Ensemble-Based Survival Models with the Self-Attended Beran Estimator Predictions

Lev V. Utkin, Semen P. Khomets,Vlada A. Efremenko,
Andrei V. Konstantinov and Natalya M. Verbova
Higher School of Artificial Intelligence Technologies
Peter the Great St.Petersburg Polytechnic University
St.Petersburg, Russia
e-mail: utkin_lv@spbstu.ru, homets_sp@spbstu.ru,
efremenko_va@spbstu.ru, konstantinov_av@spbstu.ru,
verbova_nm@spbstu.ru

Abstract

Survival analysis predicts the time until an event of interest, such as failure or death, but faces challenges due to censored data, where some events remain unobserved. Ensemble-based models, like random survival forests and gradient boosting, are widely used but can produce unstable predictions due to variations in bootstrap samples. To address this, we propose SurvBESA (Survival Beran Estimators Self-Attended), a novel ensemble model that combines Beran estimators with a self-attention mechanism. Unlike traditional methods, SurvBESA applies self-attention to predicted survival functions, smoothing out noise by adjusting each survival function based on its similarity to neighboring survival functions. We also explore a special case using Huber's contamination model to define attention weights, simplifying training to a quadratic or linear optimization problem. Numerical experiments show that SurvBESA outperforms state-of-the-art models. The implementation of SurvBESA is publicly available.

1 Introduction

Survival analysis is a vital field focused on predicting the time until an event of interest occurs [1]. A key challenge in this domain is the presence of censored data, where some events remain unobserved during the study period, resulting in observed times that are less than or equal to the true event times [2]. As a result, survival datasets typically contain a mix of censored and uncensored observations, requiring specialized methods to handle such data effectively.

One of the most powerful approaches for survival analysis is the use of ensemble-based models. These models combine multiple weak or base models to create a robust, accurate, and generalizable predictive model. Ensemble methods have been extensively applied and refined in survival analysis, often as extensions of well-established techniques. For instance, Random Survival Forests (RSFs) [3, 4, 5, 6, 7, 8, 9] adapt Random Forests to survival data, while gradient boosting-based survival models [10, 11] extend Gradient Boosting Machines. The key differences among these models lie in the choice of weak learners and the strategies used to aggregate their predictions. An innovative approach by Meier et al. [12] employs the Cox proportional hazards model [13] as a weak learner.

While the Cox model is a cornerstone of survival analysis, it has limitations: it estimates conditional survival measures using feature vectors but ignores their relative positions, leading to inaccuracies in datasets with multi-cluster structures. To address this, the Beran estimator [14] is used. This estimator calculates the conditional survival function by weighting instances based on their proximity to the analyzed instance using kernel functions, effectively performing kernel regression.

Extensive surveys and discussions on ensemble-based approaches and their applications can be found in [15, 16, 17, 18, 19, 20].

The Beran estimator's ability to incorporate relationships between feature vectors makes it an attractive choice as a weak learner in new ensemble-based models. A simple implementation involves averaging predictions from multiple Beran estimators using bagging. However, predictions from different weak models, in the form of survival functions (SFs), can vary significantly, particularly when trained on data from different bootstrap samples. This may lead to incorrect and unstable aggregation of the base model predictions, resulting in an unreliable aggregated SF for the ensemble.

We propose to aggregate the survival function predictions by applying the self-attention mechanism [21], which allows a model to weigh the importance of different elements in a sequence relative to each other. The main idea behind using self-attention is to capture dependencies among the predictions provided by the different Beran models in the ensemble and to remove noise or anomalous predictions that may arise due to a potentially "unfortunate" selection of a subset of training examples during the bootstrap sampling process used to construct a base Beran estimator. In contrast to the self-attention mechanism in transformer-based survival models [22, 23, 24], which use self-attention to weigh feature vectors from a training set, the proposed model aims to weigh the SFs predicted by the base Beran estimators from the ensemble. In other words, we propose to adjust the predicted SFs using self-attention based on the neighboring SFs. Specifically, the feature vectors in the conventional self-attention framework are replaced with predicted SFs. As a result, anomalous or noisy SFs are smoothed in accordance with the nearest SFs. The proposed model is called SurvBESA (Survival Beran Estimators Self-Attended).

We also consider an interesting special case where the self-attention weights are defined using Huber's ϵ-contamination model [25] with parameter ϵ and its imprecise extension [26]. A key feature of this case is that the training task is reduced to a quadratic or linear optimization problem. Similar approaches applying the ϵ-contamination model have been used in [27, 28].

Numerical experiments comparing SurvBESA with other survival models, including Random Survival Forests (RSF) [3], GBM Cox [29], GBM AFT [30], and the standalone Beran estimator.

The implementation of SurvBESA is publicly available at: `https://github.com/NTAILab/SurvBESA`.

The paper is organized as follows. Related work reviewing papers devoted to machine learning in survival analysis and to the self-attention mechanism can be found in Section 2. A short description of the survival analysis concepts is given in Section 3. An idea of SurvBESA as an attention-based ensemble of Beran estimators is provided in Section 4. Numerical experiments comparing different survival models trained on synthetic and real data are considered in Section 5. Concluding remarks can be found in Section 6.

2 Related work

Machine learning models in survival analysis. Comprehensive overviews of survival models can be found in [1, 31]. Many of these models are extensions of standard machine learning techniques adapted to handle survival data. In recent years, neural networks and deep learning have gained traction in survival analysis. Notable contributions include deep survival models [32, 33, 2, 34], a deep recurrent survival model [35], convolutional neural networks for survival tasks [36], and transformer-based survival models [22, 23, 24].

However, survival models based on neural networks often require large amounts of training data, which can be a significant limitation in many real-world applications. Therefore, ensemble-based survival models that utilize simpler weak learners may offer more robust and accurate predictions, particularly when data is limited or high-dimensional. Basic principles of survival ensembles was presented by Hothorn et al. [37]. Various ensemble-based survival models were also presented in [10, 11, 12, 4, 5, 6, 7, 8, 9]. However, it is interesting to point out that there are no ensemble-based models which use the Beran estimator as a weak model. At the same time, our experiments show that models based on the Beran estimators can compete with other models in case of complex data structure.

Self-attention. The self-attention mechanism, introduced by Vaswani et al. [21], is a key component of the Transformer neural network architecture. It builds on earlier works, such as Cheng et al. [38] and Parikh et al. [39]. Since its introduction, it has been widely adopted across various domains, including sentence embedding [40], machine translation and natural language processing [41, 42], speech recognition [43, 44], and image recognition [45, 46, 47, 48, 49, 50, 51, 52, 53].

Numerous survey papers have explored the diverse aspects and applications of attention and self-attention mechanisms, including [54, 55, 56, 57, 58, 59, 60, 61].

The self-attention mechanism implemented in a deep gated neural network for survival analysis was introduced in [62]. The idea behind using the self-attention mechanism in the proposed neural network is that it treats time as an additional input covariate, with the condition that the smaller the time interval, the higher the attention weight. SurvBESA, however, uses self-attention differently. It serves as an aggregation operation for the predictions of the Beran estimators.

2.1 Basic concepts of survival analysis

The dataset $\mathcal{A}$ is assumed to consist of n vectors of the form $(\mathbf{x}_i, \delta_i, T_i)$, where $i = 1, ..., n$. Here $\mathbf{x}_i^{\mathrm{T}} \in \mathbb{R}^d$ represents the feature vector for the i-th object, $T_i \in \mathbb{R}_+$ is the event time, and $\delta_i \in \{0, 1\}$ is the censoring indicator. Specifically, $\delta_i = 1$ indicates that the event is observed (uncensored), while $\delta_i = 0$ corresponds to a censored observation [63]. Given the dataset $\mathcal{A}$, the goal is to construct a survival model capable of estimating the event time T for a new object $\mathbf{x}$.

This estimation can be represented by the conditional SF, denoted $S(t \mid \mathbf{x})$, which is the probability of surviving beyond time t, i.e., $S(t \mid \mathbf{x}) = \Pr\{T > t \mid \mathbf{x}\}$. Alternatively, the estimation can be expressed through the cumulative hazard function (CHF), denoted as $H(t \mid \mathbf{x})$, which is related to the SF as $H(t \mid \mathbf{x}) = -\ln S(t \mid \mathbf{x})$. While other representations exist [1], we focus on these two.

