Fig. 2 A, E and Table 2 shows the performance over time on a test set. The Optimal model acts as a reference for ideal performance on the simulated data. For discrimination, the Optimal model performs best, followed by CBNN, DeepHit, DeepSurv and the linear models (Fig. 2 E). To obtain a more realistic performance assessment, we compared models in three case studies with a time-to-event outcome.

Table 1
Hyperparameters selected after three-fold cross-validated grid search along with the average IBS for each neural network model in the complex simulation (A), multiple myeloma (MM) case study (B), free light chain (FLC) case study (C) and prostate cancer (Prostate) case study (D).

A: Complex

Hyperparameter	CBNN	DeepSurv	DeepHit
Learning rate	0.001	0.001	0.001
Dropout	0.01	0.01	0.05
First layer nodes	75	50	50
Second layer nodes	50	50	50
Number of batches	100	500	100
Activation function	ReLU	ReLU	Linear
α	-	-	1
IBS	0.06744259	0.07529775	0.08821864

B: MM

Hyperparameter	CBNN	DeepSurv	DeepHit
Learning rate	0.001	0.01	0.01
Dropout	0.1	0.05	0.01
First layer nodes	50	100	100
Second layer nodes	50	25	25
Number of batches	500	100	100
Activation function	ReLU	ReLU	ReLU
α	-	-	0
IBS	0.09414844	0.09912649	0.1097949

C: FLC

Hyperparameter	CBNN	DeepSurv	DeepHit
Learning rate	0.001	0.001	0.01
Dropout	0.05	0.05	0.1
First layer nodes	50	100	50
Second layer nodes	10	10	10
Number of batches	100	100	100
Activation function	ReLU	ReLU	ReLU
α	-	-	1
IBS	0.09903534	0.09900328	0.09979871

D: Prostate

Hyperparameter	CBNN	DeepSurv	DeepHit
Learning rate	0.001	0.01	0.01
Dropout	0.01	0.1	0.05
First layer nodes	100	75	75
Second layer nodes	25	25	25
Number of batches	100	100	500
Activation function	ReLU	ReLU	Linear
α	-	-	1
IBS	0.07618916	0.07631809	0.07634624

