Survival Analysis: Deep Learning Is Not All You Need

Shaked Caspi

Statistics Department, Tel-Aviv University, Israel.

1 Introduction

During recent years, deep learning models have made significant advancements in handling unstructured data like photos, text, and audio, achieving state-of-the-art results [1–3]. However, for structured data, such as tabular data, classical machine learning models like XGBoost [4] continue to dominate, with deep learning models yet to surpass their performance [5]. Survival analysis, a statistical method used to analyze the time until an event of interest occurs, such as death, failure, or the occurrence of a specific outcome, offers valuable insights in various fields. It specifically accounts for censored data, where the event of interest may not have occurred for some individuals within the study period. Survival analysis problems, like tabular data problems, involve analyzing structured data. In this project, I aim to investigate whether deep learning models can outperform traditional methods in survival analysis. To do so, I have tested four different deep learning models and three machine learning models, comparing their results across five different benchmarks and two new datasets. All the experiments and results can be found here.

2 Models

2.1 ML Models

2.1.1 Regularized CoxPH

The regularized Cox proportional hazards (CoxPH) [6] model is an extension of the traditional Cox PH model that incorporates regularization techniques to improve model performance. Regularization methods, such as L1 (lasso), L2 (ridge), and elastic net, which is a combination of L1 and L2 regularization, can be applied to the Cox PH model. These methods aim to control model complexity and enhance prediction accuracy by shrinking or eliminating the coefficients of less informative variables. This process helps achieve better generalization and interpretability of the model while addressing issues related to high-dimensional covariates.

2.1.2 Random Survival Forest

The Random Survival Forest (RSF) [7] model is an ML approach that combines the concepts of random forests and survival analysis. RSF builds an ensemble of decision trees, where each tree is constructed using random subsets of the data and predictors. RSF use the log-rank splitting rule, which takes into account survival times and censoring status. Each tree in the RSF ensemble provides predictions for survival times and events, and the final survival predictions are obtained by aggregating the individual tree predictions. Instead of taking the average of end nodes, RSF combines survival predictions from individual trees using the Kaplan-Meier estimator to calculate survival probabilities over time. The estimated survival at time t is the average value across all trees in the ensemble at the specified time point. RSF offers advantages such as handling high-dimensional data, capturing nonlinear relationships, and providing variable importance measures.

2.1.3 Gradient Boosted Survival Trees

The Gradient Boosted Survival Model (GBSM) [8] is an ML approach that combines the principles of gradient boosting and survival analysis. GBSM sequentially builds an ensemble of weak prediction models, such as decision trees, by minimizing the negative log partial-likelihood of the survival data. The model incorporates gradient boosting techniques to optimize the model's performance and handle censored data effectively. Because I used the loss function of Cox's proportional hazards model, the predictions can be interpreted as the log hazard ratio, similar to the linear predictor of a Cox proportional hazards model.

2.2 DL Models

2.2.1 DeepSurv

DeepSurv model [9] is a multi-layer perceptron designed to predict the risk of experiencing an event based on input features. It builds on the CoxPH model, where the hazard function $h(t|x)$ is represented as the product of a baseline hazard function $h_0(t|x)$ and a model-specific risk score $\exp(g(x))$. In the traditional CoxPH model $g(x)$ is defined as $\beta^T x$. However, in DeepSurv, a deep neural network is utilized to learn the function $g(x)$ allowing for non-linear connections and more complex representations. The model is trained by optimizing the negative log partial-likelihood of the survival data, enabling estimation of both the coefficients β and the neural network parameters. By estimating the hazard function, DeepSurv predicts the conditional risk of an event occurring at different time points, given the input features.

2.2.2 DeepHit

DeepHit model [10] is a known discrete time DL model that directly learns the distribution of the discrete survival times using a deep neural network. The parameterization of the survival time to intervals enables us to predict for each interval the estimated probability of an event occurring and the expected time-to-event. The model is trained by minimizing a loss function that combines two losses, $\mathcal{L}_1$ and $\mathcal{L}_2$.

- $\mathcal{L}_1$ represents the log-likelihood of the joint distribution of the first hitting time and event, helping the network learn general representations.
- $\mathcal{L}_2$ incorporates cause-specific ranking loss functions to address competing risk situations.

DeepHit's advantages lie in capturing complex relationships between covariates and risk, without assuming the proportional hazard assumption unlike othert models like DeepSurv.