A key question in survival analysis is how to compare the performance of different survival models. One widely used metric is the C-index, introduced by Harrell et al. [64]. The C-index estimates the probability that the predicted event times of a pair of objects are correctly ranked [1]. Let $\mathcal{J}$ denote the set of all pairs (i, j) of objects satisfying $\delta_i = 1$ and $T_i < T_j$. The C-index can then be computed as [65, 1]:

$$C = \frac{1}{\#\mathcal{J}} \sum_{(i,j)\in\mathcal{J}} \mathbf{1}[\widehat{T}_i < \widehat{T}_j], \tag{1}$$

where $\widehat{T}_i$ and $\widehat{T}_j$ are predicted event times for objects with indices i and j, respectively.

Let $t_1 < t_2 < ... < t_n$ be an ordered sequence of times $\{T_1, ..., T_n\}$. Then the SF can be estimated using the Beran estimator [14] as follows:

$$S(t \mid \mathbf{x}, \mathcal{A}) = \prod_{t_i \leq t} \left\{ 1 - \frac{\alpha(\mathbf{x}, \mathbf{x}_i)}{1 - \sum_{j=1}^{i-1} \alpha(\mathbf{x}, \mathbf{x}_j)} \right\}^{\delta_i}, \tag{2}$$

where the weight $\alpha(\mathbf{x}, \mathbf{x}_i)$ reflects the relevance of the i-th object $\mathbf{x}_i$ to the feature vector $\mathbf{x}$. This weight can be defined using a kernel function:

$$\alpha(\mathbf{x}, \mathbf{x}_i) = \frac{K(\mathbf{x}, \mathbf{x}_i)}{\sum_{j=1}^{n} K(\mathbf{x}, \mathbf{x}_j)}. \tag{3}$$

For example, when the Gaussian kernel is used, the weights correspond to the softmax operation with a temperature parameter τ:

$$\alpha(\mathbf{x}, \mathbf{x}_i) = \text{softmax}\left(-\frac{\|\mathbf{x} - \mathbf{x}_i\|^2}{\tau} \right). \tag{4}$$

Notably, the Beran estimator generalizes the Kaplan-Meier estimator. When all weights $\alpha(\mathbf{x}, \mathbf{x}_i)$ are set to $1/n$ for $i = 1, ..., n$, the Beran estimator reduces to the Kaplan-Meier estimator.

3 Attention-based ensemble of Beran estimators

3.1 Ensemble of Beran estimators

An ensemble of Beran estimators is constructed using the standard bagging approach. This involves randomly selecting a subset $\mathcal{A}_k$ of m objects from the dataset $\mathcal{A}$. Let $t_1^{(k)} < ... < t_m^{(k)}$ be the ordered event times of the objects in $\mathcal{A}_k$. The SF predicted by the k-th Beran estimator trained on the subset $\mathcal{A}_k$ and denoted as $S^{(k)}(t \mid \mathbf{x})$ is computed as follows:

$$S(t \mid \mathbf{x}, \mathcal{A}_k) = S^{(k)}(t \mid \mathbf{x}) = \prod_{t_i^{(k)} \leq t} \left\{ \frac{1 - \sum_{j=1}^{i} \alpha(\mathbf{x}, \mathbf{x}_j)}{1 - \sum_{j=1}^{i-1} \alpha(\mathbf{x}, \mathbf{x}_j)} \right\}^{\delta_i}. \tag{5}$$

SFs $S(t \mid \mathbf{x}, \mathcal{A}_k)$, where $k = 1, ..., M$, obtained using M Beran estimators are aggregated to produce the final SF as follows:

$$S(t \mid \mathbf{x}) = \frac{1}{M} \sum_{k=1}^{M} S^{(k)}(t \mid \mathbf{x}). \tag{6}$$

This model operates similarly to the standard random forest, with a key distinction: the decision trees are replaced with Beran estimators. While random forests aggregate predictions from multiple decision trees, this model aggregates predictions from multiple Beran estimators, leveraging their ability to handle survival data effectively.

3.2 SurvBESA

One of the challenges in using the simple averaging operation to aggregate predictions from the Beran estimators is the potential for significant variability in the predictions. Specifically, some weak learners may produce anomalous survival functions (SFs) due to various factors, such as the specific subset $\mathcal{A}_k$ used for training or the multi-cluster structure of the data. These anomalous SFs can introduce bias into the final predictions, leading to unreliable estimates. To address this issue, we propose an alternative model designed to mitigate these problems and improve the robustness of the predictions.

The first key idea behind the proposed model is to define attention weights and new base SFs using the self-attention mechanism. This approach allows the model to dynamically weigh the contributions of different base SFs based on their relevance and relationships, improving the robustness and accuracy of the final predictions.

$$\widetilde{S}^{(j)}(t \mid \mathbf{x}, \theta) = \sum_{k=1, k \neq j}^{M} \beta \left(S^{(j)}(t \mid \mathbf{x}), S^{(k)}(t \mid \mathbf{x}), \theta \right) \cdot S^{(k)}(t \mid \mathbf{x}), \tag{7}$$

where $\beta(\cdot, \cdot, \cdot)$ is the self-attention weight which establishes the relationship between SFs $S^{(j)}(t \mid \mathbf{x})$ and $S^{(k)}(t \mid \mathbf{x})$; θ is the vector of training or tuning parameters.

In the context of the attention mechanism, $S^{(j)}(t \mid \mathbf{x})$ serves as the query, while $S^{(k)}(t \mid \mathbf{x})$ functions as both the key and the value. This setup allows the model to compute attention weights based on the relationships between the predicted survival functions, enabling more informed aggregation of the base predictions.

From the above, it follows that each base SF is adjusted using the self-attention mechanism, taking into account the neighboring SFs. By applying self-attention, we effectively reduce noise that may arise due to differences in the subsets $\mathcal{A}_k$. To illustrate this, consider a scenario where the data consists of two distinct clusters. Suppose a subset $\mathcal{A}_k$ is randomly sampled from one cluster, but the instance $\mathbf{x}$ being analyzed belongs to the other cluster. In this case, the Beran estimator trained on $\mathcal{A}_k$ may incorrectly associate $\mathbf{x}$ with the first cluster, as it lacks information about the second cluster. This can lead to an inaccurate prediction of the SF $S^{(i)}(t \mid \mathbf{x})$. The self-attention mechanism addresses this issue by incorporating contextual information from neighboring SFs. Specifically, each new SF $\widetilde{S}^{(i)}(t \mid \mathbf{x})$ is computed as a weighted sum of all SFs predicted by the $M-1$ other Beran estimators. The weights are determined by the distance between the SFs, ensuring that closer (more similar) SFs contribute more significantly to the final prediction. As a result, the adjusted SF $\widetilde{S}^{(i)}(t \mid \mathbf{x})$ can be viewed as a denoised and filtered version of the original prediction. This denoising property of self-attention has been highlighted in [66] as analogous to non-local means denoising, a technique designed to remove noise from data by leveraging contextual information.

After obtaining the denoised SFs $\widetilde{S}^{(i)}(t \mid \mathbf{x}, \theta)$, $i = 1, ..., M$, we can compute the aggregated SF of the ensemble as follows:

$$\widetilde{S}(t \mid \mathbf{x}, \theta) = \frac{1}{M} \sum_{k=1}^{M} \widetilde{S}^{(k)}(t \mid \mathbf{x}, \theta). \tag{8}$$

The next question is how to define the attention weights β. To address this, we first need to define a distance metric between two survival functions (SFs). A well-known distance metric for probability distributions is the Kolmogorov-Smirnov distance, which can be applied to a pair of SFs $S^{(k)}(t \mid \mathbf{x})$ and $S^{(l)}(t \mid \mathbf{x})$ as:

$$D_{KS}(S^{(k)}(t \mid \mathbf{x}), S^{(l)}(t \mid \mathbf{x})) = \sup_{t} \left| S^{(k)}(t \mid \mathbf{x}) - S^{(l)}(t \mid \mathbf{x}) \right|. \tag{9}$$

This distance measures the maximum absolute difference between the two SFs over time t, providing a way to quantify their dissimilarity.