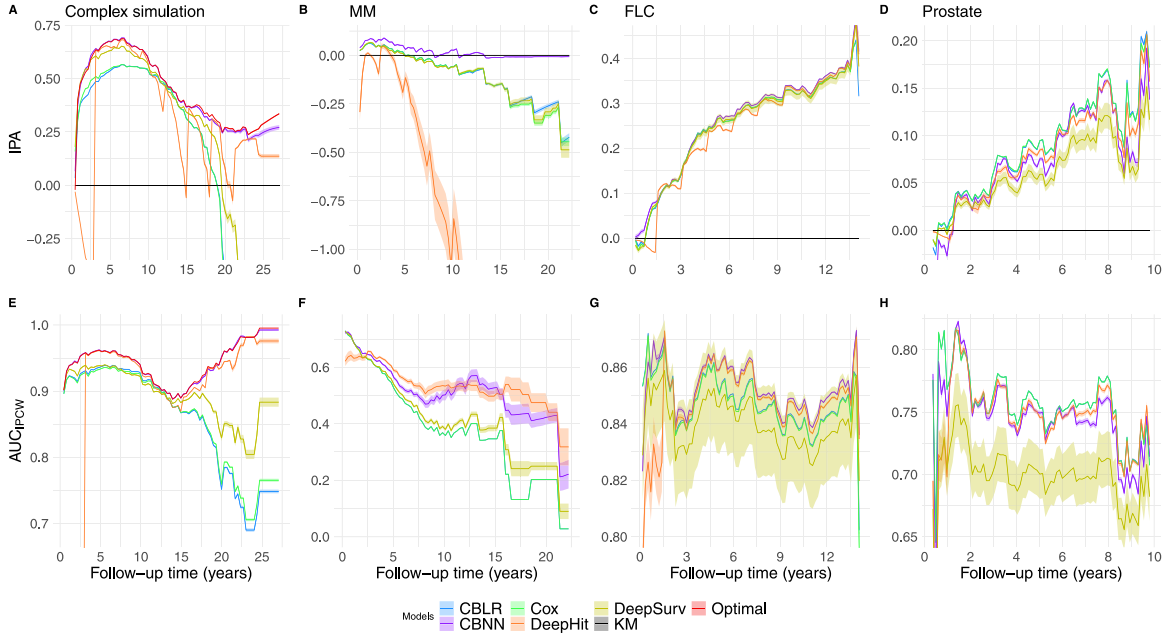

Fig. 2. Performance of each model in the complex simulation (A, E), multiple myeloma (MM) case study (B, F), free light chain (FLC) case study (C, G) and prostate cancer (Prostate) case study (D, H). The first row shows the IPA for each model in each study over follow-up time. Negative values mean the model performs worse than the null model and positive values mean the model performs better. The second row shows the AUC_{IPCW} for each model in each study over follow-up time, where higher is better. Each model-specific metric in each study shows a 95% confidence interval over 100 iterations. Metrics are shown for six models: Case-Base with Logistic Regression (CBLR), Case-Base Neural Network (CBNN), Cox Proportional Hazard (Cox), DeepHit and DeepSurv. The Kaplan–Meier (KM) model serves as a baseline, predicting the average curve for all individuals. CBLR and Cox have near identical performance, resulting in curves that overlap. The Optimal model (a CBLR model with the exact interaction terms and baseline hazard specified) shows the best performance we can expect on the simulated data.

5. Case studies

The simulation examines whether a complex baseline hazard and time-varying interactions affect method performance. The case studies assess performance in more realistic conditions, where we may not know if a flexible baseline hazard or time-varying interactions are beneficial for prediction. For each dataset, we perform a grid search with the same hyperparameter procedure as described in the simulation (three-fold cross-validated grid search). We then predict risk functions for everyone in the test set with the selected hyperparameters, which is used to calculate our metrics. We conduct 100-fold bootstrap re-samples on the training data to get confidence intervals.

5.1. Performance evaluation on multiple myeloma dataset

We expect cancer may have risk factors that vary with time (Coradini et al., 2000; Salmon & Melendez-Torres, 2023). As such, the first case study examines multiple myeloma (MM), using a cohort of 3882 patients seen at the Mayo Clinic from 1947 to 1996 until death (Kyle, 1997). Data are provided by the **survival** package (Therneau & Grambsch, 2000). We use two covariates, year of entry into the study and the time of MM diagnosis (Kyle, 1997). We see 71% incidence over 23 years (Kyle, 1997).

Fig. 2 B, F and Table 2 B demonstrate the performance over time on a test set. For discrimination, CBNN and DeepHit have a similar perfor-

Table 2
Performance at percentages of follow-up time in the complex simulation (A), multiple myeloma (MM) case study (B), free light chain (FLC) case study (C) and prostate cancer (Prostate) case study (D). Each table shows performance for each method at 25%, 50%, 75% and 100% of follow-up time. The models of interest are case-base with logistic regression (CBLR), Cox, Case-Base Neural Network (CBNN), DeepHit, DeepSurv, and Optimal (in the complex simulation). The best score at each percent of follow-up time is highlighted in bold. If tied, then all tied values are highlighted.

A:Complex	IPA				AUC_{IPCW}			
Method	25%	50%	75%	100%	25%	50%	75%	100%
Cox	0.56 (0.56,0.57)	0.41.00 (0.4,0.41)	-0.53 (-0.54,-0.52)	-1.87 (-1.89,-1.84)	0.93 (0.93,0.94)	0.89 (0.88,0.89)	0.79 (0.79,0.79)	0.77 (0.76,0.77)
CBLR	0.56 (0.56,0.57)	0.4 (0.4,0.41)	-0.46 (-0.46,-0.45)	-1.5 (-1.52,-1.48)	0.94 (0.94,0.94)	0.89 (0.89,0.89)	0.78 (0.78,0.79)	0.75 (0.75,0.75)
DeepSurv	0.65 (0.65,0.65)	0.4 (0.4,0.4)	-0.16 (-0.18,-0.13)	-1.02 (-1.08,-0.95)	0.94 (0.94,0.94)	0.89 (0.89,0.89)	0.85 (0.85,0.85)	0.88 (0.88,0.89)
DeepHit	0.68 (0.68,0.68)	0.3 (0.3,0.31)	0.00 (-0.01,0.02)	0.14 (0.13,0.15)	**0.96 (0.96,0.96)**	0.88 (0.88,0.89)	0.94 (0.94,0.95)	0.98 (0.97,0.98)
CBNN	**0.69 (0.69,0.69)**	0.43 (0.43,0.43)	0.25 (0.25,0.26)	0.27 (0.26,0.28)	**0.96 (0.96,0.96)**	**0.9 (0.9,0.9)**	**0.96 (0.96,0.96)**	0.99 (0.99,0.99)
Optimal	**0.69 (0.69,0.69)**	**0.44 (0.43,0.44)**	**0.27 (0.27,0.27)**	**0.34 (0.33,0.34)**	**0.96 (0.96,0.96)**	**0.9 (0.9,0.9)**	**0.96 (0.96,0.96)**	**1.00 (1.00,1.00)**

