SurvivalPFN: Amortized Bayesian Survival Prediction via In-Context Learning

Shi-ang Qi[1] **Vahid Balazadeh**[1,2] **Michael Cooper**[1,2]
Russell Greiner[3,4] **Rahul G. Krishnan**[1,2]
[1] Vector Institute [2] University of Toronto [3] University of Alberta
[4] Alberta Machine Intelligence Institute

Abstract

Survival analysis provides a powerful statistical framework for modeling time-to-event outcomes in the presence of censoring. However, selecting an appropriate estimator from the many specialized survival approaches often requires substantial methodological and domain expertise. We introduce SurvivalPFN, a prior-data fitted network that amortizes Bayesian inference for censored observations through in-context learning. SurvivalPFN is pretrained on a diverse family of synthetic, identifiable, and right-censored data-generating processes, enabling it to amortize survival analysis in a single forward pass during inference. As a result, the model adapts to the effective complexity of each dataset without task-specific training or hyperparameter tuning, avoids restrictive parametric assumptions, and produces calibrated survival distributions. In a large-scale benchmark spanning 71 datasets, 21 methods, and 5 evaluation metrics, SurvivalPFN achieves strong predictive performance and often improves upon established survival models. These results suggest that SurvivalPFN offers a principled and practical foundation model for survival analysis, with potential applications in high-impact domains such as healthcare, finance, and engineering (`https://github.com/rgklab/SurvivalPFN`).

1 Introduction

Survival analysis models the distribution of time to an event of interest, with applications spanning medicine [57, 93, 80, 10, 11, 21], e-commerce [60, 75, 17], engineering [73, 7, 52], and finance [61, 24, 26]. Such models are learned and evaluated on data that often exhibits *right-censoring*: for some instances, the event is not observed during the follow-up period, so we only know that the event time exceeds the censoring time.

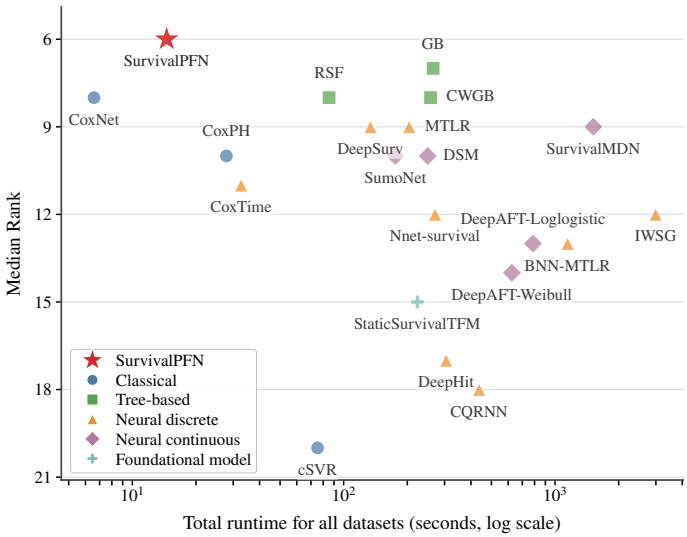


Figure 1: **Computational efficiency vs. performance across 71 datasets and 5 metrics.** SurvivalPFN achieves the best median rank while matching classical models in speed.

Various survival analysis methods have been proposed to handle right-censored data, but each imposes different inductive biases. Classical models such as Cox proportional hazards (CoxPH) [12]

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

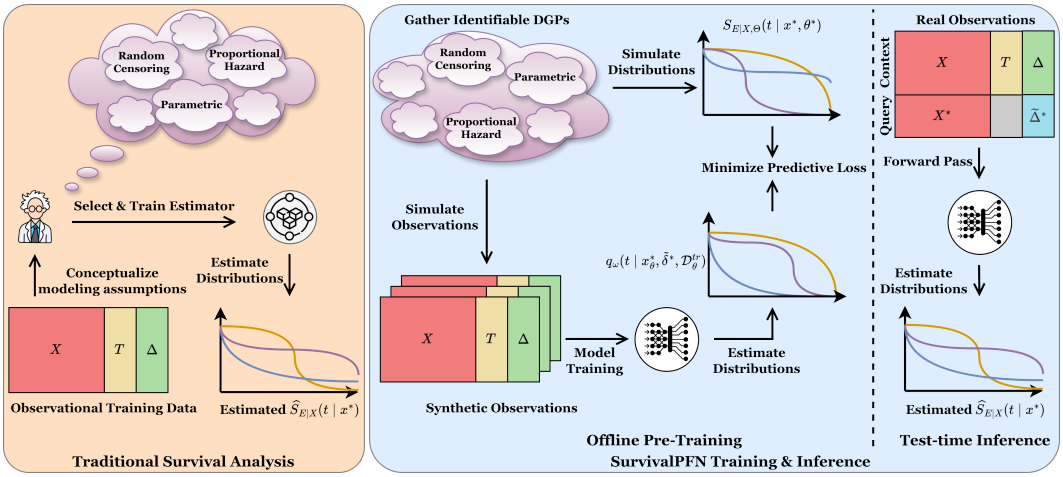


Figure 2: **Traditional survival analysis vs. SurvivalPFN.** *(Left):* Traditional survival analysis requires an analyst to select and fit a suitable estimator for the observational data. *(Right):* SurvivalPFN pre-trains on diverse synthetic, identifiable DGPs. At inference, an observed dataset is provided as context, and the survival distributions for query instances are obtained with a single forward pass.

often rely on constant hazard ratios and linear covariate effects. Ensemble methods and deep survival models improve flexibility, but typically require careful tuning and often retain structural assumptions through parametric forms [86, 72], proportional hazards [46], fixed time/quantile discretizations [104, 55, 74], or mixture-based continuous-time distributions [69, 33]. Consequently, practitioners must navigate a large set of estimators with distinct assumptions and limitations; model selection, training, and validation require substantial domain and methodological expertise.

This work aims to design a survival estimator that *(i) avoids rigid simplifying assumptions*; *(ii) adapts to the effective complexity of the observed data*; and *(iii) enables efficient inference without extensive training or hyperparameter tuning*.

To do so, we build on prior-data fitted networks (PFNs) [67]: transformer-based models [101] that learn in-context approximations of posterior predictive distributions using synthetic tasks. Rather than fitting a new survival model for each dataset, SurvivalPFN shifts computation to an offline prior-data pretraining stage. At inference time, an observed right-censored dataset is provided as context, and a single forward pass returns posterior survival distributions for new individuals. This approach provides a practical route to Bayesian survival prediction that avoids dataset-specific optimization and extensive hyperparameter tuning.

We present *SurvivalPFN*, a transformer model for survival prediction via in-context learning. Our framework uses a general-purpose prior under conditional independent censoring to generate millions of simulated data generating processes (DGPs). By training on these diverse DGPs, SurvivalPFN learns to infer conditional survival distributions directly from observed right-censoring data, yielding an easy-to-use and efficient estimator with strong empirical performance; see Figure 1. Figure 2 contrasts the SurvivalPFN workflow with traditional survival modeling. Our key contributions:

1. We introduce a framework for amortized Bayesian survival prediction via large-scale pretraining. SurvivalPFN uses a single forward pass to adapt to data complexity without task-specific training or hyperparameter tuning, while avoiding restrictive parametric assumptions.
2. We design a diverse synthetic prior over identifiable right-censored mechanisms. We characterize the conditions under which the induced posterior predictive survival distributiopn is asymptotically consistent, providing guidance for our prior construction.
3. We conduct a large-scale benchmark comparing 21 models across 71 datasets and 5 evaluation metrics, making this, to our knowledge, one of the largest survival model benchmarking studies to date. SurvivalPFN achieves the best median rank.
4. We release the code for training SurvivalPFN, together with a `scikit-learn`-style API.

2 Background

Survival Analysis and Prediction. Let $X \in \mathbb{R}^d$ denote a covariate vector, and $E, C \in \mathbb{R}_+$ represent the event and censoring times. We assume a true joint distribution P over the tuple (X, E, C, T, Δ), where $T = \min(E, C)$ and $\Delta = \mathbb{1}[E \leq C]$. Specifically, we observe N draws $\mathcal{D} = \{(x_i, t_i, \delta_i)\}_{i=1}^N$ from the distribution on observed variables $P_{\text{obs}}(X, T, \Delta)$; E and C are latent variables; only T and Δ are observed. Survival predictors aim to learn the conditional densities or survival functions:

$$f_{E|X}(t \mid x) \;=\; \Pr(E = t \mid X = x), \quad \text{or (equivalently)} \quad S_{E|X}(t \mid x) \;=\; \Pr(E > t \mid X = x).$$

Identifiable Survival Analysis. We say that the conditional survival function $S_{E|X}$ is (nonparametrically) *identified* if any two candidate data generating processes that induce the same observed law over (X, T, Δ) must also induce the same event-time survival function [99, 100]:

$$P^{(1)}_{\text{obs}}(X, T, \Delta) \;=\; P^{(2)}_{\text{obs}}(X, T, \Delta) \qquad \Longrightarrow \qquad S^{(1)}_{E|X}(t \mid x) \;=\; S^{(2)}_{E|X}(t \mid x),$$

for almost every x and all t in the identifiable support.

A sufficient condition for nonparametric identification is given by the following standard assumptions.

Assumption 2.1 (Conditional independent censoring)**.** $E \perp C \mid X$.

Assumption 2.2 (Positivity)**.** *For a time region $\mathcal{T}$ of interest,* $\Pr(C \geq t \mid X = x) > 0$*,* $\forall t \in \mathcal{T}$.

When $E \not\perp C \mid X$, the event distribution is generally not nonparametrically identifiable from (X, T, Δ) alone; identification then requires additional assumptions, *e.g.*, a specified copula family for the dependence between E and C [25, 106]. Appendix B includes more theory on identifiability.

Bayesian Survival Prediction. Consider a family of identifiable survival data-generating processes indexed by $\theta \in \Theta$, with prior $\pi(\cdot)$ over Θ. Each θ induces conditional densities and survival functions for both event and censoring times, $f_{E|X,\Theta}(e \mid x, \theta)$ and $S_{E|X,\Theta}(e \mid x, \theta)$. In Bayesian survival modeling, we place a prior density $f_\Theta(\theta)$ and infer the posterior density via Bayes' rule,

$$f_{\Theta|\mathscr{D}}(\theta \mid \mathcal{D}) \;\propto\; f_{\mathscr{D}|\Theta}(\mathcal{D} \mid \theta) f_\Theta(\theta). \tag{2.1}$$

Under conditional independent censoring, the likelihood can be decomposed into:

$$f_{\mathscr{D}|\Theta}(\mathcal{D} \mid \theta) = \prod_{i=1}^N \left[f_{E|X,\Theta}(t_i \mid x_i, \theta)\, S_{C|X,\Theta}(t_i \mid x_i, \theta)\right]^{\delta_i} \left[f_{C|X,\Theta}(t_i \mid x_i, \theta)\, S_{E|X,\Theta}(t_i \mid x_i, \theta)\right]^{1-\delta_i}$$

where, for $A \in \{E, C\}$, $S_{A|X,\Theta}(t_i \mid x_i, \theta) = \int_{t_i}^\infty f_{A|X,\Theta}(\tau \mid x_i, \theta)\, d\tau$. Given a new covariate vector x^*, the Bayesian posterior predictive distribution (PPD) of the event time is

$$f_{E|X,\mathscr{D}}(t \mid x^*, \mathcal{D}) \;=\; \int_\Theta f_{E|X,\Theta}(t \mid x^*, \vartheta) f_{\Theta|\mathscr{D}}(\vartheta \mid \mathcal{D})\, d\vartheta. \tag{2.2}$$

Analogously, the posterior predictive survival distribution (PPSD) is

$$S_{E|X,\mathscr{D}}(t \mid x^*, \mathcal{D}) \;=\; \int_\Theta S_{E|X,\Theta}(t \mid x^*, \vartheta) f_{\Theta|\mathscr{D}}(\vartheta \mid \mathcal{D})\, d\vartheta. \tag{2.3}$$

This framework is attractive because it integrates over plausible survival mechanisms rather than committing to a single fitted model. However, it is difficult to use directly with flexible survival models: evaluating the likelihood can require numerical integration to obtain $S_{E|X,\Theta}$; the normalizing constant in Equation 2.1 and the posterior predictive integrals in Equations 2.2 and 2.3 are generally intractable; and approximate inference methods such as Markov chain Monte Carlo (MCMC) [71, 1, 103] or variational inference (VI) [44, 102, 36, 81] must be rerun for each new dataset. We therefore seek an amortized procedure that preserves the posterior-predictive interpretation of Bayesian survival prediction while avoiding dataset-specific posterior computation.