By using the Gaussian kernel, the attention weight β is of the form:

$$\beta(S^{(l)}(t \mid \mathbf{x}), S^{(k)}(t \mid \mathbf{x}), \theta)$$
$$= \operatorname{softmax}\left(-\frac{D_{KS}(S^{(l)}(t \mid \mathbf{x}), S^{(k)}(t \mid \mathbf{x}))}{\theta_{kl}}\right), \tag{10}$$

where θ_{kl} is a hyperparameter or training parameter.

We have $M(M-1)$ parameters θ_{kl}, $k = 1, ..., M$, $l = 1, ..., M$, $k \neq l$. Hence, there holds $\theta = (\theta_{12}, \theta_{13}, ..., \theta_{(M-1)M})$.

The simplest way for applying the self-attention and computing the aggregated SF is to tune parameters τ and θ of kernels and to find the best ones which provide the largest C-index. However, we can extend the set of training parameters and maximize the C-index over these parameters.

By using the introduced notation $\mathcal{J}$ for the set of pairs (i, j), satisfying conditions $\delta_i = 1$ and $T_i < T_j$, we write the C-index as

$$C = \frac{1}{N} \sum_{(i,j)\in\mathcal{J}} \mathbf{1}[\widehat{T}_j - \widehat{T}_i > 0]$$
$$= \frac{1}{N} \sum_{(i,j)\in\mathcal{J}} \mathbf{1}\left[\sum_{k=1}^{M} \widehat{T}_j^{(k)} - \sum_{k=1}^{M} \widehat{T}_i^{(k)} > 0\right]. \tag{11}$$

Here $\widehat{T}_i^{(k)}$ is the expected time predicted by the k-th Beran estimator for the i-th object. Since there is a finite number of objects in the dataset, then the SF is step-wised. Hence, the conditional SF predicted by the k-th Beran estimator is

$$S^{(k)}(t \mid \mathbf{x}_i) = \sum_{l=0}^{N_k - 1} S_l^{(k)}(\mathbf{x}_i) \cdot \mathbf{1}\{t \in [t_l^{(k)}, t_{l+1}^{(k)})\}, \tag{12}$$

where $S_l^{(k)}(\mathbf{x}_i) = S^{(k)}(t_l \mid \mathbf{x}_i)$ is the SF in the time interval $[t_l^{(k)}, t_{l+1}^{(k)})$; $S_0^{(k)} = 1$ by $t_0 = 0$; N_k is the number of elements in $\mathcal{A}_k$; $t_1^{(k)} < t_2^{(k)} < ...t_{N_k}^{(k)}$ are ordered times to events from $\mathcal{A}_k$.

Hence, $\widehat{T}_i^{(k)}$ is determined as

$$\widehat{T}_i^{(k)} = \sum_{l=0}^{N_k - 1} S_l^{(k)}(\mathbf{x}_i)(t_{l+1}^{(k)} - t_l^{(k)}). \tag{13}$$

Since $t_1^{(k)}, t_2^{(k)}, ..., t_{N_k}^{(k)}$ is a part of all event times, then we can use all times t_l, $l = 1, ..., N$, but if a time $t_l \in [t_{l-1}, t_{l+1}]$ does not belong to the k-th bootstrap subset of the event times, then $S_l^{(k)}(\mathbf{x}_i) = S_{l-1}^{(k)}(\mathbf{x}_i)$. It follows from (7) and from the above definition of $\widehat{T}_i^{(k)}$ that there holds

$$\widehat{T}_i^{(j)} = \sum_{k=1, k\neq j}^{M} \beta\left(S^{(j)}(t \mid \mathbf{x}_i), S^{(k)}(t \mid \mathbf{x}_i), \theta\right) \cdot \widehat{T}_i^{(k)}. \tag{14}$$

Hence, the C-index is determined as

$$
\begin{aligned}
C &= \frac{1}{N}\sum_{(i,j)\in\mathcal{J}} \mathbf{1}[\widehat{T}_j - \widehat{T}_i > 0] \\
&= \frac{1}{N}\sum_{(i,j)\in\mathcal{J}} \mathbf{1}\left[\begin{array}{c} \sum_{l=1}^{M}\sum_{k=1,k\neq l}^{M} \beta\left(S^{(l)}(t \mid \mathbf{x}_j), S^{(k)}(t \mid \mathbf{x}_j), \theta\right)\cdot \widehat{T}_j^{(l)} \\ -\sum_{l=1}^{M}\sum_{k=1,k\neq l}^{M} \beta\left(S^{(l)}(t \mid \mathbf{x}_i), S^{(k)}(t \mid \mathbf{x}_i), \theta\right)\cdot \widehat{T}_i^{(l)} > 0 \end{array}\right]. \qquad (15)
\end{aligned}
$$

Let us denote

$$
\beta_j^{(l,k)}(\theta) = \beta\left(S^{(l)}(t \mid \mathbf{x}_j), S^{(k)}(t \mid \mathbf{x}_j), \theta\right), \qquad (16)
$$

$$
R_{ij}(\theta) = \sum_{l=1}^{M}\sum_{k=1,k\neq l}^{M}\left(\beta_j^{(l,k)}(\theta)\widehat{T}_j^{(k)} - \beta_i^{(l,k)}(\theta)\widehat{T}_i^{(k)}\right). \qquad (17)
$$

Then there holds

$$
C = \frac{1}{N}\sum_{(i,j)\in\mathcal{J}} \mathbf{1}\left[R_{ij}(\theta) > 0\right]. \qquad (18)
$$

It is proposed to replace the indicator function in the above C-index with the sigmoid function σ. Hence, optimal parameters θ of the self-attention are obtained by solving the following optimization problem:

$$
\max_{\theta} \sum_{(i,j)\in\mathcal{J}} \sigma\left(R_{ij}(\theta)\right). \qquad (19)
$$

Parameters θ as well as the parameter of the Beran estimator τ can be obtained by solving the above optimization problem by means of gradient-based algorithms. Moreover, the parameter τ can be trained for each base learner, i.e., we can have M parameters $\tau_1, ..., \tau_M$ such that each parameter defines the corresponding Beran estimator. Finally, the aggregation (8) can also be replaced with the Nadaraya-Watson kernel regression. However, this replacement may significantly complicate the optimization problem, therefore, it is not considered in this work.

3.3 An important special case

If we suppose that η is the hyperparameter which is identical for all base learners, then the optimization problem for computing parameters θ, can significantly be simplified. Let us use the definition of the attention weight $\beta_i^{(k,j)}(\theta)$ in a form proposed in [27]:

$$
\begin{aligned}
\beta_j^{(l,k)}(\theta) &= (1-\epsilon)\cdot \text{softmax}\left(D_j^{(k,l)}(\varphi)\right) + \epsilon\cdot\theta_{l,k}, \\
k &= 1, ..., M, \qquad (20)
\end{aligned}
$$

where φ is the tuning parameter; $\theta_{1,1}, ..., \theta_{M,M}$ are training parameters such that $\theta \in \Delta^{M\times M}$;

$$
D_j^{(k,l)}(\varphi) = -\frac{D_{KS}(S^{(k)}(T_j \mid \mathbf{x}_j), S^{(l)}(T_j \mid \mathbf{x}_j))}{\varphi}.
$$

The above expression is derived from the imprecise Huber's ϵ-contamination model [25], which is represented as

$$
F = (1-\epsilon)\cdot P + \epsilon\cdot\Theta, \qquad (21)
$$

where P is a discrete probability distribution contaminated by another probability distribution Θ, which is arbitrary within the unit simplex $\Delta^{M\times M\prime}$; components of Θ satisfy the conditions $\theta_{1,1}+ ... +\theta_{M,M}=1$ and $\theta_{l,k}\geq 0$ for all $k=1,...,M$, $l=1,...,M$; the contamination parameter $\epsilon\in[0,1]$ controls the size of the small simplex generated by the ϵ-contamination model. In particular, if $\epsilon=0$, then the model reduces to the softmax operation with hyperparameter φ, and the attention weight becomes independent of $\theta_{l,k}$. The distribution P consists of elements $\text{softmax}\Big(D_j^{(k,l)}(\varphi)\Big)$, $k=1,...,M$, $l=1,...,M$. The distribution Θ defines the vector $\theta=(\theta_{1,1},...,\theta_{M,M})$, which represents the training parameters of the attention mechanism.