B:MM	IPA				AUC_{IPCW}			
Method	25%	50%	75%	100%	25%	50%	75%	100%
Cox	0.00 (0.00,0.00)	-0.09 (-0.1,-0.09)	-0.26 (-0.27,-0.25)	-0.45 (-0.47,-0.42)	0.51.00 (0.51,0.51)	0.37 (0.37,0.37)	0.13 (0.13,0.13)	0.03 (0.03,0.03)
CBLR	0.00 (0.00,0.00)	-0.1.00 (-0.1,-0.09)	-0.25 (-0.26,-0.24)	-0.42 (-0.44,-0.4)	0.51.00 (0.51,0.51)	0.37 (0.37,0.37)	0.13 (0.13,0.13)	0.03 (0.03,0.03)
DeepSurv	0.00 (-0.01,0.00)	-0.09 (-0.09,-0.08)	-0.24 (-0.26,-0.22)	-0.49 (-0.53,-0.45)	0.52 (0.52,0.53)	0.39 (0.39,0.4)	0.24 (0.21,0.27)	0.09 (0.06,0.12)
DeepHit	-0.16 (-0.2,-0.12)	-1.21.00 (-1.42,-1)	-3.86 (-4.4,-3.31)	-3.66 (-4.17,-3.15)	**0.59 (0.58,0.59)**	**0.53 (0.51,0.55)**	**0.52 (0.48,0.56)**	**0.32 (0.25,0.38)**
CBNN	**0.04 (0.04,0.04)**	**0.00 (-0.01,0.00)**	**-0.01.00 (-0.01,-0.01)**	**-0.01.00 (-0.01,-0.01)**	0.55 (0.55,0.56)	0.51.00 (0.49,0.53)	0.43 (0.4,0.47)	0.22 (0.17,0.27)

C:FLC	IPA				AUC_{IPCW}			
Method	25%	50%	75%	100%	25%	50%	75%	100%
Cox	**0.19 (0.19,0.19)**	0.29 (0.29,0.29)	0.33 (0.33,0.33)	**0.44 (0.44,0.44)**	**0.85 (0.85,0.85)**	0.85 (0.85,0.86)	0.84 (0.84,0.84)	0.8 (0.79,0.8)
CBLR	**0.19 (0.19,0.19)**	**0.3 (0.29,0.3)**	0.33 (0.33,0.33)	0.32 (0.31,0.32)	**0.85 (0.85,0.85)**	**0.86 (0.86,0.86)**	0.84 (0.84,0.84)	0.8 (0.8,0.8)
DeepSurv	**0.19 (0.18,0.2)**	0.29 (0.28,0.31)	0.33 (0.32,0.34)	0.38 (0.33,0.43)	0.84 (0.82,0.85)	0.85 (0.83,0.87)	0.83 (0.82,0.85)	0.82 (0.8,0.84)
DeepHit	0.18 (0.18,0.19)	0.29 (0.29,0.29)	0.33 (0.33,0.33)	0.41.00 (0.36,0.46)	**0.85 (0.85,0.85)**	**0.86 (0.86,0.86)**	**0.85 (0.85,0.85)**	0.83 (0.83,0.84)
CBNN	**0.19 (0.19,0.2)**	**0.3 (0.3,0.3)**	**0.34 (0.34,0.34)**	0.42 (0.41,0.42)	**0.85 (0.85,0.85)**	**0.86 (0.86,0.86)**	**0.85 (0.85,0.85)**	**0.84 (0.83,0.84)**

D:Prostate	IPA				AUC_{IPCW}			
Method	25%	50%	75%	100%	25%	50%	75%	100%
Cox	**0.04 (0.04,0.04)**	**0.09 (0.09,0.09)**	**0.14 (0.14,0.14)**	**0.17 (0.17,0.17)**	**0.77 (0.76,0.77)**	**0.75 (0.75,0.75)**	**0.76 (0.76,0.76)**	0.71.00 (0.71,0.71)
CBLR	**0.04 (0.04,0.04)**	**0.09 (0.09,0.09)**	**0.14 (0.13,0.14)**	**0.17 (0.17,0.17)**	**0.77 (0.76,0.77)**	**0.75 (0.75,0.75)**	**0.76 (0.76,0.76)**	0.71.00 (0.71,0.71)
DeepSurv	0.03 (0.03,0.04)	0.06 (0.05,0.07)	0.1.00 (0.09,0.11)	0.12 (0.1,0.13)	0.71.00 (0.69,0.73)	0.7 (0.68,0.72)	0.7 (0.68,0.72)	0.68 (0.66,0.7)
DeepHit	**0.04 (0.03,0.04)**	0.08 (0.08,0.09)	0.12 (0.12,0.13)	0.16 (0.15,0.16)	**0.77 (0.76,0.77)**	**0.75 (0.74,0.75)**	0.75 (0.75,0.75)	**0.72 (0.72,0.73)**
CBNN	**0.04 (0.04,0.04)**	0.08 (0.07,0.08)	0.12 (0.12,0.13)	0.14 (0.13,0.14)	0.76 (0.76,0.76)	**0.75 (0.75,0.75)**	0.74 (0.74,0.74)	0.71.00 (0.71,0.72)