Prior-Data Fitted Networks and Amortized Bayesian Inference. Prior-data fitted networks (PFNs) amortize Bayesian posterior prediction by training transformers on synthetic tasks sampled from a prior-data generating process [67, 68]. Each task consists of a context set and query inputs, and the PFN is trained to predict query targets according to the posterior predictive distribution induced by the prior. After pretraining, posterior inference is no longer explicit: the transformer's in-context computation maps a new dataset and query points directly to predictive distributions in a single forward pass, replacing dataset-specific MCMC or VI. This connects PFNs to meta-learning [18], but replaces task-specific adaptation with in-context inference. PFNs have achieved strong transfer in tabular prediction [37, 38, 85], causal effect estimation [3, 89], and time-series prediction [39, 4], motivating our use of this paradigm for amortized Bayesian survival prediction.

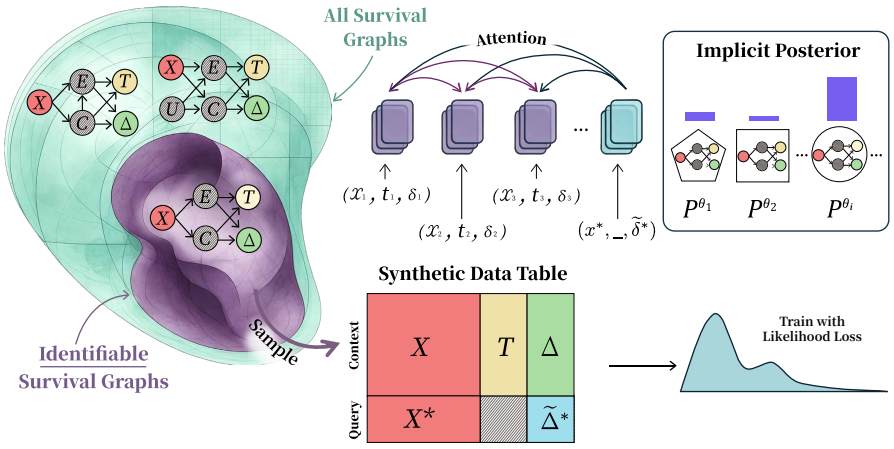


Figure 3: **Training SurvivalPFN.** At each iteration, we sample an identifiable survival DGP and use it to generate context tokens (X, T, Δ) together with query covariates X^*. Query tokens are formed by pairing X^* with query indicators $\widetilde{\Delta}^*$, and SurvivalPFN predicts the requested event- or censoring-time distribution. The model is trained by minimizing the likelihood loss.

3 Method

3.1 SurvivalPFN: Amortized Posterior Predictive Inference

Overview. SurvivalPFN learns an in-context approximation to Bayesian posterior predictive inference for right-censored survival data. Instead of specifying a tractable likelihood and performing posterior inference separately for each dataset, we specify a prior through a simulator over identifiable right-censored DGPs. A draw $\theta \sim \pi(\cdot)$ determines a joint law P^θ over (X, E, C, T, Δ). For each synthetic task, the simulator produces an observed right-censored context dataset $\mathcal{D}_\theta^{tr} = \{(x_i, t_i, \delta_i)_\theta\}_{i=1}^N$, together with held-out query covariates x_θ^* and their latent event and censoring times (e_θ^*, c_θ^*). The latent times are used only during prior-data training; at inference time, SurvivalPFN receives the same information available in ordinary survival prediction: an observed dataset $\mathcal{D}$ and query covariates x^*.

Architecture. Let q_ω denote a transformer with parameters ω. Given a context dataset and a query covariate, SurvivalPFN outputs a predictive distribution over time. We additionally provide a binary *query indicator* $\widetilde{\delta}^*$, which is a control input specifying which PPD the model should return:

$$q_\omega\left(t \,\middle|\, x_\theta^*, \widetilde{\delta}^*, \mathcal{D}_\theta^{tr}\right) \approx \begin{cases} f_{E|X,\mathcal{D}}(t \mid x_\theta^*, \mathcal{D}_\theta^{tr}), & \widetilde{\delta}^* = 1, \\ f_{C|X,\mathcal{D}}(t \mid x_\theta^*, \mathcal{D}_\theta^{tr}), & \widetilde{\delta}^* = 0. \end{cases} \tag{3.1}$$

Thus, $\widetilde{\delta}^* = 1$ asks the model for the event-time PPD, whose tail probability gives the PPSD, while $\widetilde{\delta}^* = 0$ asks for the posterior predictive censoring distribution (PPCD). This indicator is not an observed event label for the query point; rather, it specifies the prediction target.

SurvivalPFN parameterizes q_ω using the PFN-style transformer architecture of TabDPT [62] and CausalPFN [3]; see Appendix D.2 for details. As shown in Figure 3, each context row $(x_i, t_i, \delta_i)_\theta \in \mathcal{D}_\theta^{tr}$ is embedded as a context token, while each query token is formed from $(x_\theta^*, \widetilde{\delta}^*)$. Although event-time prediction is our objective, previous work has shown that explicitly modeling the censoring process can improve survival prediction by leveraging information shared between event and censoring mechanisms [2]. Motivated by this, we consider three query-indicator schedules during training:

- *Event-only:* always sets $\widetilde{\delta}^* = 1$ and trains the model directly for PPSD prediction;
- *Both:* duplicates each query with $\widetilde{\delta}^* \in \{0, 1\}$ and trains both event- and censoring-time prediction;
- *Random:* samples the query indicator according to the empirical censoring pattern in the context.

The transformer uses an asymmetric attention mask: context tokens attend to one another, while query tokens attend only to the context tokens and not to other queries, as shown by the two-way

and one-way arrows in Figure 3. Together with the absence of positional encodings, this makes predictions invariant to the ordering of the context dataset and conditionally independent across query points given the context.

The model represents each PPD as a discretized histogram over $L = 1024$ time bins. For each query token, the transformer output is projected to logits over bins, followed by a softmax:

$$q_\omega\Big(\cdot \,\Big|\, x^*_\theta, \widetilde{\delta}^*, \mathcal{D}^{tr}_\theta\Big) \;=\; \Big[q_{\omega,\ell}(x^*_\theta, \widetilde{\delta}^*, \mathcal{D}^{tr}_\theta)\Big]_{\ell=1}^{L}.$$

Before discretization, we apply a monotone transformation of time, such as `lognormal2normal` or `time2quantile`, to respect the nonnegative support of survival times and allocate resolution more evenly across the context time range; see Appendix D.3. The predicted PPSD is obtained by summing the event-time tail probability mass:

$$\widehat{S}_\omega(\tau_k \mid x^*, \mathcal{D}) \;=\; \sum_{\ell=k+1}^{L} q_{\omega,\ell}(x^*, \widetilde{\delta}^* = 1, \mathcal{D}). \tag{3.2}$$

Training. Training follows the PFN principle [67, 37]. At each gradient update, we sample $\theta \sim \pi(\cdot)$, generate a context dataset and query points from the corresponding DGP, and use the simulator-provided latent times as supervision. Given the query indicator, the supervised target is

$$r^*_\theta(\widetilde{\delta}^*) \;=\; \widetilde{\delta}^* e^*_\theta + \big(1 - \widetilde{\delta}^*\big) c^*_\theta.$$

Let $g_{\mathcal{D}^{tr}_\theta}$ be the context-fitted monotone time transformation, and let $\kappa_{\mathcal{D}^{tr}_\theta}(r) \in \{1, \dots, L\}$ denote the transformed-time bin containing $g_{\mathcal{D}^{tr}_\theta}(r)$. We train SurvivalPFN with the discrete negative log-likelihood,

$$\mathcal{L}_{\mathrm{NLL}}(\omega) \;=\; \mathbb{E}_{\theta\sim\pi(\cdot)}\, \mathbb{E}_{\mathcal{D}^{tr}_\theta,\, x^*_\theta,\, e^*_\theta,\, c^*_\theta,\, \widetilde{\delta}^*} \left[-\log q_{\omega, \kappa_{\mathcal{D}^{tr}_\theta}\left(r^*_\theta(\widetilde{\delta}^*)\right)} \Big(x^*_\theta, \widetilde{\delta}^*, \mathcal{D}^{tr}_\theta\Big) \right]. \tag{3.3}$$

This objective is a tractable prior-data likelihood for the requested latent query time. At the population optimum, the Bayes-optimal predictor is the conditional distribution of the latent target bin given the observed context and query. Thus, when the prior is restricted to identifiable right-censored DGPs, minimizing Equation 3.3 trains SurvivalPFN to approximate the Bayesian PPD induced by $\pi(\cdot)$. We also consider a smoothed cross-entropy variant, which replaces the one-hot target with a narrow Gaussian-smoothed histogram over nearby time bins; see Appendix D.4.

Inference. At inference time, SurvivalPFN is applied directly to a real right-censored dataset $\mathcal{D}$ and query covariates x^*. A single forward pass with $\widetilde{\delta}^* = 1$ returns the event-time predictive distribution, from which we compute $\widehat{S}_\omega(t \mid x^*, \mathcal{D})$ via Equation 3.2. No dataset-specific gradient updates, posterior sampling, variational optimization, or hyperparameter tuning are required. In this way, pretraining shifts the computational burden from test-time Bayesian inference to an offline prior-fitting stage, yielding a reusable amortized Bayesian survival predictor.

3.2 Consistency of the Bayesian Posterior Predictive Target

We next characterize the consistency of the Bayesian posterior predictive target that SurvivalPFN is trained to approximate. Under the regularity and identifiability conditions stated in Appendix C, this target converges to the true conditional survival function on the identifiable time region, for prior-almost every data-generating process.

Proposition 3.1 (Informal consistency)**.** *Let θ^* denote the parameter of the true survival data-generating process. Assume the prior over survival DGPs is supported only on identifiable right-censored mechanisms. Then, for every time t in the support and for almost every query covariate x^*, the Bayesian PPSD converges almost surely to the true conditional survival function:*

$$S_{E|X,\mathscr{D}}(t \mid x^*, \mathcal{D}) \;=\; \int_\Theta S_{E|X,\Theta}(t \mid x^*, \theta) f_{\Theta|\mathscr{D}}(\theta \mid \mathcal{D})\, d\theta \quad \xrightarrow[N\to\infty]{\text{a.s.}} \quad S_{E|X,\Theta}(t \mid x^*, \theta^*).$$

Proof sketch. Group survival DGPs into observational equivalence classes:

$$\theta_1 \;\sim\; \theta_2 \quad\Longleftrightarrow\quad P^{\theta_1}_{\mathrm{obs}}(X, T, \Delta) \;=\; P^{\theta_2}_{\mathrm{obs}}(X, T, \Delta).$$

The observed data can distinguish different equivalence classes, but cannot distinguish DGPs within the same class. By applying Bayesian consistency to this quotient space, the posterior concentrates on the true observational equivalence class $[\theta^*]$ as $N \to \infty$. Therefore, the Bayesian PPSD converges to the posterior average of $S_{E|X,\Theta}(t \mid x^*, \theta)$ over DGPs that are observationally equivalent to θ^*, that is, over all θ satisfying $[\theta] = [\theta^*]$. If the prior is survival-identifiable, then every DGP in this equivalence class induces the same conditional event-time survival function. Hence this posterior average is equal to the true survival function $S_{E|X,\Theta}(t \mid x^*, \theta^*)$, which gives the desired consistency. □

Moreover, if SurvivalPFN exactly amortizes this posterior predictive distribution, then its predicted survival inherits the same consistency guarantee:

$$\widehat{S}_\omega(t \mid x^*, \mathcal{D}) \xrightarrow[N\to\infty]{\text{a.s.}} S_{E|X,\Theta}(t \mid x^*, \theta^*).$$

3.3 Prior over Identifiable Survival DGPs

Optimizing our loss function does not require an explicit parameterization of the distribution P^θ. Instead, it requires a prior $\pi(\cdot)$ that can generate observational datasets, consisting of tuples (x_i, t_i, δ_i), along with query points x^*_θ with their event/censoring times. Still, not every choice of prior has the desired properties. Proposition 3.1 highlights two key design principles for the prior: *(i)* it should rule out non-identifiable right-censored mechanisms; and *(ii)* subject to this identifiability constraint, the prior should be broad enough to cover a diverse family of plausible survival mechanisms, increasing the chance that the true DGP θ^* lies within, or close to, the prior support.