2.2.3 Cox-Time

Cox-Time model [11] is a deep learning approach that builds upon the traditional Cox model by eliminating the proportionality assumption. In contrast to DeepSurv's parameterization of $g(x)$, the Cox-Time model employs a neural network to parameterize $g(t, x)$, enabling the relative risk function to vary with time and other covariates. This enhancement allows the model to capture interactions between time and predictors. Despite this increased flexibility, the Cox-Time model maintains its relative risk nature and benefits from using the same partial likelihood during training.

2.2.4 Piecewise constant Hazard (PC-Hazard)

PC-Hazard model [12] is a DL approach that assumes the continuous-time hazard rate is piecewise constant. It achieves this by partitioning time into intervals and considering piecewise constant hazards within each interval. The loss function for the PC-Hazard model is the mean negative log-likelihood, where the network learns a representation for the hazard (or, more precisely, a linear transformation of the hazard for convenience). Similar to DeepHit, the PC-Hazard model avoids making assumptions, making it valuable in scenarios where specific assumptions may not hold.

3 Benchmarks

In this section, I provide a detailed description of the benchmarks used in my experiments, following a systematic selection process. To ensure comprehensive coverage, I collected datasets used in articles that featured the selected DL models: METABRIC [9], SUPPORT [9], GBSG [9], FLCHAIN [11], and NWTCO [13]. The datasets were then ranked based on how frequently they were used across the DL model papers. The dataset information is presented in Table 1, and the experiments were conducted in the order of the dataset rankings, starting from the top-ranked dataset (rank #1).

It's important to note that all the datasets selected for my experiments are right-censored only and represent single-risk type datasets, focusing on a specific event of interest. This consistency in risk type facilitates fair and reliable comparisons across the models. In addition to the ranked datasets, I also included two additional datasets: prostateSurvival [14] and hepatoCellular [11]. These datasets had not been used in any of the referenced papers, ensuring novel test cases. Each of the datasets has a unique property that makes it interesting for testing. prostateSurvival contains more than 10,000 samples, which is more than any of the other datasets. hepatoCellular

contains more than 40 features, again surpassing any of the other datasets. Since these datasets don't have a rank due to their unique nature, I incorporated them into my experimentation to further evaluate the performance of the models.

Table 1 Datasets information and ranking

Dataset Name	n[a]	p[b]	Censored (%)	Rank
METABRIC	1,904	9	42.06	1
SUPPORT	8,873	14	31.97	1
GBSG	2,232	7	43.23	2
FLCHAIN	6,524	8	69.92	2
NWTCO	4,028	6	85.82	3
prostateSurvival	11,054	3	92.77	-
hepatoCellular	227	43	57.26	-

[a]the number of observations in the dataset
[b]the number of features in the dataset

4 Evaluation Metrics

4.1 Concordance Indices (C-index)

The Concordance Index C [15] evaluates the predictive performance of a survival model by comparing the ranking of survival times between individuals who experienced an event $(Y1, Y_2, ..., Y_n)$ and those who didn't $(V_1, V_2, ..., V_m)$ It assesses how accurately the model ranks these pairs $(Y_i > V_j)$ and provides an overall measure of the model's ability to discriminate between individuals based on their risk scores.

$$C = \frac{1}{n \cdot m} \sum_{i=1}^{n} \sum_{j=1}^{m} \mathbb{I}\{Y_i > V_j\} + \frac{1}{2}\mathbb{I}\{Y_i = V_j\}$$

It is important to note that this measure is a discrimination measure. Meaning it only test the model's ability to correctly provide a ranking of the survival times based on the individual risk scores.

4.2 Time Dependent Concordance Index

The Time-Dependent Concordance Index (C_{td}) [16] is a metric used to assess the predictive accuracy of survival models over time. Unlike the traditional C-index, which measures overall model performance based on the assumption that those who live longer should have lower risk, the C_{td} evaluates the model's ability to make accurate predictions at different time points during survival analysis. A higher C_{td} indicates better predictive accuracy, indicating that the model is capable of making more accurate predictions at various time intervals. Given the following settings

- $n + m = N$
- $conc_{ij}^{td} = \mathbb{I}\{S(t_i|x_i(t) < S(t_j|x_j(t))\} \cdot comp_{ij}$

- $comp_{ij} = \mathbb{I}\{t_i < t_j, \delta_i = 1\} + \mathbb{I}\{t_i = t_j, \delta_i = 1, \delta_j = 0\}$

The C^{td} define as

$$C_{td} = \frac{\sum_{i=1}^{N} \sum_{j=1;\, i \neq j}^{N} conc_{ij}^{td}}{\sum_{i=1}^{N} \sum_{j=1;\, i \neq j}^{N} comp_{ij}^{td}}$$