mance, followed by DeepSurv and finally the linear models performing worst (Fig. 2 F). For both discrimination and calibration, CBNN performs substantially better than the linear models and DeepSurv, with DeepHit having the worst performance overall (Fig. 2 B). Together, CBNN is the best calibrated model and one of the best at discrimination in the MM case study.

5.2. Performance evaluation on free light chain dataset

Serum free light chain (FLC) is a known diagnostic tool for assessing MM (Dispenzieri et al., 2009). We are interested in whether there is a predictive benefit if the FLC markers varies with time. As such the second case study examines the relationship between serum FLC and mortality in a random sample of half the individuals in $\frac{2}{3}$ of the residents of Olmsted County over the age of 50 (Dispenzieri et al., 2012). Data are provided by the **survival** package (Therneau & Grambsch, 2000) for 7874 subjects tracked until death with 27% incidence over 14 years (Dispenzieri et al., 2012). We use five covariates, total serum FLC (sum of kappa and lambda), age, sex, serum creatine and monoclonal gammopathy state (Dispenzieri et al., 2012).

Fig. 2 C, G and Table 2 C demonstrate the performance over time on a test set. CBNN, DeepHit and the linear models perform best at discrimination, followed by DeepSurv (Fig. 2 G). The rankings remain the same aside from DeepHit where the performance periodically drops to worse than DeepSurv (Fig. 2 C). Together, CBNN outperforms the competing models in terms of both calibration and discrimination. However, CBNN is only slightly better than the linear models.

5.3. Performance evaluation on prostate cancer dataset

As the CBNN model saw large predictive benefits on MM and small predictive benefits on FLC, we wish to examine longitudinal data with another complex risk profile. The third case study (Prostate) examines prostate cancer survival on a publicly available simulation of the Surveillance, Epidemiology, and End Results (SEER) medicare study (Lu-Yao et al., 2009). The Prostate dataset, which is contained in the **asaur** package (Moore, 2016), has a record of competing risks. As we are only interested in the single event scenario, we only keep individuals with prostate cancer death or censoring. This subset tracks 11054 individuals with three covariates: differentiation grade, age

group and cancer state. There is a 7% incidence over 10 years (Lu-Yao et al., 2009).

Fig. 2 C, G and Table 2 C demonstrate the performance over time on a test set. In the prostate case study, the linear models outperform the other models in both discrimination and calibration, followed by CBNN and DeepHit and finally DeepSurv (Fig. 2 C, G). Aside from DeepSurv, the performance is similar across all models.

6. Discussion

CBNNs model survival outcomes by using neural networks on case-base sampled data. We incorporate follow-up time as a feature, providing a data-driven estimate of a flexible baseline hazard and time-varying interactions in our hazard function. The two competing neural network models we evaluated cannot model time-varying interactions (Katzman et al., 2018; Lee et al., 2018). The CBNN model performs better due to its ability to model time-varying interactions and a complex baseline hazard.

The complex simulation requires a method that can learn both time-varying interactions and have a flexible baseline hazard. Based on our complex simulation results (Fig. 2 A, E and Table 2 A), CBNN outperforms the competitors. This simulation shows how all models perform under ideal conditions with minimal noise in the data, while the three case studies assess their performance in realistic conditions. In the MM case study, flexibility in both interaction modeling and baseline hazard improves the performance of CBNN over the other models, suggesting that this flexibility aids calibration (Fig. 2 B, F and Table 2 B). Upon examination of the FLC case study, CBNN demonstrates a small improvement to performance compared to the linear models and DeepHit for both IPA and AUC (Fig. 2 C, G and Table 2 C). In the Prostate case study, the linear models outperform the neural network ones, while CBNN and DeepHit alternate their positions depending on the follow-up time of interest and DeepSurv maintains last place (Fig. 2 C, G and Table 2 C). We attribute this to potential over-parameterization in the neural network models, as we did not test for fewer nodes in each hidden layer, even with dropout. Though the ranking places the linear models above the neural network ones, their overall performance falls within a small range of IPA and AUC values aside from DeepSurv.