Let us replace the indicator function in (18) with the hinge loss function $\max(0,x)$ similarly to the replacement proposed by Van Belle et al. [67]. By adding the regularization term $\|\theta\|^2 = \sum_{l=1,}^{M}\sum_{k=1,k\neq l}^{M}\theta_{l,k}^2$ with the hyperparameter λ which controls the strength of the regularization, the optimization problem can be written as

$$\min_{\theta_1\in\Delta^M,...,\theta_M\in\Delta^M}\left\{\sum_{(i,j)\in\mathcal{J}}\max\left(0,R_{ij}(\theta)\right)+\lambda\|\theta\|^2\right\}. \tag{22}$$

Here θ_j is the vector of M variables $\theta_{j,1},...,\theta_{j,M}$. Let us introduce the variables

$$\xi_{ij}=\max\left(0,R_{ij}(\theta)\right). \tag{23}$$

The optimization problem can be written in the following form:

$$\min_{\theta_1\in\Delta^M,...,\theta_M\in\Delta^M}\left\{\sum_{(i,j)\in\mathcal{J}}\xi_{ij}+\lambda\|\theta\|^2\right\}, \tag{24}$$

subject to $\theta\in\Delta^M$ and

$$\xi_{ij}\geq R_{ij}(\theta),\quad \xi_{ij}\geq 0,\quad \{i,j\}\in\mathcal{J}. \tag{25}$$

Let us consider how $R_{ij}(\theta)$ depends on θ in this special case. Denote

$$\begin{aligned}Q_{i,j}^{(k,l)}(\varphi,\epsilon)&=(1-\epsilon)\cdot S^{(k)}(T_j\mid\mathbf{x}_j)\cdot\text{softmax}\left(D_j^{(k,l)}(\varphi)\right)\\&-(1-\epsilon)\cdot S^{(k)}(T_i\mid\mathbf{x}_i)\cdot\text{softmax}\left(D_i^{(k,l)}(\varphi)\right),\end{aligned} \tag{26}$$

and

$$G_{i,j}^{(k)}(\epsilon)=\epsilon\cdot\left(S^{(k)}(T_j\mid\mathbf{x}_j)-S^{(k)}(T_i\mid\mathbf{x}_i)\right). \tag{27}$$

It follows from (17) that

$$R_{ij}(\theta)=\sum_{l=1,}^{M}\sum_{k=1,k\neq l}^{M}\left(Q_{i,j}^{(k,l)}(\varphi,\epsilon)+\theta_{l,k}\cdot G_{i,j}^{(k)}(\epsilon)\right). \tag{28}$$

It can be seen from (28) that the constraints are linear with respect to training parameters θ and training parameters ξ_{ij}. This implies that the obtained optimization problem is quadratic with linear constraints and has variables ξ_{ij} and θ.

Numerical experiments

In numerical experiments, the properties of the proposed SurvBESA model, trained on synthetic and real data, are investigated. In synthetic experiments, various models are compared, such as the single Beran model ("Single Beran"), the classical bagging of Beran models with simple averaging of the predictions of weak models ("Bagging"), and the SurvBESA model, which uses the self-attention mechanism, Huber ϵ-contamination model, and optimization. For real data, the models mentioned above are supplemented with GBM Cox, GBM AFT, and RSF. The performance measure for studying and comparing the models is the C-index, calculated on the test set. To evaluate the C-index for each dataset, cross-validation is performed. Instances for training, validation, and testing are randomly selected in each run. Hyperparameters are tuned using the Optuna library.

3.4 Synthetic data

Training instances $\mathbf{x} \in \mathbb{R}^5$ are generated randomly based on the uniform distribution within the hyperrectangle $\prod_{j=1}^{5}[a^{(j)}, b^{(j)}]$. Censoring indicators δ are generated randomly according to the Bernoulli distribution with probabilities $\Pr\{\delta = 0\} = 1 - p$ and $\Pr\{\delta = 1\} = p$. The number of instances N in the training set along with the Bernoulli distribution parameter are varied to study their impact on the model performance. Event times are generated according to the Weibull distribution with the shape parameter k. The event times depend on $\mathbf{x}$ through the following relationship:

$$T = \frac{\sin\left(c \cdot \sum_{i=1}^{5} x^{(i)}\right) + c}{\Gamma(1 + 1/k)} \cdot (-\log(u))^{1/k},$$

where c is a parameter, and u is a random variable uniformly distributed on the interval $[0, 1]$, $x^{(i)}$ is the i-th feature of $\mathbf{x}$.

To make the synthetic data more complex, the expected value of T is modified according to the Weibull distribution as $\sin\left(c \cdot \sum_{i=1}^{5} x^{(i)}\right) + c$. In other words, the mean event time varies as a sinusoidal function. The term $\sum_{i=1}^{5} x^{(i)}$ is used to model feature interactions.

Sets of the hyperparameter values for different models are as follows:

- **SurvBESA**: $\epsilon \in [0, 1]$; $w \in \{10^{-3}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}, 10^{3}\}$; $\tau \in \{10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}, 10^{3}\}$; the subset size varies in the range $[0.1, 0.7]$; the number of the Beran estimators is in the interval $[5, 50]$; the gradient descent step size is from the set$\{10^{-3}, 10^{-2}, 10^{-1}\}$.
- **Bagging**: $\tau \in \{10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}, 10^{3}\}$; the subset size varies in the range $[0.1, 0.7]$; the number of the Beran estimators is in $[5, 50]$.
- **Single Beran**: $\tau \in \{10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}, 10^{3}\}$.

These hyperparameters, if not the subject of study, are determined using the Optuna library [68]. Other hyperparameters are tested manually, selecting those that yield the best results.

To investigate how the C-index depends on various parameters, corresponding experiments are conducted. In all experiments for SurvBESA, the training parameters are optimized using Adam with 100 epochs. The initial parameters of the synthetic data are: $c = 3$, $k = 6$, $a^{(j)} = -2.0$, $b^{(j)} = 5$, $j = 2, \dots, d$. The initial number of points N in the training set is 200, in the validation set is 100, and

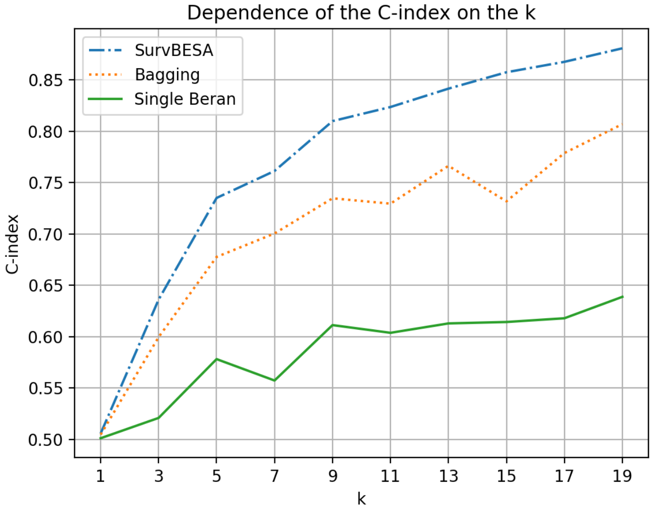

Figure 1: Dependence of the C-index on the parameter k

in the test set is 100, with the proportion of censored data being 0.2. Some of the above parameters are varied in the experiments to study their impact on the resulting C-index. For each value of the investigated hyperparameters of the models and data generation parameters, 25 experiments were conducted, and their results were averaged.