Our synthetic prior follows a hierarchical generative process. For each synthetic task, we first sample task-level parameters $\theta = (\theta_X, \theta_E, \theta_C)$, specifying the covariate distribution, event-time mechanism, censoring-time mechanism, and censoring regime, respectively. We use randomly initialized multilayer perceptrons (MLPs), following Hollmann et al. [37], to construct flexible nonlinear generative mechanisms. The covariates are generated by an unconditional MLP, whereas event and censoring times are generated by separate mechanisms conditional on the covariates. Specifically, for subject i, we generate the covariates and latent event and censoring times from:

$$x_i \;=\; g_X(u_{X,i}; \theta_X), \quad e_i \;=\; g_E(x_i, u_{E,i}; \theta_E), \quad c_i \;=\; g_C(x_i, u_{C,i}; \theta_C),$$

where $u_{X,i}$, $u_{E,i}$, and $u_{C,i}$ denote independent exogenous random variables. We can then construct observed time and event indicator following the causal observation rule. Conditional on the sampled DGP parameters, the event and censoring mechanisms have independent stochastic components and share only the observed covariates X.

To cover even more diverse survival regimes (see data diversity in Figure 4 right), the prior mixes four families of synthetic survival generators. The *naive prior* treats generated table outputs as raw event and censoring times. The *survival-distribution prior* samples smooth random monotone maps, such as Bernstein maps, and pushes uniform noise through them to obtain flexible distributions for event and censoring. The *mixture prior* samples event and censoring times from mixtures of Weibull or log-normal components with covariate-dependent mixture parameters. Finally, the *kitchen-sink prior* is a meta-prior that samples one of the above generators for each task, thereby increasing prior diversity.

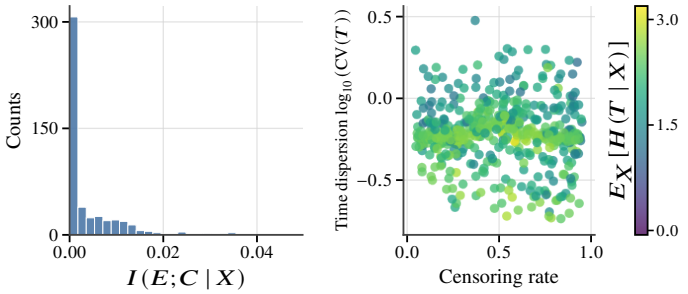


Figure 4: **Summary over 500 generated datasets.** *(Left):* Histogram of conditional mutual information. *(Right):* Diversity coverage over censoring rate and observed-time dispersion, colored by conditional observed-time entropy.

For each synthetic dataset, we also sample one of four censoring mechanisms: *uniform censoring*, where C is sampled from a uniform time range; *random censoring*, where C is generated from an

independent tabular mechanism; *administrative censoring*, where subjects have different entry times but share a common study end date; and *conditional independent censoring*, where both E and C depend on X but are generated from independent conditional blocks. The observed survival data are then formed using the standard survival law. Appendix D.1 provides full prior-generation details.

4 Experiments and Results

We conduct extensive experiments to investigate the following research questions (RQs):

RQ1. How does SurvivalPFN compare with survival baselines in predictive performance?

RQ2. How efficient is SurvivalPFN compared with other survival estimators?

RQ3. How sensitive is SurvivalPFN to the proportion of training/context samples?

RQ4. How does SurvivalPFN compare with general tabular foundation models (TFMs)?

RQ5. Does SurvivalPFN use context samples to inform its survival predictions??

RQ6. How do priors, query schedules, transformations, and losses affect performance?

We evaluate SurvivalPFN on a large-scale benchmark covering diverse real-world regimes. The benchmark contains 91 datasets (see Figure 5 for a preview): 20 are used only for SurvivalPFN checkpoint selection, and the remaining 71 are held out for final evaluation. Datasets are drawn from `SurvSet` [16] and additional textbook, software-package, and recent-publication sources. Table 4 and Appendix F provide full dataset descriptions and summaries.

Models. We evaluate 21 survival models from five families (Table 2); details are in Appendices E.1-E.3.

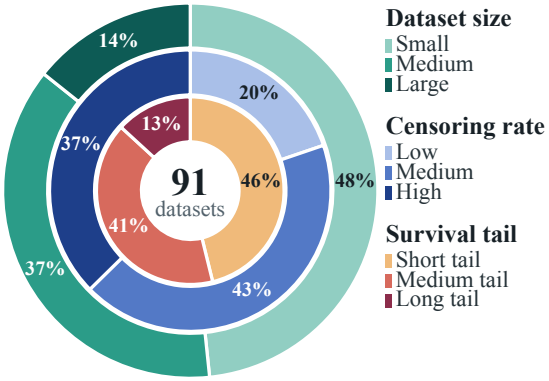


Figure 5: Dataset size (cutoffs at 500 and 5000), censoring rate and tail rate (cutoffs at 33% and 67%).

- *Tabular foundation models:* SurvivalPFN and StaticSurvivalTFM [47].
- *Classical survival models:* CoxPH [12], CoxNet [95], cSVR [79].
- *Tree-based models:* GB [87], CWGB [41], and RSF [42].
- *Neural discrete-time models:* DeepHit [55], DeepSurv [46], MTLR [104, 19], Nnet-survival [5, 23], CoxTime [54], IWSG [32], CQRNN [74], and BNN-MTLR [82].
- *Neural continuous-time models:* DSM [69], SuMoNet [88], SurvivalMDN [33], DeepAFT-Weibull [72], and DeepAFT-Loglogistic [72].

Metrics. We evaluate models using five metrics (Appendix E.4): IPCW-adjusted integrated Brier score (IBS; probabilistic accuracy) [27], concordance index (CI; discrimination) [35], D-calibration (distributional calibration) [30], median-time mean absolute error (MAE; time prediction) [81], and log-rank reliability (agreement with observed time-to-event outcomes).

Experimental Protocol. For each dataset, we conduct 10 repeated experiments using independent 70%/30% train/test splits. We report the mean $\pm$ standard deviation across the 10 repetitions. For aggregate comparison, models are ranked separately within each dataset and metric, with rank 1 assigned to the best-performing method. Appendices E.5-E.6 provide further details.

RQ1: Predictive Performance. Figure 6 summarizes model ranks across all benchmark datasets, with better-performing methods appearing farther to the right. SurvivalPFN achieves the strongest overall rank among the compared methods, indicating robust performance across the full benchmark suite. Metric-wise, SurvivalPFN is among the leading methods for IBS, MAE, and Log-Rank, showing strong probabilistic survival prediction, accurate time estimation, and strong agreement with observed time-to-event outcomes. It also remains competitive on CI and D-calibration, although several other baselines (*e.g.*, RSF, CWGB, DeepSurv) rank higher in these metric-specific rankings.

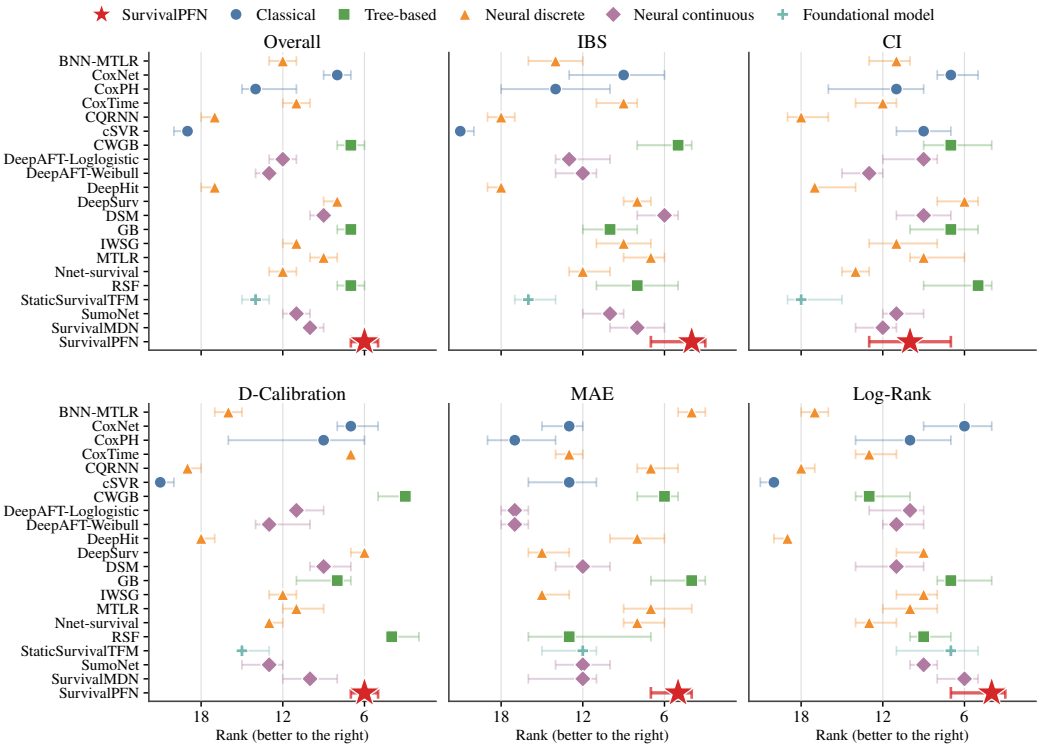


Figure 6: **Model ranks across 71 benchmark datasets.** Points/stars denote median ranks across datasets, with horizontal bars showing 95% bootstrap confidence intervals for the median rank.

Additional stratified results (based on sample size and censoring rate) are provided in Appendix G.1. There, SurvivalPFN shows its largest advantage on small datasets (Figure 12), while its relative performance decreases as dataset size increases (Figures 13 and 14). This behavior is consistent with the *motivation of amortized Bayesian survival prediction*: when each downstream dataset contains limited observations, SurvivalPFN can leverage the inductive structure learned during prior-data pretraining rather than fitting a flexible survival model from scratch. In contrast, across low-, medium-, and high-censoring regimes (Figures 15-17), SurvivalPFN remains consistently strong.

RQ2: Computational Efficiency. Figure 1 compares the computational efficiency of SurvivalPFN with all baselines. We report the total training-plus-inference time across datasets, excluding hyperparameter-tuning time for neural-network-based methods. SurvivalPFN is highly efficient: it is only modestly slower than CoxNet, the fastest baseline, while achieving the best overall ranking across the five evaluation metrics. By contrast, tree-based and neural-network-based methods require substantially greater computation.

RQ3: Sensitivity to Training-Set Size. Figure 7 shows how model performance changes as the fraction of training data varies. SurvivalPFN shows stable predictive performance across split ratios, maintaining consistently low IBS and high CI even when only a small fraction of the data is used for training. In contrast, several baselines are more sensitive to limited training data: CoxNet, GB, and CWGB perform poorly at the smallest ratio and improve markedly as more data become available. These trends suggest that SurvivalPFN is comparatively robust in low-data regimes. Appendix G.3 and Figure 19 contain results for more datasets.

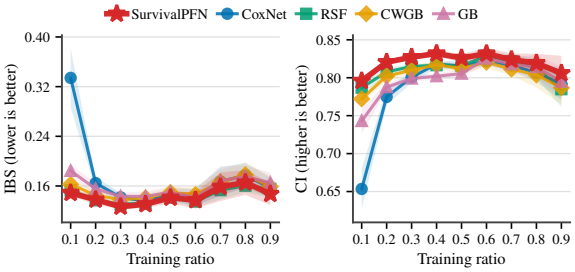


Figure 7: **Performance on the PBC dataset for SurvivalPFN and top-performing models.** Shaded regions denote standard errors over 10 repeated runs.

RQ4: Comparison with General TFMs. Figure 8 compares SurvivalPFN with general-purpose tabular foundational regressors by training them only on uncensored instances. SurvivalPFN achieves the best overall rank and is consistently the top-ranked method across all metrics. This suggests that directly adapting generic TFMs to survival outcomes is insufficient: explicitly pretraining on identifiable right-censored survival tasks yields substantially more reliable survival prediction.