Compared to CBNN, the neural network competitors have limitations. DeepSurv is a proportional hazards model and does not estimate the baseline hazard (Katzman et al., 2018). DeepHit requires an alpha hyperparameter, assumes a single distribution for the baseline hazard and models the survival function directly (Lee et al., 2018). The alternative neural network methods match on time, while CBNN models time directly. By modeling time directly, CBNN can approximate a flexible baseline hazard with whichever time-varying effects may be relevant to the outcome of interest.

While we apply a full grid search with three-fold cross-validation for all neural network models, there are an infinite number of hyperparameters we did not test. A completely exhaustive search is computationally infeasible; therefore, it is reasonable to expect that there exists a set of hyperparameters that is better suited for each model in each study. However, we test a reasonably large range while accounting for potential over-fitting by including dropout (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014). We provide the same options to all models aside from DeepHit, which has a method specific hyperparameter α (Lee et al., 2018). Another method of particular interest is DSM, as it is a fully parametric model (Nagpal et al., 2021). However, with our hyperparameter options, DSM did not converge in the complex simulations, which may have been due to issues in software or method specific limitations. Therefore, we did not include this method in our comparison. With these limitations in mind, we summarize the differences in performance across all methods in our comparisons.

If prediction is our goal, we suggest CBNN as the best model in the Complex simulation, MM case study and FLC case study. The linear models were competitive in the FLC case study and performed best in the Prostate case study. Though neural network interpretability is steadily improving (Zhang, Tiňo, Leonardis, & Tang, 2021), there is still a trade-off compared to regression models, especially when the predictive performance gain is minimal, like in the FLC case study. We suggest that reference models should be included when assessing neural network models. Both a null model (KM curve) and a linear model (either Cox or a flexible baseline hazard model like CBLR) provide insight as to whether the neural network model is learning anything useful beyond linear predictors that were not accounted for.

7. Conclusions

Our study was motivated by the lack of easily implemented survival models based on neural networks that can approximate time-varying interactions. This led us to apply the case-base sampling technique, which has previously been used with logistic regression (Hanley & Miettinen, 2009), to neural network models for a data-driven approach to time-varying interaction modeling. In our paper, we aim to compare CBNNs with existing methods. We assess the discrimination and calibration of each method in a simulation with time-varying interactions and a complex baseline hazard, and three case studies. CBNNs outperform all competitors in the complex simulation and two case studies while maintaining competitive performance in a final case study. Once we perform case-base sampling and adjust for the sampling bias, we can use a sigmoid activation function to predict our hazard function. Our approach provides an alternative to incorporating censored individuals, treating survival outcomes as binary ones. Forgoing the requirement of custom loss functions, CBNNs only require the use of standard components in machine learning libraries (specifically, the add layer to adjust for sampling bias and the sigmoid activation function) after case-base sampling. Due to the simplicity in its implementation and by extension user experience, CBNNs are both a user-friendly approach to data-driven, single event survival analysis and is easily extendable to any feed-forward neural network framework.

CRediT authorship contribution statement

Jesse Islam: Conceived the proof of concept, developed the theoretical formalism, developed the neural network code base, wrote the majority of the manuscript, performed the simulations and analyses, discussed the results and contributed to the final manuscript. **Maxime Turgeon:** Wrote the majority of the case-base sampling section, provided key insights into the core sampling technique (case-base), discussed the results and contributed to the final manuscript. **Robert Sladek:** Provided multiple edits to the manuscript structure and key insights, discussed the results and contributed to the final manuscript. **Sahir Bhatnagar:** Provided key insights into the core sampling technique (case-base), discussed the results and contributed to the final manuscript.

Declaration of competing interest

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

Data and code availability

The MM and FLC datasets are available in the **survival** package in R (Therneau & Grambsch, 2000). The Prostate dataset is available as part of the **asaur** package in R (Moore, 2016). The code for this manuscript and its analyses can be found at https://github.com/Jesse-Islam/cbnnManuscript. The software package making CBNNs easier to implement can be found at https://github.com/Jesse-Islam/cbnn.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work the author(s) used Microsoft 365 (Word) and ProWritingAid in order to improve grammar, assess typos and improve general sentence structure. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

Acknowledgments

We thank Dr. James Meigs, the project leader of UM1DK078616 and R01HL151855, for his support and helpful discussions. We would also like to thank Dr. James Hanley for his support and discussions while extending the case-base methodology.

Financial disclosure

The subcontracts from National Institute of Diabetes and Digestive and Kidney Diseases UM1DK078616 and National Heart Lung and Blood Institute R01HL151855 to R.S. supported this work. The work was also supported as part of the Congressionally Directed Medical Research Programs (CDMRP) award W81XWH-17-1-0347.

References

Allaire, J., & Chollet, F. (2021). Keras: R interface to 'keras'. URL https://CRAN.R-project.org/package=keras R package version 2.7.0.