Dependence of the C-index on the parameter k: The parameter k is varied in the set $[1, 19]$ with a step size of 2. The study examined how the C-index depends on the parameter k of the Weibull distribution. In the case where $k = 1$, an exponential distribution is used for generation. If $k = 2$, a Rayleigh distribution is used. The larger the parameter k, the less noise in the data. This can be observed in Fig. 1, which illustrates the dependence of the C-index on the parameter k for different models. The C-index increases with increasing k, which is the expected behavior. At the same time, the C-index for the SurvBESA model is higher for all values of k compared to the Bagging and Single Beran models. Additionally, the C-index for the Bagging model is higher than that of the single Beran model.

Dependence of the C-index on the number of points: The next question is how the number of points in the training set impact the model accuracy (the C-index). The correpsonding results are depicted in Fig. 2.

Dependence of the C-index on the proportion of uncensored data: Ten values are chosen for the proportion of uncensored data p, starting from 0.1 and ending at 0.9, with equal steps. This experiment is necessary to study the ability of models to learn in the presence of a large amount of censored data, which is often encountered in real-world datasets. An interesting observation is that as the proportion of uncensored data increases, the C-index values for all models decrease. This can be observed in Fig. 3.

Dependence of the C-index on the number of weak models: The number of weak models is varied in the set $[1, 31]$ with a step size of 4, while the subset size is fixed at 0.4 and not tuned. The case where the number of weak models equals 1 is equivalent to the Single Beran model. The dependence

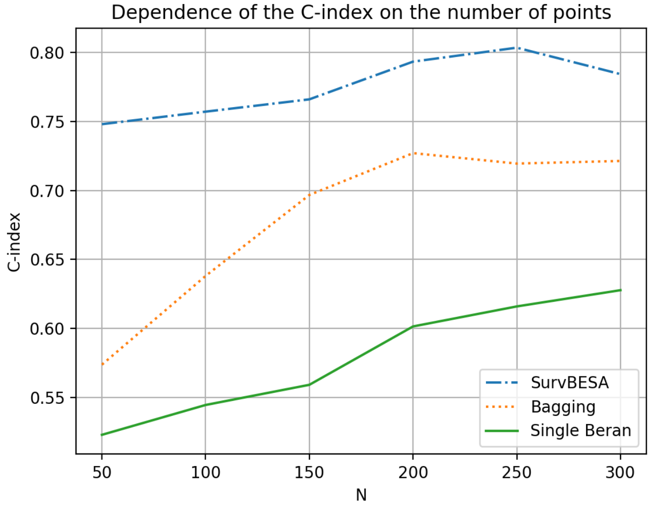

Figure 2: Dependence of the C-index on the number of points in the training set

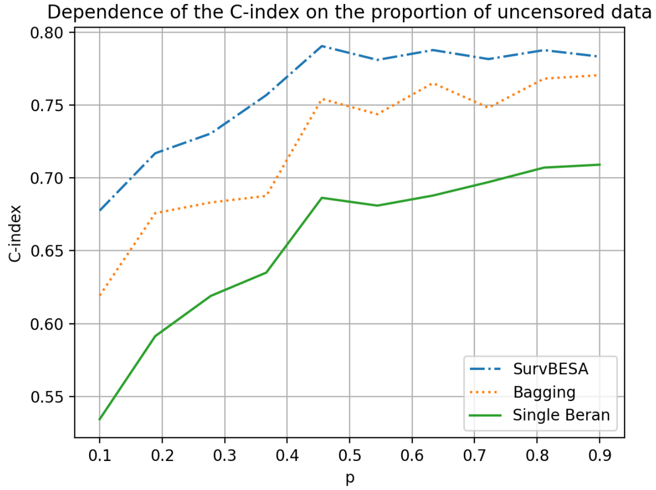

Figure 3: Dependence of the C-index on the proportion of uncensored data

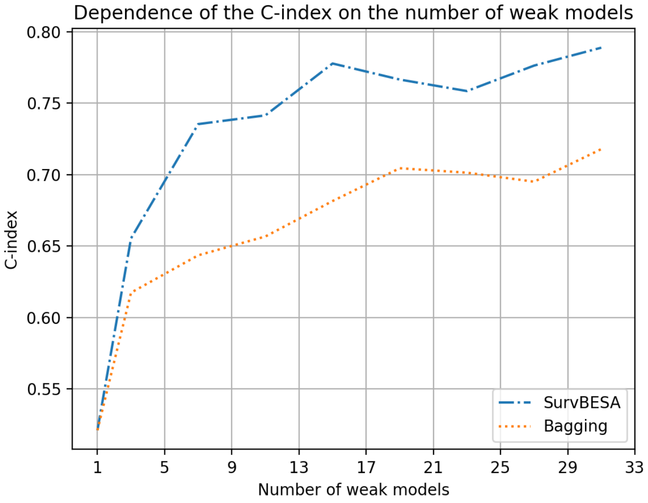

Figure 4: Dependence of the C-index on the number of weak models

is shown in Fig. 4. It can be observed that for both models, the C-index increases with the number of weak models, which aligns with the core idea of bagging.

Dependence of the C-index on the subset size: The subset size is varied in the range $[0.1, 0.9]$. The number of weak models is fixed at 25. The dependence is shown in Fig. 5. From the graph, it can be seen that the C-index increases with the subset size and then plateaus. This can be attributed to the fact that the parameter τ is tuned using the training data that is not included in the Beran model. Additionally, the random subspace partitioning used in building the ensemble may play a significant role.

Dependence of the C-index on the parameter c: Ten values are chosen for the parameter c, starting from 0.1 and ending at 6, with equal steps. The larger the parameter c, the "faster" the expected time changes, making its accurate prediction more challenging. This is consistent with the results shown in Fig. 6. The C-index decreases as the parameter c increases. It is also worth noting that the more complex the function relating the expected time to the features $\mathbf{x}$, the greater the difference between SurvBESA and the other models. In other words, as the complexity of the synthetic dataset increases, the C-index for the SurvBESA model decreases more slowly compared to the other models.

3.5 Real data

To compare SurvBESA with various survival models on real data, the models described above are used. They are tested on the following real benchmark datasets: *Veterans*: 137 examples, 6 features; *AIDS*: 2139 examples, 23 features; *Breast Cancer*: 198 examples, 80 features; *WHAS500*: 500 examples, 14 features; *PBC*: 418 examples, 17 features; *WPBC*: 198 examples, 34 features; *HTD*: 69 examples, 9 features; *CML*: 507 examples, 7 features; *Rossi*: 432 examples, 62 features; *Lung Cancer*: 228 examples, 9 features.

To evaluate the results, 100 iterations of different splits of the dataset into training, validation, and

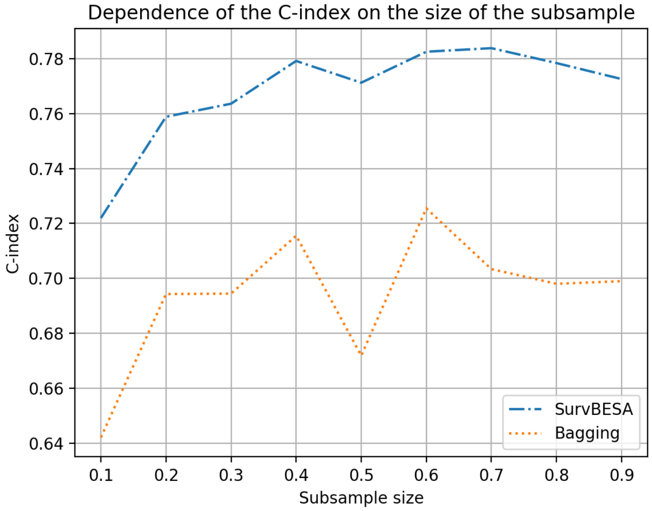

Figure 5: Dependence of the C-index on the subset size

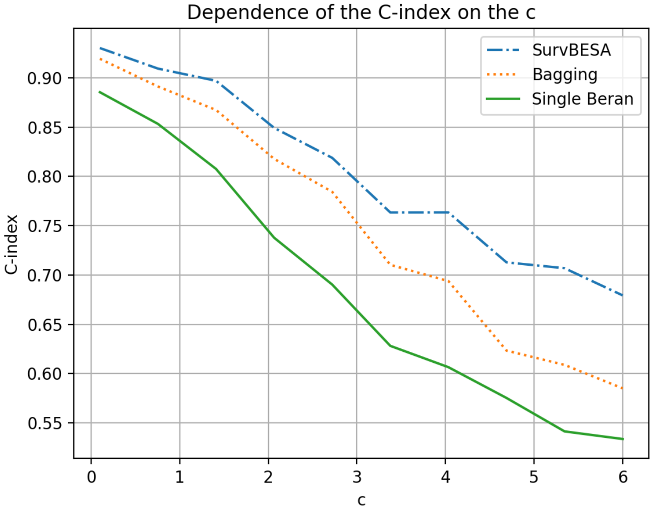

Figure 6: Dependence of the C-index on the parameter c

test sets are performed. The training set contains 60% of the examples, the validation set contains 20%, and the test set contains 20%. Hyperparameters are tuned on the validation set, and the C-index is measured on the test set. The average C-index values across different iterations are the final results used for comparison.