4.3 Integrated Brier Score

The Integrated Brier Score (IBS) [17] is a calibration metric used to evaluate the accuracy of predicted probabilities in survival analysis. It measures the mean squared difference between the predicted probabilities and the actual event occurrences at different time points. A lower IBS indicates better model performance. The IBS accounts for censored data by using the inverse probability of censoring estimations, calculated using the Kaplan-Meier method.

$$BS(t) = \frac{(0 - \hat{S}(t, x_i))^2 \cdot \mathbb{I}\{t_i \leq t, \delta_i = 1\}}{\hat{G}(t_i^-)} + \frac{(1 - \hat{S}(t, x_i))^2 \cdot \mathbb{I}\{t_i > t\}}{\hat{G}(t_i)}$$

$$IBS = \int_{t_{min}}^{t_{max}} IBS(t)dt$$

5 Experiments

5.1 Experiments Settings

I conducted a series of experiments using Python, evaluating five benchmarks and two additional datasets. For implementation, I utilized open-source packages, lifelines [18], Scikit-Survival [19], and pycox [13]. While I had hoped to test additional DL models using the Auton package [20], I encountered some instability in its implementation, so it was not used in the experiments. It's worth noting that compared to R, I found Python's support for DL models in survival analysis somewhat limited.

To validate the models, I employed 5-fold cross-validation and conducted a hyperparameter grid search. As a preprocessing step, all continuous features were standardized. I have made the code, relevant information, and all the results of the cross-validation available on the attached repository. The results include both the means and standard deviations of the performance metrics obtained during the cross-validation process. Due to size constraints on GitHub, I couldn't upload all the models, network structures, and weights. Additionally, before starting the experiments, I randomly divided each dataset into training (90%) and test (10%) sets. The training portion was used for the experiments, with 75% utilized for actual training, and the remaining 25% used for validation. The test dataset was reserved for evaluation purposes, and each model was tested using the same test dataset.

5.2 C Index Results

Fig. 1 C Index results - Benchmarks

	METABRIC	SUPPORT	GBSG	FLCHAIN	NWTCO
Reg CoxPH	0.6639	0.5902	0.6562	**0.7968**	0.6825
RSF	0.6643	0.6247	0.6599	0.795	0.6869
GBSM	0.6552	**0.6283**	0.6578	0.7922	**0.6993**
DeepSurv	**0.671**	0.6276	0.6561	0.7951	0.6943
DeepHit	0.5798	0.5173	0.5274	0.7341	0.5988
PC-Hazard	0.4927	0.4981	0.4168	0.2487	0.5135
CoxTime	0.6392	0.6098	**0.6629**	0.7839	0.6851

Fig. 2 C Index results - new datasets

	prostateSurvival	hepatoCellular
Reg CoxPH	0.7835	0.7375
RSF	0.777	0.7938
GBSM	0.7765	0.8
DeepSurv	0.7757	0.7562
DeepHit	**0.7836**	0.3625
PC-Hazard	0.2673	0.575
CoxTime	0.7743	**0.8125**

5.3 TD-Concordance Index Results

Fig. 3 TD-Concordance Index results - Benchmarks

	METABRIC	SUPPORT	GBSG	FLCHAIN	NWTCO
Reg CoxPH	0.6639	0.5902	0.6562	**0.7968**	0.6823
RSF	0.659	**0.6428**	0.6595	0.7898	0.7022
GBSM	0.6552	0.6283	0.6578	0.7919	0.6953
DeepSurv	**0.6717**	0.6276	0.6561	0.7951	0.694
DeepHit	0.6038	0.6195	**0.6669**	0.7851	**0.7135**
PC-Hazard	0.4976	0.4731	0.5551	0.3158	0.395
CoxTime	0.6577	0.6159	0.6579	0.7633	0.6856

Fig. 4 TD-Concordance Index results - new datasets

	prostateSurvival	hepatoCellular
Reg CoxPH	0.7586	0.7375
RSF	0.758	0.7938
GBSM	0.7516	0.8
DeepSurv	**0.766**	0.7562
DeepHit	0.7603	0.7625
PC-Hazard	0.6062	0.6062
CoxTime	0.7571	**0.8312**