Barbieri, S., Mehta, S., Wu, B., Bharat, C., Poppe, K., Jorm, L., et al. (2022). Predicting cardiovascular risk from national administrative databases using a combined survival analysis and deep learning approach. *International Journal of Epidemiology*, *51*(3), 931–944.

Bhatnagar, S. R., Turgeon, M., Islam, J., Hanley, J. A., & Saarela, O. (2020). Casebase: An alternative framework for survival analysis and comparison of event rates. arXiv preprint arXiv:2009.10264.

Bhatnagar*, S. R., Turgeon*, M., Islam, J., Hanley, J. A., & Saarela, O. (2022). The R Journal: casebase: An alternative framework for survival analysis and comparison of event rates. *The R Journal*, *14*, 59–79. http://dx.doi.org/10.32614/RJ-2022-052.

Bice, N., Kirby, N., Bahr, T., Rasmussen, K., Saenz, D., Wagner, T., et al. (2020). Deep learning-based survival analysis for brain metastasis patients with the national cancer database. *Journal of Applied Clinical Medical Physics*, *21*(9), 187–192.

Blanche, P., Proust-Lima, C., Loubere, L., Berr, C., Dartigues, J.-F., & Jacqmin-Gadda, H. (2015). Quantifying and comparing dynamic predictive accuracy of joint models for longitudinal marker and time-to-event in presence of censoring and competing risks. *Biometrics*, *71*(1), 102–113.

Brilleman, S. L., Wolfe, R., Moreno-Betancur, M., & Crowther, M. J. (2020). Simulating survival data using the simsurv R package. *Journal of Statistical Software*, *97*(3), 1–27. http://dx.doi.org/10.18637/jss.v097.i03.

Chen, C., Cao, Y., Li, W., Liu, Z., Liu, P., Tian, X., et al. (2023). The pathological risk score: A new deep learning-based signature for predicting survival in cervical cancer. *Cancer Medicine*, *12*(2), 1051–1063.

Chen, R. J., Lu, M. Y., Wang, J., Williamson, D. F., Rodig, S. J., Lindeman, N. I., et al. (2020). Pathomic fusion: an integrated framework for fusing histopathology and genomic features for cancer diagnosis and prognosis. *IEEE Transactions on Medical Imaging*, *41*(4), 757–770.

Ching, T., Zhu, X., & Garmire, L. X. (2018). Cox-nnet: an artificial neural network method for prognosis prediction of high-throughput omics data. *PLoS Computational Biology*, *14*(4), Article e1006076.

Coradini, D., Daidone, M. G., Boracchi, P., Biganzoli, E., Oriana, S., Bresciani, G., et al. (2000). Time-dependent relevance of steroid receptors in breast cancer. *Journal of Clinical Oncology*, *18*(14), 2702–2709.

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

Dispenzieri, A., Katzmann, J. A., Kyle, R. A., Larson, D. R., Therneau, T. M., Colby, C. L., et al. (2012). Use of nonclonal serum immunoglobulin free light chains to predict overall survival in the general population. In *Mayo clinic proceedings, Vol. 87* (pp. 517–523). Elsevier.

Dispenzieri, A., Kyle, R., Merlini, G., Miguel, J., Ludwig, H., Hájek, R., et al. (2009). International Myeloma Working Group guidelines for serum-free light chain analysis in multiple myeloma and related disorders. *Leukemia*, *23*(2), 215–224.

Gao, D., Grunwald, G. K., Rumsfeld, J. S., Schooley, L., MacKenzie, T., & Shroyer, A. L. W. (2006). Time-varying risk factors for long-term mortality after coronary artery bypass graft surgery. *Annals of Thoracic Surgery*, *81*(3), 793–799.

Gensheimer, M. F., & Narasimhan, B. (2019). A scalable discrete-time survival model for neural networks. *PeerJ*, *7*, Article e6257.

Gerds, T. A., & Kattan, M. W. (2021). *Medical risk prediction models: with ties to machine learning* (1st ed.). Chapman and Hall/CRC, http://dx.doi.org/10.1201/9781138384484, R package version 2021.10.10.

Giunchiglia, E., Nemchenko, A., & van der Schaar, M. (2018). Rnn-surv: A deep recurrent model for survival analysis. In *Artificial neural networks and machine learning–ICANN 2018: 27th international conference on artificial neural networks, rhodes, Greece, October 4-7, 2018, proceedings, part III 27* (pp. 23–32). Springer.

Graf, E., Schmoor, C., Sauerbrei, W., & Schumacher, M. (1999). Assessment and comparison of prognostic classification schemes for survival data. *Statistics in Medicine*, *18*(17–18), 2529–2545.