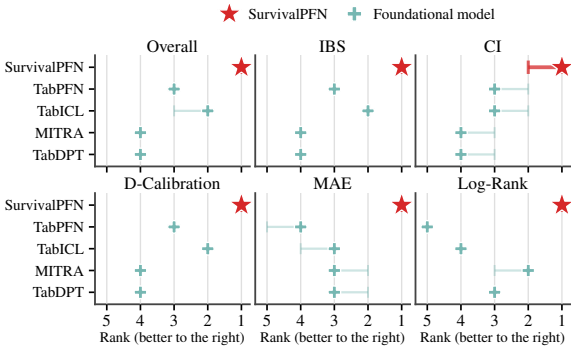


Figure 8: **Comparison of SurvivalPFN with selected general TFMs across 61 benchmark datasets, excepting the 10 high-dimensiontal datasets.** Plotting conventions follow Figure 6.

RQ5: Evidence of using context. To assess whether SurvivalPFN uses the association between context covariates and survival outcomes, we conducted an outcome-permutation diagnostic on 71 datasets. Specifically, we kept the context covariates, context size, and query samples fixed, and jointly permuted the observed survival times and event indicators across context rows. This preserves the context's outcome distribution while disrupting its association with the covariates. Figure 9 compares standard and permuted performance, with each point representing a dataset's mean over the ten repetitions. Permutation generally increases IBS and decreases CI, placing points above the equality line in the IBS panel and below it in the CI panel. Across datasets, mean CI decreases from 67.15% to 51.00% – approaching chance-level discrimination. For data-level performance, please refer Figure 21.

RQ6: Ablation Studies. Due to space constraints, Appendix G.6 reports how prior design, query schedules, monotone transformations, and objective functions affect SurvivalPFN performance.

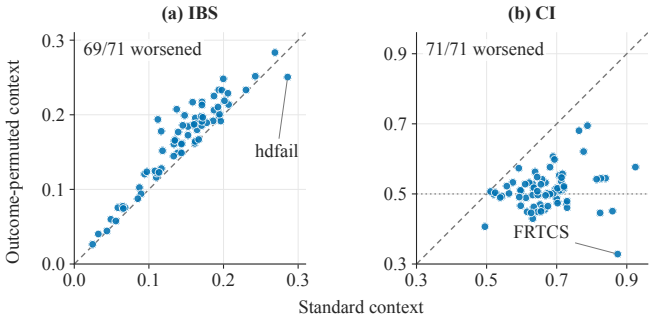


Figure 9: **Prediction performance under context outcome permutation.** Standard and outcome-permuted contexts are compared across 71 datasets for IBS and CI. Each point represents a dataset's mean over ten paired repetitions. Dashed diagonals indicate equal performance; the dotted horizontal line marks CI 0.5.

5 Related Work

Classical and Deep Survival Analysis. A broad range of estimators has been developed for right-censored data. CoxPH [12] relies on proportional hazards and linear covariate effects, while fully parametric models such as exponential [63] and Weibull [76] models impose explicit distributional forms. Modern machine-learning methods improve flexibility, but introduce other assumptions. Neural Cox-based models, such as DeepSurv [46] and CoxTime [54], relax linear covariate effects but retain Cox-style hazard modeling. Discrete-time models, including MTLR [104, 19] and DeepHit [55, 56], depend on a chosen time grid and may become overparameterized with many bins. Continuous-time neural models are more flexible, but still impose structure through parametric mixtures [69], latent-variable models [86, 65], monotonic density estimators [88, 9], or neural ODE hazards [29, 98].

Bayesian Survival Analysis. Bayesian survival models quantify uncertainty by placing priors over model parameters and integrating over posterior uncertainty. Recent neural variants include BNN-ISD, which uses Bayesian neural networks to obtain credible intervals and supports feature selection [82]; Bayesian LSTM-SURV, which combines the survival likelihood with Bayesian mixed-effects updating under a Weibull parametric form [22]; and NeuralSurv, which uses variational inference for Bayesian deep survival prediction [66]. While these methods demonstrate the value of Bayesian uncertainty quantification, they remain tied to method-specific assumptions and per-dataset inference or updating.

Concurrent Work on TFMs for Survival Analysis. Two concurrent works also explore TFMs for survival analysis.

Kim et al. [47] convert survival prediction to a sequence of binary classification tasks over discretized time points, enabling off-the-shelf TFMs without survival-specific pretraining – it is not a foundation model designed specifically for survival prediction. This simple reduction avoids new pretraining, but substantially expands each instance into many time-indexed examples; as a result, context length scales with the number of bins. This has two practical consequences. First, it substantially increases computation. Second, once the expanded context exceeds the backbok TFMs context limit, the method must subsample the context and performance can suffer when the TFM sees only a compressed view of the risk set. We include Kim et al. [47]'s static formulation as STATICSURVIVALTFM

Seletkov et al. [92] instead pretrain a survival-specific in-context model on synthetic data from parametric extended-hazard mechanisms. This is closer to SurvivalPFN, but its prior is limited to parametric hazard families and random censoring ($E \perp C$), potentially limiting broader use. In contrast, SurvivalPFN uses a broader family of identifiable right-censored DGPs, supports covariate-dependent censoring under conditional independence, avoids explicit parametric structure, and provides a posterior-predictive consistency argument. SIC is not directly compared because public weights/code are not yet available.

6 Conclusions, Limitations, and Future Work

We introduced SurvivalPFN, a prior-data fitted network for amortized Bayesian survival prediction from right-censored data. By pretraining on diverse, identifiable survival data-generating processes, SurvivalPFN produces posterior predictive survival distributions in a single forward pass, without dataset-specific training or hyperparameter tuning. Across 71 real-world datasets, SurvivalPFN achieves strong overall performance while remaining computationally efficient, suggesting that PFN-style in-context learning is a promising foundation for flexible survival prediction. Because survival models already inform high-stakes decisions – from clinical risk scores [58] to resource-allocation policies [48] – we believe that advances in accuracy, efficiency, and uncertainty quantification for survival models can generate substantial human benefit.

Several limitations remain. First, SurvivalPFN relies on conditional independent censoring to preserve identifiability; under dependent censoring, the event-time distribution is not nonparametrically identifiable from observed data alone, and our method is not valid. Extending SurvivalPFN to identifiable dependent-censoring priors, such as copula methods [25, 106], is an important direction. Additionally, SurvivalPFN inherits the size-scalability trade-off of PFN-style models, with reduced relative performance on larger tables [38, 3]; improving long-context inference is left for future work.

Acknowledgment

This research received support from Vector Institute, the Natural Science and Engineering Research Council of Canada (NSERC) and the Alberta Machine Intelligence Institute (Amii). The authors extend their gratitude to the anonymous reviewers for their insightful feedback and valuable suggestions.

References

[1] Christophe Andrieu, Nando De Freitas, Arnaud Doucet, and Michael I Jordan. An introduction to mcmc for machine learning. *Machine learning*, 50(1):5–43, 2003.

[2] Shi ang Qi, Yakun Yu, and Russell Greiner. Explicitly modeling censoring produces superior survival predictors. In *Forty-third International Conference on Machine Learning*, 2026. URL `https://openreview.net/forum?id=0xJt4PqPJv`.

[3] Vahid Balazadeh, Hamidreza Kamkari, Valentin Thomas, Benson Li, Junwei Ma, Jesse C Cresswell, and Rahul G Krishnan. Causalpfn: Amortized causal effect estimation via in-context learning. *arXiv preprint arXiv:2506.07918*, 2025.

[4] David Berghaus, Patrick Seifner, Kostadin Cvejoski, César Ojeda, and Ramsés J Sánchez. In-context learning of temporal point processes with foundation inference models. *arXiv preprint arXiv:2509.24762*, 2025.

[5] Elia Biganzoli, Patrizia Boracchi, Luigi Mariani, and Ettore Marubini. Feed forward neural networks for the analysis of censored survival data: a partial logistic regression approach. *Statistics in medicine*, 17(10):1169–1186, 1998.

[6] Norman E Breslow. Analysis of survival data under the proportional hazards model. *International Statistical Review/Revue Internationale de Statistique*, pages 45–57, 1975.

[7] DC Broadway, M Iester, M Schulzer, and GR Douglas. Survival analysis for success of molteno tube implants. *British Journal of Ophthalmology*, 85(6):689–695, 2001.

[8] Davide Chicco and Giuseppe Jurman. Machine learning can predict survival of patients with heart failure from serum creatinine and ejection fraction alone. *BMC medical informatics and decision making*, 20(1):16, 2020.

[9] Pawel Chilinski and Ricardo Silva. Neural likelihoods via cumulative distribution functions. In *Conference on Uncertainty in Artificial Intelligence*, pages 420–429. PMLR, 2020.

[10] Michael Cooper, Zongliang Ji, and Rahul G Krishnan. Machine learning in computational histopathology: Challenges and opportunities. *Genes, Chromosomes and Cancer*, 62(9): 540–556, 2023.

[11] Michael J Cooper, Xiang Gao, Xun Zhao, Dariia Khoroshchuk, Yingke Wang, Amirhossein Azhie, Maryam Naghibzadeh, Sandra Holdsworth, Jed Adam Gross, Michael Brudno, et al. Dynameld: a dynamic model of end-stage liver disease for equitable prioritization. *medRxiv*, pages 2024–11, 2024.

[12] David R Cox. Regression models and life-tables. *Journal of the Royal Statistical Society: Series B (Methodological)*, 34(2):187–202, 1972.

[13] Christina Curtis, Sohrab P Shah, Suet-Feung Chin, Gulisa Turashvili, Oscar M Rueda, Mark J Dunning, Doug Speed, Andy G Lynch, Shamith Samarajiwa, Yinyin Yuan, et al. The genomic and transcriptomic architecture of 2,000 breast tumours reveals novel subgroups. *Nature*, 486 (7403):346–352, 2012.

[14] Aaron Defazio, Xingyu Yang, Harsh Mehta, Konstantin Mishchenko, Ahmed Khaled, and Ashok Cutkosky. The road less scheduled. *Advances in Neural Information Processing Systems*, 37:9974–10007, 2024.

[15] Joseph Leo Doob. Application of the theory of martingales. *Le Calcul des Probabilites et ses Applications*, pages 23–27, 1949. URL `https://cir.nii.ac.jp/crid/1573387449499005824`.

[16] Erik Drysdale. Survset: An open-source time-to-event dataset repository. *arXiv preprint arXiv:2203.03094*, 2022.

[17] Juan Pablo Equihua, Henrik Nordmark, Maged Ali, and Berthold Lausen. Modelling customer churn for the retail industry in a deep learning based sequential framework. *arXiv preprint arXiv:2304.00575*, 2023.

[18] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

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

[20] Stephane Fotso et al. PySurvival: Open source package for survival analysis modeling, 2019–. URL `https://www.pysurvival.io/`.

[21] Xiang Gao, Michael Cooper, Maryam Naghibzadeh, Amirhossein Azhie, Mamatha Bhat, and Rahul G Krishnan. Predicting long-term allograft survival in liver transplant recipients. *arXiv preprint arXiv:2408.05437*, 2024.

[22] Yuanyuan Gao, Shuo Li, Di Wang, Jianming Mao, and Linhan Ouyang. A neural network-based survival analysis model considering censored data for failure prediction. *IEEE Transactions on Automation Science and Engineering*, 22:24585–24598, 2025.

[23] Michael F Gensheimer and Balasubramanian Narasimhan. A scalable discrete-time survival model for neural networks. *PeerJ*, 7:e6257, 2019.

[24] Adrian Gepp and Kuldeep Kumar. The role of survival analysis in financial distress prediction. *International research journal of finance and economics*, 16(16):13–34, 2008.

[25] Ali Hossein Foomani Gharari, Michael Cooper, Russell Greiner, and Rahul G Krishnan. Copula-based deep survival models for dependent censoring. In *Uncertainty in Artificial Intelligence*, pages 669–680. PMLR, 2023.

[26] Pierre Giot and Armin Schwienbacher. Ipos, trade sales and liquidations: Modelling venture capital exits using survival analysis. *Journal of Banking & Finance*, 31(3):679–702, 2007.

[27] Erika Graf, Claudia Schmoor, Willi Sauerbrei, and Martin Schumacher. Assessment and comparison of prognostic classification schemes for survival data. *Statistics in medicine*, 18 (17-18):2529–2545, 1999.

[28] Léo Grinsztajn, Klemens Flöge, Oscar Key, Felix Birkel, Philipp Jund, Brendan Roof, Benjamin Jäger, Dominik Safaric, Simone Alessi, Adrian Hayler, et al. Tabpfn-2.5: Advancing the state of the art in tabular foundation models. *arXiv preprint arXiv:2511.08667*, 2025.