5.4 Integrated Brier Score Results

Fig. 5 Integrated Brier Score results - Benchmarks

	METABRIC	SUPPORT	GBSG	FLCHAIN	NWTCO
Reg CoxPH	**0.1361**	0.1954	0.1798	**0.1022**	0.1186
RSF	0.1513	0.1816	0.1785	0.1032	0.113
GBSM	0.1528	0.181	0.1788	0.1029	**0.1101**
DeepSurv	0.1419	**0.1806**	0.1783	0.1035	0.1143
DeepHit	0.3381	0.2856	0.3711	0.1791	0.5128
PC-Hazard	0.4477	0.3248	0.3752	0.7933	0.8398
CoxTime	0.1479	0.1917	**0.1768**	0.1223	0.1216

Fig. 6 Integrated Brier Score results - new datasets

	prostateSurvival	hepatoCellular
Reg CoxPH	**0.0586**	0.195
RSF	0.061	0.1605
GBSM	0.0615	0.1574
DeepSurv	0.0602	0.1535
DeepHit	0.077	**0.1497**
PC-Hazard	0.5331	0.5067
CoxTime	0.0587	0.1597

6 Limitations

Before I discussed my conclusions, I wanted to present the limitations I had in my experiments and in the software I used.

6.1 Limitation regarding the experiments

- I restricted my testing to specific hyperparameters, investigating different hyperparameter configurations might yield potentially improved results.

- One of the hyperparameters I examined was the network architecture. I made an effort to explore the best architectural suggestions from each article but I believe that an experienced DL researcher specializing in survival analysis could potentially suggest better architectures.
- there are additional measurements that can be computed, such as time-dependent AUC.
- I did not explore all the available DL models designed for survival analysis.

6.2 Limitation regarding the software (based on my experience only)

- Training times often exceeded expectations. The largest dataset I used comprised 11,054 samples, which in big data, is relatively small. Training DeepHit on this dataset took nearly **two days**, whereas RegCoxPH required just **6 minutes**. Perhaps with better hardware, this gap will decrease.
- It is not straight forward to compare models from different packages. I wanted to use the same measurement methods for every one of the models, and it took adjustments.

7 Conclusions

In the C-Index measurement, we observed that the ML models outperformed two out of the five benchmarks. Among these, the two best models were DeepSurv (METABRIC, 0.671) and CoxTime (GBSG, 0.6629). Across all DL models, DeepSurv closely approached the performance of the best ML model in other three benchmarks. This is reasonable since DeepSurv upholds the PH assumption, which is the basis assumption of this measurement. When considering the new datasets, two DL models demonstrated improved results: DeepHit (prostateSurvival, 0.7836) and CoxTime (hepatoCellular, 0.8125). The largest gap between any top two models measure over hepatoCellular (0.0125) for CoxTime and GBSM but from a practical perspective, this difference can be considered negligible.

In the C^{td} measurement, DL models demonstrated slightly better results, achieving the highest C^{td} scores in three out of the five tested benchmarks and the two new datasets. The best models were DeepSurv (METABRIC, 0.6717; prostateSurvival, 0.766), DeepHit (GBSG, 0.6669; NWTCO, 0.7135), CoxTime (hepatoCellular, 0.8312), RegCoxPH (FLCHAIN, 0.7968), and RSF (SUPPORT, 0.6428). The results changed compared to the C-Index measurement, especially for DeepHit, which didn't uphold the PH assumption. This exemplified the importance of using suitable measurements for the methods, otherwise, the comparison is not valid.

The IBS is the most important measure among the three, as it focuses on calibration. After reviewing the results, I saw that DL models only dominated in two benchmarks and one new dataset, which is not the same as in the other measurements. It is interesting to note that the model supposed to be the simplest, the Regularized CoxPH model, managed to outperform two benchmarks and one new dataset (METABRIC, 0.1361; FLCHAIN, 0.1022; prostateSurvival, 0.0586). This is important information considering that the training time of the Regularized CoxPH model was significantly shorter than that of the other models.

These findings are a good indication that DL models still haven't dominated the field of survival analysis. However, they also show that DL models are not far behind, as some of them achieved the best scores in the measurements. In my opinion, additional research and development are still needed in order to better utilize deep learning for survival analysis as the main approach in general.