Gulli, A., & Pal, S. (2017). *Deep learning with keras*. Packt Publishing Ltd.

Hanley, J. A., & Miettinen, O. S. (2009). Fitting smooth-in-time prognostic risk functions via logistic regression. *International Journal of Biostatistics*, *5*(1).

Hao, Y., Jing, X. Y., & Sun, Q. (2022). Joint learning sample similarity and correlation representation for cancer survival prediction. *BMC bioinformatics*, *23*(1), 553.

Hao, L., Kim, J., Kwon, S., & Ha, I. D. (2021). Deep learning-based survival analysis for high-dimensional survival data. *Mathematics*, *9*(11), 1244.

Huang, Z., Johnson, T. S., Han, Z., Helm, B., Cao, S., Zhang, C., et al. (2020). Deep learning-based cancer survival prognosis from RNA-seq data: approaches and evaluations. *BMC Medical Genomics*, *13*, 1–12.

Hughes-Hallett, D., Gleason, A. M., & McCallum, W. G. (2020). *Calculus: Single and multivariable*. John Wiley & Sons.

Jackson, C. (2016). flexsurv: A platform for parametric survival modeling in R. *Journal of Statistical Software*, *70*(8), 1–33. http://dx.doi.org/10.18637/jss.v070.i08, R package version 2.0.

Jia, L., Ren, X., Wu, W., Zhao, J., Qiang, Y., & Yang, Q. (2023). DCCAFN: deep convolution cascade attention fusion network based on imaging genomics for prediction survival analysis of lung cancer. *Complex & Intelligent Systems*, 1–16.

Kattan, M. W., & Gerds, T. A. (2018). The index of prediction accuracy: an intuitive measure useful for evaluating risk prediction models. *Diagnostic and Prognostic Research*, *2*(1), 1–7.

Katzman, J. L., Shaham, U., Cloninger, A., Bates, J., Jiang, T., & Kluger, Y. (2018). DeepSurv: personalized treatment recommender system using a Cox proportional hazards deep neural network. *BMC Medical Research Methodology*, *18*(1), 24.

Kaynar, G., Cakmakci, D., Bund, C., Todeschi, J., Namer, I. J., & Cicek, A. E. (2023). Pideel: metabolic pathway-informed deep learning model for survival analysis and pathological classification of gliomas. *Bioinformatics*, *39*(11), btad684.

Kim, D. W., Lee, S., Kwon, S., Nam, W., Cha, I.-H., & Kim, H. J. (2019). Deep learning-based survival prediction of oral cancer patients. *Scientific Reports*, *9*(1), 1–10.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Kopper, P., Wiegrebe, S., Bischl, B., Bender, A., & Rügamer, D. (2022). DeepPAMM: Deep piecewise exponential additive mixed models for complex hazard structures in survival analysis. In *Pacific-Asia conference on knowledge discovery and data mining* (pp. 249–261). Springer.

Kyle, R. (1997). Long term survival in multiple myeloma. *New England Journal of Medicine*.

Lee, C., Zame, W. R., Yoon, J., & Schaar, M. v. d. (2018). DeepHit: A deep learning approach to survival analysis with competing risks. In *AAAI conference on artificial intelligence* (pp. 2314–2321). Software from pycox version 0.2.2.

Li, R., Wu, X., Li, A., & Wang, M. (2022). HFBSurv: hierarchical multimodal fusion with factorized bilinear models for cancer survival prediction. *Bioinformatics*, *38*(9), 2587–2594.

Lu-Yao, G. L., Albertsen, P. C., Moore, D. F., Shih, W., Lin, Y., DiPaola, R. S., et al. (2009). Outcomes of localized prostate cancer following conservative management. *Jama*, *302*(11), 1202–1209.

Mantel, N. (1973). Synthetic retrospective studies and related topics. *Biometrics*, 479–486.

Meng, X., Wang, X., Zhang, X., Zhang, C., Zhang, Z., Zhang, K., et al. (2022). A novel attention-mechanism based cox survival model by exploiting pan-cancer empirical genomic information. *Cells*, *11*(9), 1421.

Mobadersany, P., Yousefi, S., Amgad, M., Gutman, D. A., Barnholtz-Sloan, J. S., Velázquez Vega, J. E., et al. (2018). Predicting cancer outcomes from histology and genomics using convolutional networks. *Proceedings of the National Academy of Sciences*, *115*(13), E2970–E2979.

Moore, D. F. (2016). asaur: Data sets for "Applied Survival Analysis Using R". URL https://CRAN.R-project.org/package=asaur R package version 0.50.