[29] Stefan Groha, Sebastian M Schmon, and Alexander Gusev. A general framework for survival analysis and multi-state modelling. *arXiv preprint arXiv:2006.04893*, 2020.

[30] Humza Haider, Bret Hoehn, Sarah Davis, and Russell Greiner. Effective ways to build and evaluate individual survival distributions. *Journal of Machine Learning Research*, 21(85): 1–63, 2020.

[31] Scott M Hammer, David A Katzenstein, Michael D Hughes, Holly Gundacker, Robert T Schooley, Richard H Haubrich, W Keith Henry, Michael M Lederman, John P Phair, Manette Niu, et al. A trial comparing nucleoside monotherapy with combination therapy in hiv-infected adults with cd4 cell counts from 200 to 500 per cubic millimeter. *New England Journal of Medicine*, 335(15):1081–1090, 1996.

[32] Xintian Han, Mark Goldstein, Aahlad Puli, Thomas Wies, Adler Perotte, and Rajesh Ranganath. Inverse-weighted survival games. *Advances in neural information processing systems*, 34: 2160–2172, 2021.

[33] Xintian Han, Mark Goldstein, and Rajesh Ranganath. Survival mixture density networks. In *Machine Learning for Healthcare Conference*, pages 224–248. PMLR, 2022.

[34] Frank E Harrell, Robert M Califf, David B Pryor, Kerry L Lee, and Robert A Rosati. Evaluating the yield of medical tests. *Jama*, 247(18):2543–2546, 1982.

[35] Frank E Harrell Jr, Kerry L Lee, and Daniel B Mark. Multivariable prognostic models: issues in developing models, evaluating assumptions and adequacy, and measuring and reducing errors. *Statistics in medicine*, 15(4):361–387, 1996.

[36] Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *the Journal of machine Learning research*, 14(1):1303–1347, 2013.

[37] Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. Tabpfn: A transformer that solves small tabular classification problems in a second. *arXiv preprint arXiv:2207.01848*, 2022.

[38] Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637(8045):319–326, 2025.

[39] Shi Bin Hoo, Samuel Müller, David Salinas, and Frank Hutter. From tables to time: Extending tabpfn-v2 to time series forecasting. *arXiv preprint arXiv:2501.02945*, 2025.

[40] David W Hosmer Jr, Stanley Lemeshow, and Susanne May. *Applied survival analysis: regression modeling of time-to-event data*. John Wiley & Sons, 2008.

[41] Torsten Hothorn, Peter Bühlmann, Sandrine Dudoit, Annette Molinaro, and Mark J Van Der Laan. Survival ensembles. *Biostatistics*, 7(3):355–373, 2006.

[42] Hemant Ishwaran, Udaya B Kogalur, Eugene H Blackstone, Michael S Lauer, et al. Random survival forests. *Annals of Applied Statistics*, 2(3):841–860, 2008.

[43] Alistair EW Johnson, Lucas Bulgarelli, Lu Shen, Alvin Gayles, Ayad Shammout, Steven Horng, Tom J Pollard, Sicheng Hao, Benjamin Moody, Brian Gow, et al. Mimic-iv, a freely accessible electronic health record dataset. *Scientific data*, 10(1):1, 2023.

[44] Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

[45] OJWF Kardaun. Statistical survival analysis of male larynx-cancer patients-a case study. *Statistica neerlandica*, 37(3):103–125, 1983.

[46] Jared L Katzman, Uri Shaham, Alexander Cloninger, Jonathan Bates, Tingting Jiang, and Yuval Kluger. Deepsurv: personalized treatment recommender system using a cox proportional hazards deep neural network. *BMC medical research methodology*, 18:1–12, 2018.

[47] Da In Kim, Wei Siang Lai, and Kelly W Zhang. Tabular foundation models can do survival analysis. *arXiv preprint arXiv:2601.22259*, 2026.

[48] W Ray Kim, Ajitha Mannalithara, Julie K Heimbach, Patrick S Kamath, Sumeet K Asrani, Scott W Biggins, Nicholas L Wood, Sommer E Gentry, and Allison J Kwong. Meld 3.0: the model for end-stage liver disease updated for the modern era. *Gastroenterology*, 161(6): 1887–1895, 2021.

[49] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[50] John P Klein and Melvin L Moeschberger. *Survival analysis: techniques for censored and truncated data*, volume 1230. Springer, 2003.

[51] David G Kleinbaum and Mitchel Klein. *Survival analysis a self-learning text*. Springer, 1996.

[52] David Kuhajda. Using survival analysis to evaluate medical equipment battery life. *Biomedical instrumentation & technology*, 50(3):184–189, 2016.

[53] Neeraj Kumar, Shi-ang Qi, Li-Hao Kuan, Weijie Sun, Jianfei Zhang, and Russell Greiner. Learning accurate personalized survival models for predicting hospital discharge and mortality of covid-19 patients. *Scientific reports*, 12(1):4472, 2022.

[54] Håvard Kvamme, Ørnulf Borgan, and Ida Scheel. Time-to-event prediction with neural networks and cox regression. *Journal of machine learning research*, 20(129):1–30, 2019.

[55] Changhee Lee, William Zame, Jinsung Yoon, and Mihaela Van Der Schaar. Deephit: A deep learning approach to survival analysis with competing risks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[56] Changhee Lee, Jinsung Yoon, and Mihaela Van Der Schaar. Dynamic-deephit: A deep learning approach for dynamic survival analysis with competing risks based on longitudinal data. *IEEE Transactions on Biomedical Engineering*, 67(1):122–133, 2019.

[57] Chunyang Li, Vikas Patil, Kelli M Rasmussen, Christina Yong, Hsu-Chih Chien, Debbie Morreall, Jeffrey Humpherys, Brian C Sauer, Zachary Burningham, and Ahmad S Halwani. Predicting survival in veterans with follicular lymphoma using structured electronic health record information and machine learning. *International Journal of Environmental Research and Public Health*, 18(5):2679, 2021.

[58] Gregory YH Lip, Robby Nieuwlaat, Ron Pisters, Deirdre A Lane, and Harry JGM Crijns. Refining clinical risk stratification for predicting stroke and thromboembolism in atrial fibrillation using a novel risk factor-based approach: the euro heart survey on atrial fibrillation. *Chest*, 137(2):263–272, 2010.

[59] Charles Lawrence Loprinzi, John A Laurie, H Sam Wieand, James E Krook, Paul J Novotny, John W Kugler, Joan Bartel, Marlys Law, Marilyn Bateman, and Nancy E Klatt. Prospective evaluation of prognostic variables from patient-completed questionnaires. north central cancer treatment group. *Journal of Clinical Oncology*, 12(3):601–607, 1994.

[60] Junxiang Lu. Predicting customer churn in the telecommunications industry—-an application of survival analysis modeling using sas. *SAS User Group International (SUGI27) Online Proceedings*, 114:27, 2002.

[61] Martti Luoma and Erkki K Laitinen. Survival analysis as a tool for company failure prediction. *Omega*, 19(6):673–678, 1991.

[62] Junwei Ma, Valentin Thomas, Rasa Hosseinzadeh, Alex Labach, Hamidreza Kamkari, Jesse C Cresswell, Keyvan Golestan, Guangwei Yu, Anthony L Caterini, and Maksims Volkovs. Tabdpt: Scaling tabular foundation models on real data. *arXiv preprint arXiv:2410.18164*, 2024.

[63] William Mendenhall and RJ Hader. Estimation of parameters of mixed exponentially distributed failure time distributions from censored life test data. *Biometrika*, 45(3-4):504–520, 1958.

[64] Margaret Merrell and Lawrence E Shulman. Determination of prognosis in chronic disease, illustrated by systemic lupus erythematosus. *Journal of chronic diseases*, 1(1):12–32, 1955.

[65] Xenia Miscouridou, Adler Perotte, Noémie Elhadad, and Rajesh Ranganath. Deep survival analysis: Nonparametrics and missingness. In *Machine Learning for Healthcare Conference*, pages 244–256. PMLR, 2018.

[66] Mélodie Monod, Alessandro Micheli, and Samir Bhatt. Neuralsurv: Deep survival analysis with bayesian uncertainty quantification. *arXiv preprint arXiv:2505.11054*, 2025.

[67] Samuel Müller, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka, and Frank Hutter. Transformers can do bayesian inference. *arXiv preprint arXiv:2112.10510*, 2021.

[68] Samuel Müller, Arik Reuter, Noah Hollmann, David Rügamer, and Frank Hutter. Position: The future of bayesian prediction is prior-fitted. *arXiv preprint arXiv:2505.23947*, 2025.

[69] Chirag Nagpal, Xinyu Li, and Artur Dubrawski. Deep survival machines: Fully parametric survival regression and representation learning for censored data with competing risks. *IEEE Journal of Biomedical and Health Informatics*, 25(8):3163–3175, 2021.

[70] Chirag Nagpal, Willa Potosnak, and Artur Dubrawski. Auton-survival: an open-source package for regression, counterfactual estimation, evaluation and phenotyping with censored time-to-event data. In *Machine Learning for Healthcare Conference*, pages 585–608. PMLR, 2022.

[71] Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

[72] Patrick A Norman, Wanlu Li, Wenyu Jiang, and Bingshu E Chen. deepaft: A nonlinear accelerated failure time model with artificial neural network. *Statistics in Medicine*, 43(19): 3689–3701, 2024.

[73] Dimitris Papathanasiou, Konstantinos Demertzis, and Nikos Tziritas. Machine failure prediction using survival analysis. *Future Internet*, 15(5):153, 2023.

[74] Tim Pearce, Jong-Hyeon Jeong, Jun Zhu, et al. Censored quantile regression neural networks for distribution-free survival analysis. *Advances in neural information processing systems*, 35: 7450–7461, 2022.

[75] África Periáñez, Alain Saas, Anna Guitart, and Colin Magne. Churn prediction in mobile social games: Towards a complete assessment using survival ensembles. In *2016 IEEE international conference on data science and advanced analytics (DSAA)*, pages 564–573. IEEE, 2016.

[76] Richard Peto and Peter Lee. Weibull distributions for continuous-carcinogenesis experiments. *Biometrics*, pages 457–470, 1973.

[77] Beatriz Pineiro-Lamas, Ana Lopez-Cheda, Ricardo Cao, Laura Ramos-Alonso, Gabriel Gonzalez-Barbeito, Cayetana Barbeito-Caamano, and Alberto Bouzas-Mosquera. A cardiotoxicity dataset for breast cancer patients. *Scientific Data*, 10(1):527, 2023.

[78] Sebastian Pölsterl. scikit-survival: A library for time-to-event analysis built on top of scikit-learn. *Journal of Machine Learning Research*, 21(212):1–6, 2020.

[79] Sebastian Pölsterl, Nassir Navab, and Amin Katouzian. Fast training of support vector machines for survival analysis. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 243–259. Springer, 2015.

[80] Shi-ang Qi, Neeraj Kumar, Jian-Yi Xu, Jaykumar Patel, Sambasivarao Damaraju, Grace Shen-Tu, and Russell Greiner. Personalized breast cancer onset prediction from lifestyle and health history information. *Plos one*, 17(12):e0279174, 2022.

[81] Shi-ang Qi, Neeraj Kumar, Mahtab Farrokh, Weijie Sun, Li-Hao Kuan, Rajesh Ranganath, Ricardo Henao, and Russell Greiner. An effective meaningful way to evaluate survival models. *Proceedings of machine learning research*, 202:28244, 2023.

[82] Shi-ang Qi, Neeraj Kumar, Ruchika Verma, Jian-Yi Xu, Grace Shen-Tu, and Russell Greiner. Using bayesian neural networks to select features and compute credible intervals for personalized survival prediction. *IEEE Transactions on Biomedical Engineering*, 70(12):3389–3400, 2023.

[83] Shi-ang Qi, Weijie Sun, and Russell Greiner. Survivaleval: A comprehensive open-source python package for evaluating individual survival distributions. In *Proceedings of the AAAI Symposium Series*, volume 2, pages 453–457, 2023.

[84] Shi-Ang Qi, Yakun Yu, and Russell Greiner. Conformalized survival distributions: A generic post-process to increase calibration. In *International Conference on Machine Learning*, pages 41303–41339. PMLR, 2024.

[85] Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. Tabiclv2: A better, faster, scalable, and open tabular foundation model. *arXiv preprint arXiv:2602.11139*, 2026.