SurvBESA is implemented using Python software. The corresponding software is available at: `https://github.com/NTAILab/SurvBESA`.

Hyperparameters for comparison of models:

- *SurvBESA*: $\epsilon \in [0,1]$; $w \in \{10^{-3}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}, 10^{3}\}$; $\tau \in \{10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}, 10^{3}\}$; the subset size varies in the range $[0.1, 0.7]$; the number of estimators is in $[5, 50]$; the gradient descent step size is in $\{10^{-3}, 10^{-2}, 10^{-1}\}$.

- *Bagging*: $\tau \in \{10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}, 10^{3}\}$; the subset size varies in the range $[0.1, 0.7]$; the number of estimators is in $[5, 50]$.

- *Single Beran*: $\tau \in \{10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}, 10^{3}\}$.

- *RSF*: the number of weak estimators varies in the range $[20, 300]$; the maximum tree depth varies in the range $[2, 15]$; the minimum number of samples in a leaf node varies in the range $[2, 15]$.

- *GBM Cox*: the gradient descent step size varies in $\{10^{-3}, 10^{-2}, 10^{-1}\}$; the number of iterations varies in the range $[20, 300]$; the maximum tree depth varies in the range $[2, 10]$.

- *GBM AFT*: the gradient descent step size varies in $\{10^{-3}, 10^{-2}, 10^{-1}\}$; the number of iterations varies in the range $[20, 300]$; the maximum tree depth varies in the range $[2, 10]$.

The comparison results of different models for the above datasets are presented in Table 1 where the best results for each dataset are highlighted in bold. It can be observed that SurvBESA outperforms other models on most datasets. It is also important to note that the bagging Beran models perform better than a single Beran model but does not always outperform RSF, GBM Cox, and GBM AFT models. Although the results of SurvBESA are better than those of other models, SurvBESA is significantly more computationally complex, which may be important in some applications.

To formally demonstrate that SurvBESA surpasses other methods, we employ the t-test, as proposed by Demsar [69], to assess whether the mean difference in performance between two classifiers is significantly different from zero. In this case, the t statistic follows the Student's t-distribution with $10-1$ degrees of freedom. The computed t statistics yield the corresponding p-value. The first sample consists of the C-indices obtained from SurvBESA (see Table 1) whereas the second sample consists of the best results provided by other models. The test show that SurvBESA significantly outperforms other analyzed models, as the corresponding p-value is equal to 0.0017, i.e., it is smaller than 0.05.

3.6 Additional experiments with real data

Figs. 7-9 demonstrate the gradient descent training process of the SurvBESA model. The datasets used are Veteran, AIDS, and Breast Cancer. The graphs show the change in the C-index with each epoch for the training and test sets, as well as the approximation of the C-index using the sigmoid function proposed in this work. The C-index approximation serves as the loss function, which is optimized using the Adam method with the learning rate 0.1. It is evident that with each iteration,

Table 1: Results of comparison using the C-index metric for methods: ensemble of Beran models with simple averaging, single Beran model, SurvBESA, RSF, GBM Cox, and GBM AFT on various datasets

Dataset	SurvBESA	Bagging	Single Beran	RSF	GBM Cox	GBM AFT
Veteran	**0.7414**	0.6835	0.6346	0.7001	0.6821	0.7004
AIDS	**0.7686**	0.7104	0.6466	0.7339	0.7198	0.5800
Breast Cancer	**0.7317**	0.6706	0.6595	0.6806	0.7021	0.6979
WHAS500	**0.7730**	0.7374	0.7181	0.7610	0.7528	0.7492
WPBC	**0.7652**	0.6896	0.6495	0.6387	0.6159	0.6078
HTD	**0.8315**	0.7864	0.7421	0.7809	0.7749	0.7932
Lung	**0.6875**	0.6430	0.6256	0.6115	0.5799	0.5588
Rossi	**0.6243**	0.5863	0.5245	0.5182	0.5117	0.5380
PBC	0.8152	0.7712	0.7619	0.8099	**0.8153**	0.7969
CML	**0.7149**	0.7050	0.7009	0.7129	0.7136	0.7114

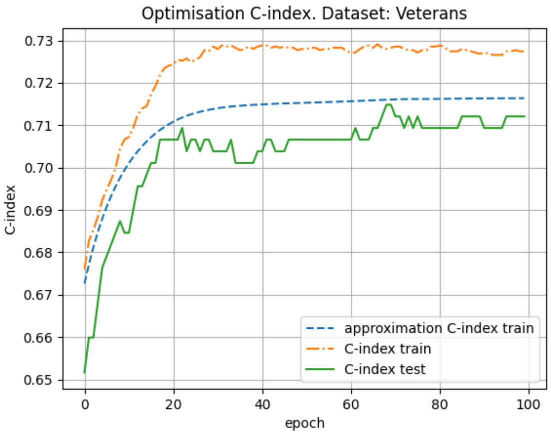

Figure 7: Illustration of the SurvBESA training process for the Veteran dataset

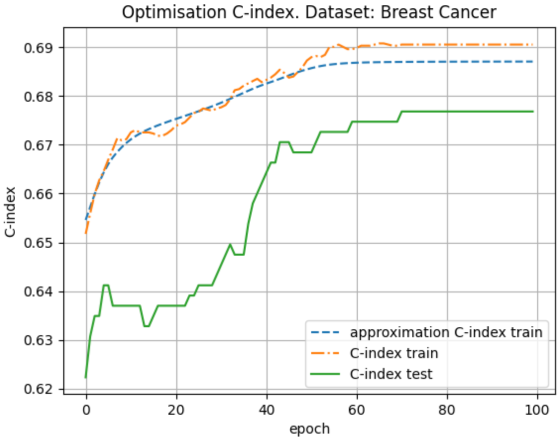

Figure 8: Illustration of the SurvBESA training process for the Breast Cancer dataset

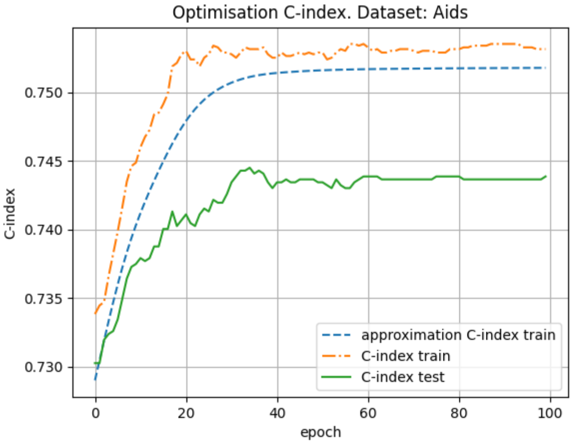

Figure 9: Illustration of the SurvBESA training process for the AIDS dataset

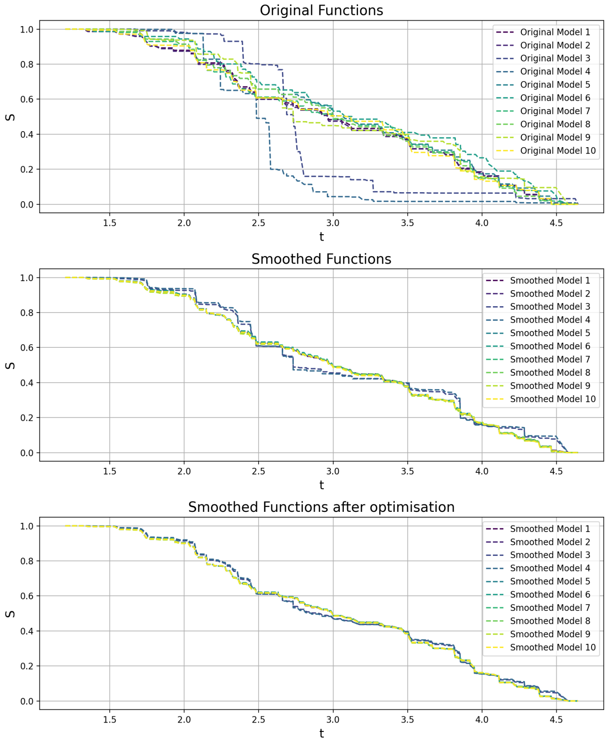

Figure 10: An illustrative example of the SF transformation produced by the weak Beran models using the self-attention mechanism