Na, E., Cho, S., Kim, D. J., Choi, J., & Han, E. (2020). Time-varying and dose-dependent effect of long-term statin use on risk of type 2 diabetes: a retrospective cohort study. *Cardiovascular Diabetology*, *19*(1), 1–11.

Nagpal, C., Jeanselme, V., & Dubrawski, A. (2021). Deep parametric time-to-event regression with time-varying covariates. In *Survival prediction-algorithms, challenges and applications* (pp. 184–193). PMLR.

Nagpal, C., Li, X. R., & Dubrawski, A. (2021). Deep survival machines: Fully parametric survival regression and representation learning for censored data with competing risks. *IEEE Journal of Biomedical and Health Informatics*, Software downloaded March 10, 2021.

R. Core Team (2021). *R: A Language and Environment for Statistical Computing*. URL https://www.R-project.org/ Version 4.1.2.

Royston, P., & Parmar, M. K. (2002). Flexible parametric proportional-hazards and proportional-odds models for censored survival data, with application to prognostic modelling and estimation of treatment effects. *Statistics in Medicine*, *21*(15), 2175–2197.

Saarela, O. (2016). A case-base sampling method for estimating recurrent event intensities. *Lifetime Data Analysis*, *22*, 589–605.

Saarela, O., & Hanley, J. A. (2015). Case-base methods for studying vaccination safety. *Biometrics*, *71*(1), 42–52.

Salmon, D., & Melendez-Torres, G. (2023). Clinical effectiveness reporting of novel cancer drugs in the context of non-proportional hazards: a review of nice single technology appraisals. *International Journal of Technology Assessment in Health Care*, *39*(1), Article e16.

She, Y., Jin, Z., Wu, J., Deng, J., Zhang, L., Su, H., et al. (2020). Development and validation of a deep learning model for non-small cell lung cancer survival. *JAMA Network Open*, *3*(6), e205842.

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

Tang, Z., Liu, L., Chen, Z., Ma, G., Dong, J., Sun, X., et al. (2023). Explainable survival analysis with uncertainty using convolution-involved vision transformer. *Computerized Medical Imaging and Graphics*, *110*, Article 102302.

Therneau, T. M., & Grambsch, P. M. (2000). *Modeling survival data: extending the cox model*. New York: Springer, R package version 3.2-11.

Ushey, K., Allaire, J., & Tang, Y. (2021). Reticulate: Interface to 'Python'. URL https://CRAN.R-project.org/package=reticulate R package version 1.22.

Vale-Silva, L. A., & Rohr, K. (2021). Long-term cancer survival prediction using multimodal deep learning. *Scientific Reports*, *11*(1), 13505.

Van Rossum, G., & Drake, F. L. (2009). *Python 3 reference manual*. Scotts Valley, CA: CreateSpace, Version 3.8.12.

Wang, Z., Li, R., Wang, M., & Li, A. (2021). GPDBN: deep bilinear network integrating both genomic data and pathological images for breast cancer prognosis prediction. *Bioinformatics*, *37*(18), 2963–2970.

Wulczyn, E., Steiner, D. F., Xu, Z., Sadhwani, A., Wang, H., Flament-Auvigne, I., et al. (2020). Deep learning-based survival prediction for multiple cancer types using histopathology images. *PLoS One*, *15*(6), Article e0233678.

Yin, Q., Chen, W., Wu, R., & Wei, Z. (2022). Cox-ResNet: A survival analysis model based on residual neural networks for gene expression data. In *2022 IEEE international conference on networking, sensing and control* (pp. 1–6). IEEE.

Yu, H., Huang, T., Feng, B., & Lyu, J. (2022). Deep-learning model for predicting the survival of rectal adenocarcinoma patients based on a surveillance, epidemiology, and end results analysis. *BMC Cancer*, *22*(1), 1–14.

Zadeh Shirazi, A., Fornaciari, E., Bagherian, N. S., Ebert, L. M., Koszyca, B., & Gomez, G. A. (2020). DeepSurvNet: deep survival convolutional network for brain cancer survival rate classification based on histopathological images. *Medical & Biological Engineering & Computing*, *58*, 1031–1045.

Zhang, Y., Tiňo, P., Leonardis, A., & Tang, K. (2021). A survey on neural network interpretability. *IEEE Transactions on Emerging Topics in Computational Intelligence*, *5*(5), 726–742.

Zhu, J., Liu, X., Zhang, J., Li, J., Chen, L., Huang, C., et al. (2022). Time-varying association between body mass index and all-cause mortality in patients with hypertension. *International Journal of Obesity*, *46*(2), 316–324.

Zhu, X., Yao, J., Zhu, F., & Huang, J. (2017). Wsisa: Making survival prediction from whole slide histopathological images. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 7234–7242).