[86] Rajesh Ranganath, Adler Perotte, Noémie Elhadad, and David Blei. Deep survival analysis. In *Machine Learning for Healthcare Conference*, pages 101–114. PMLR, 2016.

[87] Greg Ridgeway. The state of boosting. *Computing science and statistics*, pages 172–181, 1999.

[88] David Rindt, Robert Hu, David Steinsaltz, and Dino Sejdinovic. Survival regression with proper scoring rules and monotonic neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 1190–1205. PMLR, 2022.

[89] Jake Robertson, Arik Reuter, Siyuan Guo, Noah Hollmann, Frank Hutter, and Bernhard Schölkopf. Do-pfn: In-context learning for causal effect estimation. *arXiv preprint arXiv:2506.06039*, 2025.

[90] James M Robins and Dianne M Finkelstein. Correcting for noncompliance and dependent censoring in an aids clinical trial with inverse probability of censoring weighted (ipcw) log-rank tests. *Biometrics*, 56(3):779–788, 2000.

[91] Peter H Rossi, Richard A Berk, and Kenneth J Lenihan. Money, work and crime: some experimental results, 1980.

[92] Dmitrii Seletkov, Paul Hager, Rickmer Braren, Daniel Rueckert, and Raphael Rehms. Survival in-context: Prior-fitted in-context learning tabular foundation model for survival analysis. *arXiv preprint arXiv:2603.29475*, 2026.

[93] Yunlang She, Zhuochen Jin, Junqi Wu, Jiajun Deng, Lei Zhang, Hang Su, Gening Jiang, Haipeng Liu, Dong Xie, Nan Cao, et al. Development and validation of a deep learning model for non–small cell lung cancer survival. *JAMA network open*, 3(6):e205842–e205842, 2020.

[94] Marek Sikora, Łukasz Wróbel, and Adam Gudyś. Guider: A guided separate-and-conquer rule learning in classification, regression, and survival settings. *Knowledge-Based Systems*, 173:1–14, 2019.

[95] Noah Simon, Jerome H Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for cox's proportional hazards model via coordinate descent. *Journal of statistical software*, 39: 1–13, 2011.

[96] Cancer Genome Atlas Research Network Tissue source sites: Duke University Medical School McLendon Roger 1 Friedman Allan 2 Bigner Darrell 1, Emory University Van Meir Erwin G. 3 4 5 Brat Daniel J. 5 6 M. Mastrogianakis Gena 3 Olson Jeffrey J. 3 4 5, Henry Ford Hospital Mikkelsen Tom 7 Lehman Norman 8, MD Anderson Cancer Center Aldape Ken 9 Alfred Yung WK 10 Bogler Oliver 11, University of California San Francisco VandenBerg Scott 12 Berger Mitchel 13 Prados Michael 13, et al. Comprehensive genomic characterization defines human glioblastoma genes and core pathways. *Nature*, 455(7216):1061–1068, 2008.

[97] Lin-Quan Tang, Chao-Feng Li, Jing Li, Wen-Hui Chen, Qiu-Yan Chen, Lian-Xiong Yuan, Xiao-Ping Lai, Yun He, Yun-Xiu-Xiu Xu, Dong-Peng Hu, et al. Establishment and validation of prognostic nomograms for endemic nasopharyngeal carcinoma. *Journal of the National Cancer Institute*, 108(1):djv291, 2016.

[98] Weijing Tang, Jiaqi Ma, Qiaozhu Mei, and Ji Zhu. Soden: A scalable continuous-time survival model through ordinary differential equation networks. *Journal of Machine Learning Research*, 23(34):1–29, 2022.

[99] Anastasios Tsiatis. A nonidentifiability aspect of the problem of competing risks. *Proceedings of the National Academy of Sciences*, 72(1):20–22, 1975.

[100] Anastasios A Tsiatis. *Semiparametric theory and missing data*, volume 4. Springer, 2006.

[101] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[102] Martin J Wainwright, Michael I Jordan, et al. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1–2):1–305, 2008.

[103] Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688, 2011.

[104] Chun-Nam Yu, Russell Greiner, Hsiu-Chin Lin, and Vickie Baracos. Learning patient-specific cancer survival distributions as a sequence of dependent regressors. *Advances in neural information processing systems*, 24, 2011.

[105] Ahmet Zehir, Ryma Benayed, Ronak H Shah, Aijazuddin Syed, Sumit Middha, Hyunjae R Kim, Preethi Srinivasan, Jianjiong Gao, Debyani Chakravarty, Sean M Devlin, et al. Mutational landscape of metastatic cancer revealed from prospective clinical sequencing of 10,000 patients. *Nature medicine*, 23(6):703–713, 2017.

[106] Weijia Zhang, Chun Kai Ling, and Xuanhui Zhang. Deep copula-based survival analysis for dependent censoring with identifiability guarantees. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 20613–20621, 2024.

[107] Xiyuan Zhang, Danielle C Maddix, Junming Yin, Nick Erickson, Abdul Fatir Ansari, Boran Han, Shuai Zhang, Leman Akoglu, Christos Faloutsos, Michael W Mahoney, et al. Mitra: Mixed synthetic priors for enhancing tabular foundation models. *arXiv preprint arXiv:2510.21204*, 2025.

Appendix Contents

A Notation **19**

B Identifiability and Non-identifiability under Right Censoring **19**

C Posterior-Predictive Consistency **21**

C.1 Notation and regularity assumptions 22

C.2 Observed-law equivalence and survival identifiability 22

C.3 Consistency of the Bayesian PPSD 23

C.4 Implication for SurvivalPFN 24

D Additional SurvivalPFN Details **24**

D.1 Synthetic Prior and DGP Simulation 24

D.2 Architecture Details 28

D.3 Monotone Time Transformations 29

D.4 Training Objective 31

D.5 Inference Procedure 32

E Benchmarking Details **32**

E.1 Baseline Model Details 32

E.2 Unified Time Grid for Consistent Evaluation 35

E.3 Hyperparameter Tuning 36

E.4 Evaluation Metrics 37

E.5 Compute Environment and Runtime Protocol 38

E.6 Performance Reporting Protocol 39

F Benchmark Dataset Details **40**

G Additional Experimental Results **43**

G.1 RQ1: Predictive Performance 43

G.2 RQ2: Computational Efficiency 45

G.3 RQ3: Sensitivity to Training-Set Size 45

G.4 RQ4: Compare with General Tabular Foundational Models 45

G.5 RQ5: Evidence of Using Context 47

G.6 RQ6: Ablation Studies 47

H Concurrent Works on Tabular Foundation Models for Survival Analysis **49**

A Notation

Table 1 summarizes all the notation used throughout the paper. Following the convention, we use uppercase letters for random variables and lowercase letters for realizations.

Table 1: Summary of notation in the paper.

Notation	Definition
$\mathcal{X}, \mathbb{R}_+$	Covariate space and nonnegative time domain.
$i \in \{1, \dots, N\}$	Index for an individual observation.
d	Number of covariates/features.
$X \in \mathcal{X}$	Covariates.
$E \in \mathbb{R}_+$	Latent event time of interest.
$C \in \mathbb{R}_+$	Latent censoring time.
$T = \min(E, C)$	Observed follow-up time.
$\Delta = \mathbb{1}\{E \le C\}$	Event indicator: $\Delta = 1$ for observed event and $\Delta = 0$ for right-censored.
(x_i, t_i, δ_i)	Observed right-censored tuple.
(e_i, c_i)	Latent event and censoring times.
$\mathscr{D}$	Random observed dataset.
$\mathcal{D} = \{(x_i, t_i, \delta_i)\}_{i=1}^N$	Realization of $\mathscr{D}$.
$\mathcal{D}_\theta^{tr}, \mathcal{D}_\theta^{te}$	Synthetic context/training set and query/test set sampled from DGP parameter θ during prior-data pretraining.
ω	Trainable parameters of SurvivalPFN.
$\theta \in \Theta$	Latent parameter indexing a survival data-generating process.
θ^*	True DGP parameter for a downstream dataset.
$\pi(\cdot)$	Prior distribution over survival DGPs used to generate synthetic training tasks.
P^θ	Joint law of (X, E, C, T, Δ) under DGP parameter θ.
P^θ_{obs}	Observational law of (X, T, Δ) induced by P^θ.
$f^\theta_{E\|X}(t \mid x)$	Conditional density function for event time, under θ.
$F^\theta_{E\|X}(t \mid x)$	Conditional CDF for event time, $P^\theta(E \le t \mid X = x)$.
$S^\theta_{E\|X}(t \mid x)$	Conditional survival function for event time, $P^\theta(E > t \mid X = x) = 1 - F^\theta_{E\|X}(t \mid x)$.
$\lambda^\theta_{E\|X}(t \mid x)$	Conditional hazard function, $\lambda^\theta_E(t \mid x) = f^\theta_{E\|X}(t \mid x)/S^\theta_{E\|X}(t \mid x)$.
$S^\theta_{C\|X}(t \mid x)$	Conditional survival function for censoring time, $P^\theta(C \ge t \mid X = x)$.
$\lambda^\theta_{C\|X}(t \mid x)$	Conditional censoring hazard.
$f_{E\|X,\mathscr{D}}(t \mid x, \mathcal{D})$	Posterior predictive event-time distribution.
$S_{E\|X,\mathscr{D}}(t \mid x, \mathcal{D})$	Posterior predictive survival distribution (PPSD).
x^*	Query covariates for a new individual.
$\widetilde{\delta}^*$	Query indicator supplied to SurvivalPFN.
$q_\omega(\cdot \mid x^*, \widetilde{\delta}^*, \mathcal{D})$	SurvivalPFN's predictive distribution over discretized time bins.
L	Number of time bins used to represent the predictive distribution.
$\{\mathcal{I}_\ell\}_{\ell=1}^L$	SurvivalPFN's transformed-time bins used to represent the discretized predictive distribution.
$\mathcal{G} = \{t_1 < \cdots < t_m\}$	Discrete time grid for benchmarking discrete-time survival models.
$\text{CV}(T)$	Coefficient of variation of observed times: $\text{CV}(T) = \text{sd}(T)/\mathbb{E}[T]$, when used in DGP diagnostics.

B Identifiability and Non-identifiability under Right Censoring

In the context of survival analysis and causal inference, identifiability is the prerequisite for learning. If a model is non-identifiable, infinite data cannot distinguish between multiple underlying "truths"

(*e.g.*, whether a drug works or if patients are simply dropping out due to side effects). Without identifiability, no consistent estimator exists, and any conclusions drawn from the data rely entirely on untestable assumptions rather than empirical evidence.

Tsiatis [99, Theorem 2] first proved that the latent joint distribution of event time E and censoring time C is not identifiable from the (infinite) observed data (T, Δ) without additional assumptions. Formally, we restate the theorem using the notation that is consistent within this paper, by considering E and C as two competing events:

Theorem B.1 (Non-identifiability; Marginal)**.** *Let $S_{E,C}(e, c) = P(\, E > e, C > c\,)$ be an arbitrary joint survival function where E and C are dependent. There exists a different joint survival function $S^*_{E,C}(e, c)$, constructed such that E and C are independent – $S^*_{E,C}(e, c) = S^*_E(e)\, S^*_C(c)$ – which generates the exact same observed data distribution $P(T, \Delta)$.*

This means, without the assumption of random censoring ($E \perp C$), the marginal survival function $S_{E|X}(t)$ cannot be uniquely determined. This theorem can be easily extend to the conditional setting:

Theorem B.2 (Non-identifiability; Conditional)**.** *Let X be a set of covariates. Let the true conditional joint survival function be $S_{E,C|X}(e, c \mid x) = P(E > e, C > c \mid X = x)$, where $E \not\perp C \mid X$. For any such dependent model, there exists a valid conditional independent model $S^*_{E,C|X}(e, c \mid x) = S^*_{E|X}(e \mid x) S^*_{C|X}(c \mid x)$ such that the observable distributions of $(T, \Delta \mid X)$ are identical.*

Proof. While the proof for this conditional non-identifiability is straightforward via following Tsiatis [99]'s step, we present the proof for completeness.

Let the observed data be characterized by the conditional sub-survival functions:

$$\begin{aligned}\Phi(t \mid x) &= P(T > t, \Delta = 1 \mid x)\\ \Psi(t \mid x) &= P(T > t, \Delta = 0 \mid x)\end{aligned}$$

These functions completely describe the likelihood of the observed data.