the C-index for both the training and test sets increases and eventually plateaus, indicating the model's ability to learn.

Fig. 10 shows the change in the SFs obtained from the Beran models as a result of using the self-attention mechanism and optimization. It can be observed that initially, some Beran models (models 3 and 4 in the first picture in Fig. 10) produce SFs that significantly differ from the others and can be viewed as outliers, leading to a certain bias in the final predicted result. After applying self-attention (the second picture in Fig. 10) and optimization (the third picture in Fig. 10), the SFs are transformed into a more general form.

4 Conclusion

In this paper, a novel ensemble-based survival model called SurvBESA has been introduced. It leverages the Beran estimator as its weak learner and incorporates the self-attention mechanism to aggregate predictions from multiple base models. The proposed model addresses key challenges in survival analysis, such as handling censored data and improving the robustness of predictions in the presence of multi-cluster data structures. By applying self-attention to the survival functions predicted by individual Beran estimators, SurvBESA effectively reduces noise and stabilizes predictions, leading to more accurate and reliable survival estimates.

Extensive numerical experiments on both synthetic and real-world datasets demonstrate the superiority of SurvBESA over traditional survival models, including Random Survival Forests (RSF), Gradient Boosting Machines (GBM) with Cox and AFT loss functions, and standalone Beran estimators. The results highlight SurvBESA's ability to adapt to complex data structures and its robustness in scenarios with high levels of censoring. Furthermore, the model's performance is consistently strong across various datasets, making it a versatile tool for survival analysis tasks.

A key contribution of this work is the integration of the Huber ϵ-contamination model into the self-attention framework, which simplifies the training process by reducing it to a quadratic or linear optimization problem. This use of the ϵ-contamination model not only enhances computational efficiency, but also provides a principled way to handle uncertainty in the predictions of weak learners.

The implementation of SurvBESA is publicly available, enabling researchers and practitioners to apply and extend the model to their own survival analysis problems. Future work could explore further optimizations of the self-attention mechanism, extensions to other types of weak learners, and applications to larger and more diverse datasets.

In conclusion, SurvBESA represents a significant advancement in ensemble-based survival analysis, offering a robust, accurate, and generalizable framework for predicting time-to-event outcomes in the presence of censored data. Its ability to leverage the strengths of the Beran estimator while mitigating its limitations through self-attention makes it a promising tool for both research and practical applications in survival analysis.

While the Beran estimator has proven effective as a weak learner, investigating the integration of other types of weak learners into the SurvBESA framework could broaden its applicability. For instance, incorporating neural network-based survival models or other non-parametric estimators could provide additional flexibility and performance gains.

Future research could focus on refining the self-attention mechanism within SurvBESA to further enhance its ability to capture dependencies among survival function predictions. This could involve exploring alternative attention architectures, such as multi-head attention or sparse attention, to improve computational efficiency and prediction accuracy.

5 Acknowledgement

The research is partially funded by the Ministry of Science and Higher Education of the Russian Federation as part of state assignments "Development and research of machine learning models for solving fundamental problems of artificial intelligence for the fuel and energy complex" (topic FSEG-2024-0027).

References

[1] P. Wang, Y. Li, and C.K. Reddy. Machine learning for survival analysis: A survey. *ACM Computing Surveys (CSUR)*, 51(6):1–36, 2019.

[2] M.Z. Nezhad, N. Sadati, K. Yang, and D. Zhu. A deep active survival analysis approach for precision treatment recommendations: Application of prostate cancer. arXiv:1804.03280v1, April 2018.

[3] N.A. Ibrahim, A. Kudus, I. Daud, and M.R. Abu Bakar. Decision tree for competing risks survival probability in breast cancer study. *International Journal Of Biological and Medical Research*, 3(1):25–29, 2008.

[4] U.B. Mogensen, H. Ishwaran, and T.A. Gerds. Evaluating random forests for survival analysis using prediction error curves. *Journal of Statistical Software*, 50(11):1–23, 2012.

[5] M. Schmid, M.N. Wright, and A. Ziegler. On the use of harrell's c for clinical risk prediction via random survival forests. *Expert Systems with Applications*, 63:450–459, 2016.

[6] H. Wang and L. Zhou. Random survival forest with space extensions for censored data. *Artificial intelligence in medicine*, 79:52–61, 2017.

[7] L.V. Utkin, A.V. Konstantinov, V.S. Chukanov, M.V. Kots, M.A. Ryabinin, and A.A. Meldo. A weighted random survival forest. *Knowledge-Based Systems*, 177:136–144, 2019.

[8] L.V. Utkin and A.V. Konstantinov. Random survival forests incorporated by the nadaraya-watson regression. *Informatics and Automation*, 21(5):851–880, 2022.

[9] M.N. Wright, T. Dankowski, and A. Ziegler. Unbiased split variable selection for random survival forests using maximally selected rank statistics. *Statistics in Medicine*, 36(8):1272–1284, 2017.

[10] Yifei Chen, Zhenyu Jia, Dan Mercola, and Xiaohui Xie. A gradient boosting algorithm for survival analysis via direct optimization of concordance index. *Computational and mathematical methods in medicine*, 2013(1):873595, 2013.

[11] Pei Liu, Bo Fu, and Simon X Yang. Hitboost: survival analysis via a multi-output gradient boosting decision tree method. *IEEE Access*, 7:56785–56795, 2019.

[12] R. Meier, S. Graw, J. Usset, R. Raghavan, J. Dai, P. Chalise, S. Ellis, B. Fridley, and D. Koestler. An ensemble-based Cox proportional hazards regression framework for predicting survival in metastatic castration-resistant prostate cancer (mcrpc) patients. *F1000Research*, 5:2677, 2016.

[13] D.R. Cox. Regression models and life-tables. *Journal of the Royal Statistical Society, Series B (Methodological)*, 34(2):187–220, 1972.

[14] R. Beran. Nonparametric regression with randomly censored survival data. Technical report, University of California, Berkeley, 1981.

[15] Xibin Dong, Zhiwen Yu, Wenming Cao, Yifan Shi, and Qianli Ma. A survey on ensemble learning. *Frontiers of Computer Science*, 14:241–258, 2020.

[16] A.J. Ferreira and M.A.T. Figueiredo. Boosting algorithms: A review of methods, theory, and applications. In C. Zhang and Y. Ma, editors, *Ensemble Machine Learning: Methods and Applications*, pages 35–85. Springer, New York, 2012.

[17] Y. Ren, L. Zhang, and P. N. Suganthan. Ensemble classification and regression-recent developments, applications and future directions [review article]. *IEEE Computational Intelligence Magazine*, 11(1):41–53, 2016.

[18] O. Sagi and L. Rokach. Ensemble learning: A survey. *WIREs Data Mining and Knowledge Discovery*, 8(e1249):1–18, 2018.

[19] M. Wozniak, M. Grana, and E. Corchado. A survey of multiple classifier systems as hybrid systems. *Information Fusion*, pages 3–17, 2014.