References

[1] He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition. arXiv (2015). https://doi.org/10.48550/ARXIV.1512.03385. https://arxiv.org/abs/1512.03385

[2] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention Is All You Need. arXiv (2017). https://doi.org/10.48550/ARXIV.1706.03762. https://arxiv.org/abs/1706.03762

[3] Oord, A.v.d., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., Kavukcuoglu, K.: WaveNet: A Generative Model for Raw Audio. arXiv (2016). https://doi.org/10.48550/ARXIV.1609.03499. https://arxiv.org/abs/1609.03499

[4] Chen, T., Guestrin, C. ACM (2016). https://doi.org/10.1145/2939672.2939785. https://doi.org/10.1145%2F2939672.2939785

[5] Borisov, V., Leemann, T., Seßler, K., Haug, J., Pawelczyk, M., Kasneci, G.: Deep Neural Networks and Tabular Data: A Survey. arXiv (2021). https://doi.org/10.48550/ARXIV.2110.01889. https://arxiv.org/abs/2110.01889

[6] Zou, H.: The adaptive lasso and its oracle properties. Taylor & Francis (2006)

[7] Ishwaran, H., Kogalur, U.B., Blackstone, E.H., Lauer, M.S.: Random survival forests. Institute of Mathematical Statistics (2008). https://doi.org/10.1214/08-aoas169. https://doi.org/10.1214%2F08-aoas169

[8] Hothorn, T., Bühlmann, P., Dudoit, S., Molinaro, A., Van Der Laan, M.J.: Survival ensembles (2005). https://doi.org/10.1093/biostatistics/kxj011. https://doi.org/10.1093/biostatistics/kxj011

[9] Katzman, J.L., Shaham, U., Cloninger, A., Bates, J., Jiang, T., Kluger, Y.: DeepSurv: personalized treatment recommender system using a Cox proportional hazards deep neural network. Springer (2018). https://doi.org/10.1186/s12874-018-0482-1. https://doi.org/10.1186%2Fs12874-018-0482-1

[10] Lee, C., Zame, W., Yoon, J., van der Schaar, M.: DeepHit: A Deep Learning Approach to Survival Analysis With Competing Risks (2018). https://doi.org/10.1609/aaai.v32i1.11842. https://ojs.aaai.org/index.php/AAAI/article/view/11842

[11] Li, L., Yan, J., Xu, J., Liu, C.-Q., Zhen, Z.-J., Chen, H.-W., Ji, Y., Wu, Z.-P., Hu, J.-Y., Zheng, L., Lau, W.Y.: CXCL17 Expression Predicts Poor Prognosis and Correlates with Adverse Immune Infiltration in Hepatocellular Carcinoma. Public Library of Science (2014). https://doi.org/10.1371/journal.pone.0110064. https://doi.org/10.1371/journal.pone.0110064

[12] Kvamme, H., Ørnulf Borgan: Continuous and Discrete-Time Survival Prediction with Neural Networks (2019)

[13] Kvamme, H.: Pycox: Time-to-event prediction with PyTorch. https://github.com/havakv/pycox

[14] Lu-Yao, G.L., Albertsen, P.C., Moore, D.F., Shih, W., Lin, Y., DiPaola, R.S., Barry, M.J., Zietman, A., O'Leary, M., Walker-Corkery, E., Yao, S.-L.: Outcomes of Localized Prostate Cancer Following Conservative Management (2009). https://doi.org/10.1001/jama.2009.1348. https://doi.org/10.1001/jama.2009.1348

[15] Uno, H., Cai, T., Pencina, M.J., D'Agostino, R.B., Wei, L.J.: On the C-statistics for evaluating overall adequacy of risk prediction procedures with censored survival data (2011). https://doi.org/10.1002/sim.4154. https://onlinelibrary.wiley.com/doi/abs/10.1002/sim.4154

[16] Antolini, L., Boracchi, P., Biganzoli, E.: A time-dependent discrimination index for survival data (2005). https://doi.org/10.1002/sim.2427. https://onlinelibrary.wiley.com/doi/abs/10.1002/sim.2427

[17] Graf, E., Schmoor, C., Sauerbrei, W., Schumacher, M.: Assessment and comparison of prognostic classification schemes for survival data (1999). https://doi.org/10.1002/(SICI)1097-0258(19990915/30)18:17/18⟨2529::AID-SIM274⟩3.0.CO;2-5. https://onlinelibrary.wiley.com/doi/abs/10.1002/%28SICI%291097-0258%2819990915/30%2918%3A17/18%3C2529%3A%3AAID-SIM274%3E3.0.CO%3B2-5

[18] Davidson-Pilon, C.: lifelines: survival analysis in Python. The Open Journal (2019). https://doi.org/10.21105/joss.01317. https://doi.org/10.21105/joss.01317

[19] Pölsterl, S.: scikit-survival: A Library for Time-to-Event Analysis Built on Top of scikit-learn (2020). http://jmlr.org/papers/v21/20-729.html

[20] Nagpal, C., Potosnak, W., Dubrawski, A.: auton-survival: an Open-Source Package for Regression, Counterfactual Estimation, Evaluation and Phenotyping with Censored Time-to-Event Data (2022)