We construct a "proxy" independent world, denoted by a superscript ^{ind}, by defining its conditional hazards to match the observed cause-specific hazards of the original world:

$$\begin{aligned}\lambda^{\text{ind}}_{E|X}(t \mid x) &= \lim_{dt \to 0} \frac{\Pr^{\text{ind}}(t \le T < t + dt, \Delta = 1 \mid T \ge t, X = x)}{dt} = \frac{-\frac{\partial}{\partial t}\Phi(t \mid x)}{\Phi(t \mid x) + \Psi(t \mid x)}\\ \lambda^{\text{ind}}_{C|X}(t \mid x) &= \lim_{dt \to 0} \frac{\Pr^{\text{ind}}(t \le T < t + dt, \Delta = 0 \mid T \ge t, X = x)}{dt} = \frac{-\frac{\partial}{\partial t}\Psi(t \mid x)}{\Phi(t \mid x) + \Psi(t \mid x)}\end{aligned}$$

We define the marginal survival functions in the proxy world as:

$$\begin{aligned}S^{\text{ind}}_{E|X}(t \mid x) &= \exp\left(-\int_0^t \lambda^{\text{ind}}_{E|X}(u \mid x) du\right),\\ S^{\text{ind}}_{C|X}(t \mid x) &= \exp\left(-\int_0^t \lambda^{\text{ind}}_{C|X}(u \mid x) du\right).\end{aligned}$$

And the joint distribution (with conditional independence):

$$S^{\text{ind}}_{E,C|X}(e, c \mid x) = S^{\text{ind}}_{E|X}(e \mid x)\, S^{\text{ind}}_{C|X}(c \mid x)$$

To verify that this proxy world generates the same data $\Phi(t \mid x)$, we calculate the probability of observing an event in the proxy world:

$$\begin{aligned}\Phi^{\text{ind}}(t \mid x) &= \int_t^\infty f^{\text{ind}}_{E|X}(u \mid x) S^{\text{ind}}_{C|X}(u \mid x) du\\ &= \int_t^\infty \lambda^{\text{ind}}_{E|X}(u \mid x) S^{\text{ind}}_{E|X}(u \mid x) S^{\text{ind}}_{C|X}(u \mid x) du\\ &= \int_t^\infty \lambda^{\text{ind}}_{E|X}(u \mid x) S^{\text{ind}}(u, u \mid x) du\end{aligned}$$

Substituting the definitions of $\lambda^{\text{ind}}_{E|X}$ and noting that $S^{\text{ind}}(u, u \mid x) = \Phi(u \mid x) + \Psi(u \mid x)$ (the overall survival probability matches the sum of sub-survival functions):

$$\begin{aligned}\Phi^{\text{ind}}(t \mid x) &= \int_t^\infty \left(\frac{-\frac{\partial}{\partial u}\Phi(u \mid x)}{\Phi(u \mid x) + \Psi(u \mid x)} \right) (\Phi(u \mid x) + \Psi(u \mid x)) du \\ &= \int_t^\infty -\frac{\partial}{\partial u}\Phi(u \mid x) du \\ &= \Phi(t \mid x)\end{aligned}$$

Since $\Phi^{\text{ind}}(t \mid x) = \Phi(t \mid x)$ (and by symmetry $\Psi^{\text{ind}}(t \mid x) = \Psi(t \mid x)$), the independent model S^{ind} is indistinguishable from the true dependent model S. □

While Theorem B.2 states that we cannot distinguish *dependent* from *independent* censoring using data alone, the corollary below establishes that if we are willing to assume independence (either marginal or conditional), the latent event distribution becomes identifiable.

Corollary B.1 (Identifiability under Independence)**.** *Suppose we restrict our attention to the class of models that satisfy conditional independent censoring, $E \perp C \mid X$ (including $E \perp C$). Then, the marginal survival function of the event, $S_{E|X}(t \mid x)$, is uniquely identifiable from the observed data distribution. That is, if two models S and S^{ind} both satisfy conditional independence but have different event marginals ($S_{E|X} \neq S^{\text{ind}}_{E|X}$), they must generate distinct observed data distributions.*

Proof. We prove this by contradiction. Let $\lambda_{E|X}(t \mid x)$ denote the *cause-specific* hazard derived purely from the data (X, T, Δ):

$$\lambda_{E|X}(t \mid x) \;=\; \lim_{dt \to 0} \frac{P(t \leq T < t + dt, \Delta = 1 \mid T \geq t, X = x)}{dt}$$

Let $\lambda(t \mid x)$ denote the *net* hazard of the event of interest:

$$\lambda(t \mid x) \;=\; \lim_{dt \to 0} \frac{P(t \leq E < t + dt \mid E \geq t, X = x)}{dt}$$

Under the assumption of conditional independence ($E \perp C \mid X$), standard survival theory dictates that the net hazard is equal to the cause-specific hazard: $\lambda_{E|X}(t \mid x) = \lambda(t \mid x)$. Since the hazard function uniquely defines the survival function via $S_{E|X}(t \mid x) = \exp(-\int_0^t \lambda_{E|X}(u \mid x) du)$, the latent distribution $S_{E|X}$ is uniquely determined by the observed function $\lambda_{E|X}$.

Now, consider two models with different marginals, $S^{(1)}_{E|X}(t \mid x) \neq S^{(2)}_{E|X}(t \mid x)$. This inequality implies their net hazards must differ: $\lambda^{(1)}(t \mid x) \neq \lambda^{(2)}(t \mid x)$. By the equality derived above, their observed cause-specific hazards must also differ: $\lambda^{(1)}_{E|X}(t \mid x) \neq \lambda^{(2)}_{E|X}(t \mid x)$. Different hazards imply different observed data distributions. Thus, the model is identifiable. □

These properties imply that while SurvivalPFN can be robustly trained under assumptions of marginal or conditional independence, it cannot learn dependent censoring mechanisms from observed data alone.

C Posterior-Predictive Consistency

This appendix formalizes Proposition 3.1. The result concerns the Bayesian posterior predictive survival distribution (PPSD) defined in Equation 2.3. It shows that, under an identifiable survival prior, the Bayesian PPSD is asymptotically consistent for the true conditional event-time survival function. We then state the corresponding idealized implication for SurvivalPFN when the transformer exactly amortizes this Bayesian target.

C.1 Notation and regularity assumptions

Recall that a survival data-generating process (DGP) $P^\theta(X, E, C, T, \Delta)$ is indexed by $\theta \in \Theta$. We write P^θ_{obs} for the marginal distribution over the observable random variables (X, T, Δ). We still use $\pi(\cdot)$ to denote the prior over Θ induced by the synthetic prior-data generator used to pretrain SurvivalPFN.

For each θ, let

$$S_{E|X,\Theta}(t \mid x, \theta) \;:=\; \Pr(E > t \mid X = x, \Theta = \theta)$$

denote the conditional event-time survival function. For an observed dataset $\mathcal{D} = \{(X_i, T_i, \Delta_i)\}_{i=1}^N$, where $(X_i, T_i, \Delta_i) \overset{\text{i.i.d.}}{\sim} P^\theta_{\text{obs}}$, and a query covariate vector x^*, the Bayesian PPSD is (repeated from Equation 2.3)

$$S_{E|X,\mathscr{D}}(t \mid x^*, \mathcal{D}) \;=\; \int_\Theta S_{E|X,\Theta}(t \mid x^*, \vartheta) f_{\Theta|\mathscr{D}}(\vartheta \mid \mathcal{D})\, d\vartheta.$$

Assumption C.1 (Regularity)**.** *We assume the following standard regularity conditions.*

1. $(\Theta, \mathcal{B}_\Theta)$ *is a standard Borel parameter space.*
2. *The maps* $\theta \mapsto P^\theta$ *and* $\theta \mapsto P^\theta_{\text{obs}}$ *are measurable.*
3. *The image set* $\{P^\theta_{\text{obs}} : \theta \in \Theta\}$ *is a Borel subset of the space of probability measures over* (X, T, Δ).
4. *For each time* t *in the evaluation region, there exists a version of* $S_{E|X,\Theta}(t \mid x^*, \theta)$ *that is jointly measurable in* (x^*, θ).

These assumptions are technical rather than substantive. They ensure that priors, posteriors, conditional expectations, and the quotient construction below are well-defined. They are satisfied by the usual finite-dimensional parameter spaces and measurable simulators used in statistical and machine learning models.

C.2 Observed-law equivalence and survival identifiability

The full latent law $P^\theta(X, E, C, T, \Delta)$ is generally not identifiable from right-censored observations. The data can identify only the observed law P^θ_{obs} over (X, T, Δ). We therefore group DGP parameters by observational equivalence:

$$\theta_1 \;\sim\; \theta_2 \qquad \Longleftrightarrow \qquad P^{\theta_1}_{\text{obs}} \;=\; P^{\theta_2}_{\text{obs}}.$$

Let

$$\mathcal{Q} \;:=\; \Theta/\!\sim$$

denotes the set of observational quotient space (equivalence classes) induced by $\sim$, and let $[\theta] \in \mathcal{Q}$ denote the equivalence class of θ. Each class $[\theta]$ contains all latent survival DGPs that are indistinguishable.

Definition C.1 (Survival-identifiable prior)**.** *Fix a time* t *in the evaluation region. We say that the prior* π *is* survival-identifiable at time t *if there exists a measurable map*

$$F_t : \mathcal{X} \times \mathcal{Q} \;\to\; [0, 1]$$

such that, for π*-almost every* θ *and* P^π_X*-almost every* x *(except where* θ *or* x *has probability* 0*),*[1]

$$S_{E|X,\Theta}(t \mid x, \theta) \;=\; F_t(x, [\theta]).$$

Equivalently, within the support of the prior, any two DGPs that induce the same observational distribution over (X, T, Δ) *must also induce the same conditional event-time survival function at time* t.

The conditional independent censoring and positivity assumptions discussed in Section 2 provide sufficient conditions for this definition: under those assumptions, $S_{E|X}(t \mid x)$ is a functional of the observed law $P(X, T, \Delta)$ on the identifiable time region.

[1] In the following, we will just use the phrase "every θ" and "every x" for simplicity.

C.3 Consistency of the Bayesian PPSD

Proposition C.1 (Formal consistency)**.** *Fix a time t in the evaluation region. Under Assumption C.1, there exist sets $\mathcal{X}_0 \subseteq \mathcal{X}$ and $\Theta_0 \subseteq \Theta$ with*

$$P_X^{\pi}(\mathcal{X}_0) \;=\; 1, \qquad \pi(\Theta_0) \;=\; 1,$$

such that for every $x^* \in \mathcal{X}_0$ and every $\theta^* \in \Theta_0$, then

$$S_{E|X,\mathscr{D}}(t \mid x^*, \mathcal{D}) \;\xrightarrow[N\to\infty]{\text{a.s.}}\; S_{E|X,\Theta}(t \mid x^*, \theta^*) \tag{C.1}$$

if and only if the prior π is survival-identifiable at time t.

For any finite or countable evaluation grid $\mathcal{G}$, the same result holds simultaneously for all $t \in \mathcal{G}$ by intersecting the corresponding full-measure sets.

Proof. We first define the quotient-level target. For fixed t and x, we define

$$M_t(x^*, [\theta]) \;:=\; \mathbb{E}_\pi\left[S_{E|X,\Theta}(t \mid x^*, \vartheta) \mid [\vartheta] = [\theta]\right].$$

This is the prior-average survival probability among all DGPs that induce the same observed law as θ. Since $0 \le S_{E|X,\Theta}(t \mid x^*, \vartheta) \le 1$, this conditional expectation is integrable.

By construction, two different elements of $\mathcal{Q}$ correspond to two different observational distributions. Thus, the quotient parameter $[\theta]$ is identifiable from the observed distribution. Assumption C.1 ensures that the quotient model is measurable and that Doob's consistency theorem [15] applies to posterior expectations of integrable functions on this quotient space.