[20] Z.-H. Zhou. *Ensemble Methods: Foundations and Algorithms.* CRC Press, Boca Raton, 2012.

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[22] Shi Hu, Egill Fridgeirsson, Guido van Wingen, and Max Welling. Transformer-based deep survival analysis. In *Survival Prediction-Algorithms, Challenges and Applications*, pages 132–148. PMLR, 2021.

[23] Zhihao Tang, Li Liu, Zongyi Chen, Guixiang Ma, Jiyan Dong, Xujie Sun, Xi Zhang, Chaozhuo Li, Qingfeng Zheng, Lin Yang, et al. Explainable survival analysis with uncertainty using convolution-involved vision transformer. *Computerized Medical Imaging and Graphics*, 110:102302, 2023.

[24] Zifeng Wang and Jimeng Sun. Survtrace: Transformers for survival analysis with competing events. In *Proceedings of the 13th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics*, pages 1–9, 2022.

[25] P.J. Huber. *Robust Statistics.* Wiley, New York, 1981.

[26] P. Walley. *Statistical Reasoning with Imprecise Probabilities.* Chapman and Hall, London, 1991.

[27] L.V. Utkin and A.V. Konstantinov. Attention-based random forest and contamination model. *Neural Networks*, 154:346–359, 2022.

[28] L.V. Utkin, A.V. Konstantinov, and S.R. Kirpichenko. Attention and self-attention in random forests. *Progress in Artificial Intelligence*, 12:257–273, 2023.

[29] G. Ridgeway. The state of boosting. *Computing science and statistics*, 31:172–181, 1999.

[30] A. Barnwal, H. Cho, and T. Hocking. Survival regression with accelerated failure time model in xgboost. *Journal of Computational and Graphical Statistics*, 31(4):1292–1302, 2022.

[31] Simon Wiegrebe, Philipp Kopper, Raphael Sonabend, Bernd Bischl, and Andreas Bender. Deep learning for survival analysis: a review. *Artificial Intelligence Review*, 57(65):1–34, 2024.

[32] J.L. Katzman, U. Shaham, A. Cloninger, J. Bates, T. Jiang, and Y. Kluger. Deepsurv: Personalized treatment recommender system using a Cox proportional hazards deep neural network. *BMC medical research methodology*, 18(24):1–12, 2018.

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

[34] J. Yao, X. Zhu, F. Zhu, and J. Huang. Deep correlational learning for survival prediction from multi-modality data. In *Medical Image Computing and Computer–Assisted Intervention – MICCAI 2017*, volume 10434 of *Lecture Notes in Computer Science*, pages 406–414. Springer, Cham, 2017.

[35] Kan Ren, Jiarui Qin, Lei Zheng, Zhengyu Yang, Weinan Zhang, Lin Qiu, and Yong Yu. Deep recurrent survival analysis. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 4798–4805, 2019.

[36] C. Haarburger, P. Weitz, O. Rippel, and D. Merhof. Image-based survival analysis for lung cancer patients using CNNs. arXiv:1808.09679v1, Aug 2018.

[37] T. Hothorn, P. Bühlmann, S. Dudoit, A. Molinaro, and M.J. van der Laan. Survival ensembles. *Biostatistics*, 7(3):355–373, 2006.

[38] J. Cheng, L. Dong, and M. Lapata. Long short-term memory-networks for machine reading. arXiv:1601.06733, Jan 2016.

[39] A. Parikh, O. Tackstrom, D. Das, and J. Uszkoreit. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249–2255. Association for Computational Linguistics, 2016.

[40] Z. Lin, M. Feng, C.N. dos Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio. A structured self-attentive sentence embedding. In *The 5th International Conference on Learning Representations (ICLR 2017)*, pages 1–15, 2017.

[41] J. Devlin, M.W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv:1810.04805, Oct 2018.

[42] F. Wu, A. Fan, A. Baevski, Y. Dauphin, and M. Auli. Pay less attention with lightweight and dynamic convolutions. In *International Conference on Learning Representations (ICLR 2019)*, pages 1–14, 2019.

[43] D. Povey, H. Hadian, P. Ghahremani, K. Li, and S. Khudanpur. A time-restricted self-attention layer for asr. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5874–5878. IEEE, 2018.

[44] K. Shim, J. Choi, and W. Sung. Understanding the role of self attention for efficient speech recognition. In *The Tenth International Conference on Learning Representations (ICLR)*, volume https://openreview.net/forum?id=AvcfxqRy4Y, pages 1–19, 2022.

[45] Z. Chen, L. Xie, J. Niu, X. Liu, and L. Wei. Joint self-attention and scale-aggregation for self-calibrated deraining network. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 2517–2525, 2020.

[46] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, and J. Uszkoreit. An image is worth 16x16 words: Transformers for image recognition at scale. arXiv:2010.11929, Oct 2020.

[47] M.-H. Guo, Z.-N. Liu, T.-J. Mu, and S.-M. Hu. Beyond self-attention: External attention using two linear layers for visual tasks. arXiv:2105.02358, May 2021.

[48] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021.

[49] Z. Shen, I. Bello, R. Vemulapalli, X. Jia, and C.H. Chen. Global self-attention networks for image recognition. arXiv:2010.03019, Oct 2020.

[50] D. Soydaner. Attention mechanism in neural networks: where it comes and where it goes. *Neural Computing and Applications*, 34(16):13371–13385, 2022.

[51] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang. Residual attention network for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3156–3164, 2017.

[52] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018.

[53] H. Zhao, J. Jia, and V. Koltun. Exploring self-attention for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10076–10085, 2020.

[54] G. Brauwers and F. Frasincar. A general survey on attention mechanisms in deep learning. *IEEE Transactions on Knowledge and Data Engineering*, 35(4):3279–3298, 2021.

[55] T. Lin, Y. Wang, X. Liu, and X. Qiu. A survey of transformers. arXiv:2106.04554, Jul 2021.

[56] T. Goncalves, I. Rio-Torto, L.F. Teixeira, and J.S. Cardoso. A survey on attention mechanisms for medical applications: are we moving towards better algorithms? arXiv:2204.12406, Apr 2022.

[57] M. Hassanin, S. Anwar, I. Radwan, F.S. Khan, and A. Mian. Visual attention methods in deep learning: An in-depth survey. *Information Fusion*, 108:102417, 2024.

[58] S. Khan, M. Naseer, M. Hayat, S.W. Zamir, F.S. Khan, and M. Shah. Transformers in vision: A survey. *ACM Computing Surveys (CSUR)*, pages 1–38, 2022.

[59] A. Santana and E. Colombini. Neural attention models in deep learning: Survey and taxonomy. arXiv:2112.05909, Dec 2021.

[60] D. Soydaner. Attention mechanism in neural networks: Where it comes and where it goes. arXiv:2204.13154, Apr 2022.

[61] Y. Xu, H. Wei, M. Lin, Y. Deng, K. Sheng, M. Zhang, F. Tang, W. Dong, F. Huang, and C. Xu. Transformers in computational visual media: A survey. *Computational Visual Media*, 8(1):33–62, 2022.

[62] Xulin Yang and Hang Qiu. Deep gated neural network with self-attention mechanism for survival analysis. *IEEE Journal of Biomedical and Health Informatics*, 2024.

[63] D. Hosmer, S. Lemeshow, and S. May. *Applied Survival Analysis: Regression Modeling of Time to Event Data*. John Wiley & Sons, New Jersey, 2008.

[64] F. Harrell, R. Califf, D. Pryor, K. Lee, and R. Rosati. Evaluating the yield of medical tests. *Journal of the American Medical Association*, 247:2543–2546, 1982.

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

[66] R. Vidal. Attention: Self-expression is all you need. ICLR 2022, OpenReview.net. https://openreview.net/forum?id=MmujBClawFo, 2022.

[67] V. Van Belle, K. Pelckmans, J.A.K. Suykens, and S. Van Huffel. Support vector machines for survival analysis. In *Proceedings of the Third International Conference on Computational Intelligence in Medicine and Healthcare (CIMED2007)*, pages 1–8, 2007.

[68] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2623–2631, 2019.

[69] J. Demsar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.