Therefore, for every true parameter θ^* (except where θ has probability 0), if $\mathcal{D} \sim P_{\text{obs}}^{\theta^*}$, then

$$\mathbb{E}_\pi\left[M_t(x^*, [\theta]) \mid \mathcal{D}\right] \;\xrightarrow[N\to\infty]{\text{a.s.}}\; M_t(x^*, [\theta^*]). \tag{C.2}$$

We now show that the left-hand side of Equation C.2 is the Bayesian PPSD. Since the observed data distribution depends on θ only through the equivalence class $[\theta]$, we have the conditional independence

$$\mathcal{D} \perp \theta \mid [\theta].$$

Hence,

$$\begin{aligned}
\mathbb{E}_\pi\left[M_t(x^*, [\theta]) \mid \mathcal{D}\right] \;&=\; \mathbb{E}_\pi\left[\mathbb{E}_\pi\left[S_{E|X,\Theta}(t \mid x^*, \theta) \mid [\theta]\right] \mid \mathcal{D}\right] \\
&=\; \mathbb{E}_\pi\left[\mathbb{E}_\pi\left[S_{E|X,\Theta}(t \mid x^*, \theta) \mid [\theta], \mathcal{D}\right] \mid \mathcal{D}\right] \\
&=\; \mathbb{E}_\pi\left[S_{E|X,\Theta}(t \mid x^*, \theta) \mid \mathcal{D}\right] \\
&=\; \int_\Theta S_{E|X,\Theta}(t \mid x^*, \vartheta) f_{\Theta|\mathscr{D}}(\vartheta \mid \mathcal{D})\, d\vartheta \\
&=\; S_{E|X,\mathscr{D}}(t \mid x^*, \mathcal{D}).
\end{aligned}$$

Combining this equality with Equation C.2 gives

$$S_{E|X,\mathscr{D}}(t \mid x^*, \mathcal{D}) \;\xrightarrow[N\to\infty]{\text{a.s.}}\; M_t(x^*, [\theta^*]). \tag{C.3}$$

Without identifiability, this is the strongest possible conclusion: the PPSD converges to the prior-average survival function over DGPs that are observationally equivalent to the truth.

Finally, note π is survival-identifiable at time t. Then, by Definition C.1, all DGPs in the same observational equivalence class have the same survival probability at (t, x^*). Therefore,

$$\begin{aligned}
M_t(x^*, [\theta^*]) \;&=\; \mathbb{E}_\pi\left[S_{E|X,\Theta}(t \mid x^*, \vartheta) \mid [\vartheta] = [\theta^*]\right] \\
&=\; S_{E|X,\Theta}(t \mid x^*, \theta^*).
\end{aligned}$$

Substituting this into Equation C.3 proves Equation C.1. This establishes sufficiency.

For necessity, suppose that the Bayesian PPSD is consistent in the sense of Equation C.1 for every θ^*. From Equation C.3, the same sequence also converges almost surely to $M_t(x^*, [\theta^*])$. By uniqueness of almost-sure limits,

$$S_{E|X,\Theta}(t \mid x^*, \theta^*) \;=\; M_t(x^*, [\theta^*])$$

for every θ^*. Since the right-hand side depends on θ^* only through its observational equivalence class $[\theta^*]$, the survival functional $S_{E|X,\Theta}(t \mid x^*, \theta^*)$ also depends only on $[\theta^*]$. Thus, Definition C.1 holds with $F_t(x^*, [\theta]) = M_t(x^*, [\theta])$, up to arbitrary definition on null sets. Hence the prior is survival-identifiable at time t. □

C.4 Implication for SurvivalPFN

Proposition C.1 characterizes the Bayesian posterior predictive target. SurvivalPFN is a finite neural network trained to approximate this target, so the theorem does not by itself prove finite-sample or finite-capacity consistency of the trained transformer. It does, however, imply consistency in the idealized exact-amortization limit.

Corollary C.1 (Consistency of SurvivalPFN)**.** *Assume the conditions of Proposition C.1. Suppose that, for $\widetilde{\delta}^* = 1$, SurvivalPFN exactly amortizes the Bayesian posterior predictive event-time distribution, so that its predicted survival curve satisfies*

$$\widehat{S}_\omega(t \mid x^*, \mathcal{D}) \;=\; S_{E|X,\mathscr{D}}(t \mid x^*, \mathcal{D}).$$

Then, for every true DGP parameter θ^* and every x,

$$\widehat{S}_\omega(t \mid x^*, \mathcal{D}) \;\xrightarrow[N\to\infty]{\text{a.s.}}\; S_{E|X,\Theta}(t \mid x^*, \theta^*).$$

More generally, the same conclusion holds if the amortization error vanishes:

$$\left|\widehat{S}_\omega(t \mid x^*, \mathcal{D}) - S_{E|X,\mathscr{D}}(t \mid x^*, \mathcal{D})\right| \;\xrightarrow[N\to\infty]{}\; 0.$$

Proof. The exact-amortization case follows immediately by substituting $\widehat{S}_\omega(t \mid x^*, \mathcal{D}) = S_{E|X,\mathscr{D}}(t \mid x^*, \mathcal{D})$ into Proposition C.1. The vanishing-error case follows from the triangle inequality:

$$\begin{aligned}&\left|\widehat{S}_\omega(t \mid x^*, \mathcal{D}) - S_{E|X,\Theta}(t \mid x^*, \theta^*)\right| \\ &\quad\le\; \left|\widehat{S}_\omega(t \mid x^*, \mathcal{D}) - S_{E|X,\mathscr{D}}(t \mid x^*, \mathcal{D})\right| + \left|S_{E|X,\mathscr{D}}(t \mid x^*, \mathcal{D}) - S_{E|X,\Theta}(t \mid x^*, \theta^*)\right|.\end{aligned}$$

The first term vanishes by assumption, and the second term vanishes almost surely by Proposition C.1. □

D Additional SurvivalPFN Details

D.1 Synthetic Prior and DGP Simulation

SurvivalPFN is pretrained on synthetic right-censored survival datasets sampled from a prior $\pi(\cdot)$ over identifiable survival data-generating processes. A draw $\theta \sim \pi(\cdot)$ specifies all random choices needed to generate one synthetic task, including the covariate distribution, event-time mechanism, censoring-time mechanism, censoring type, and target censoring rate.

The simulator returns both the observed survival dataset

$$\mathcal{D}_\theta^{tr} \;=\; \{(x_i, t_i, \delta_i)_\theta\}_{i=1}^N,$$

and the corresponding latent event and censoring times $\{(e_i, c_i)_\theta\}_{i=1}^N$, which are used only for constructing supervised query targets during prior-data training.

Random Tabular Generators. All survival priors are built from a generic tabular generator. We write G for a random table generator that can sample either an unconditional table or a conditional table. Concretely, a call to G first samples generator-specific latent parameters ζ, and then produces either

$$X \;\sim\; G_\zeta(\cdot), \qquad Y \mid X \;\sim\; G_\zeta(\cdot \mid X).$$

based on the requirement.

This abstraction also lets the same survival-prior design use different tabular generators, including PFN-style random multilayer perceptrons, random structural causal models, tree-based generators, or mixtures of these sources.

Censoring Mechanisms. We include several censoring mechanisms to cover common right-censoring regimes while retaining identifiability.

Uniform censoring. Event times are generated from the event-time prior, while censoring times are sampled from a uniform distribution. Depending on the time-generation family, the support is either based on the generated event-time range or on a sampled global time horizon:

$$C_i \sim \mathrm{Unif}\left(\min_j E_j, \max_j E_j\right), \qquad \text{or} \qquad C_i \sim \mathrm{Unif}(0, t_{\max}).$$

Random censoring. Event times are generated conditionally on X, while censoring times are generated from a separate unconditional tabular mechanism:

$$E \mid X \sim \Pr(E \mid X), \qquad C \sim \Pr(C).$$

This produces censoring distributions that are more diverse than simple uniform censoring while remaining independent of the event-time noise conditional on the task.

Administrative censoring. Administrative censoring simulates staggered study entry with a common study end date. The simulator samples entry times A_i, chooses a fixed administrative end time a_*, and sets

$$C_i = a_* - A_i.$$

This captures a common survival-analysis setting in which subjects enter at different calendar times but are all censored at the same study termination date.

Independent censoring. For independent (or covariate-dependent) censoring, both event and censoring times are generated conditional on X, but from independent blocks:

$$E \perp C \mid X, \qquad \Pr(E, C \mid X) = \Pr(E \mid X)\,\Pr(C \mid X).$$

This mechanism is central to the identifiable-prior construction: it allows censoring to depend on covariates while preserving the standard conditional independent censoring assumption used for nonparametric identification.

Prior Family 1: Naive Survival Prior. The simplest prior treats generated table values as raw event and censoring times. Let G_ζ denote a conditional event/censoring table generator. For the event-time branch, we sample

$$E = G_\zeta(X; 1),$$

where $G_\zeta(X; 1)$ denotes one conditional output column given X. The censoring branch is then chosen according to the censoring type: uniform censoring samples C from a uniform distribution; tabular censoring samples C from an unconditional table generator; administrative censoring constructs C from entry times; and the conditional independent branch samples separate event and censoring columns conditional on X. This prior is intentionally simple and flexible: it exposes the model to irregular, nonparametric time patterns without imposing an explicit survival-time family.

Prior Family 2: Survival-Distribution Prior. The second prior generates smooth distributions using random monotone maps. For each dataset, the simulator samples a time horizon $t_{\max} > 0$, a number of knots K, and unconstrained coefficients

$$c = (c_1, \ldots, c_K) \in \mathbb{R}^K.$$

These coefficients are converted into positive increments

$$\Delta_j = \frac{\exp(c_j)}{\sum_{\ell=1}^{K} \exp(c_\ell)}, \qquad j = 1, \ldots, K,$$

which define monotone control points

$$b_0 = 0, \qquad b_j = \sum_{\ell=1}^{j} \Delta_\ell, \qquad j = 1, \ldots, K.$$

The resulting monotone Bernstein map is

$$f_c(u) = \sum_{j=0}^{K} b_j \binom{K}{j} u^j (1-u)^{K-j}, \qquad u \in [0, 1].$$

A raw time is then sampled by pushing uniform noise through this monotone map:

$$\tau \;=\; t_{\max} f_c(U), \qquad U \;\sim\; \mathrm{Unif}(0,1).$$

This construction can be viewed as a random smooth quantile-like function. It gives a flexible family of event-time and censoring-time distributions without committing to a fixed parametric hazard form. Under conditional independent censoring, event and censoring coefficient blocks are generated independently given X:

$$\begin{aligned} E_i &= t_{\max} f_{c_i^E}(U_i^E), \\ C_i &= t_{\max} f_{c_i^C}(U_i^C), \end{aligned}$$

with independent $U_i^E, U_i^C \sim \mathrm{Unif}(0,1)$ and independent coefficient blocks conditional on x_i.

Prior Family 3: Mixture Prior. The third prior samples event and censoring times from mixtures of distributions. In our implementation, the component family is sampled from Weibull and lognormal distributions, and the number of components K_{mix} is sampled from a finite candidate set. For each row i, a table generator emits hidden values

$$h_i \;=\; (a_{i1}, b_{i1}, r_{i1}, \ldots, a_{iK_{\mathrm{mix}}}, b_{iK_{\mathrm{mix}}}, r_{iK_{\mathrm{mix}}}).$$

The logits r_{ij} define mixture weights

$$\pi_{ij} \;=\; \frac{\exp(r_{ij})}{\sum_{\ell=1}^{K_{\mathrm{mix}}} \exp(r_{i\ell})}, \qquad \sum_{j=1}^{K_{\mathrm{mix}}} \pi_{ij} \;=\; 1,$$

and a component index is sampled as

$$Z_i \;\sim\; \mathrm{Categorical}(\pi_{i1}, \ldots, \pi_{iK_{\mathrm{mix}}}).$$

For a Weibull component, positive shape and scale parameters are obtained by

$$\begin{aligned} \kappa_{ij} &= \mathrm{softplus}(a_{ij}) + 0.1, \\ \lambda_{ij} &= \mathrm{softplus}(b_{ij}) + 0.1, \end{aligned}$$

and the sampled time is

$$\tau_i \;=\; \lambda_{iZ_i} \left[-\log(1-U_i)\right]^{1/\kappa_{iZ_i}}, \qquad U_i \;\sim\; \mathrm{Unif}(0,1).$$

For a lognormal component,

$$\begin{aligned} \mu_{ij} &= \mathrm{softplus}(a_{ij}), \\ \sigma_{ij} &= \mathrm{softplus}(b_{ij}), \end{aligned}$$

and

$$\log \tau_i = \mu_{iZ_i} + \sigma_{iZ_i}\varepsilon_i, \qquad \varepsilon_i \sim \mathcal{N}(0,1).$$

This mixture prior provides a complementary source of smooth positive-time distributions with heavy tails and multimodality.

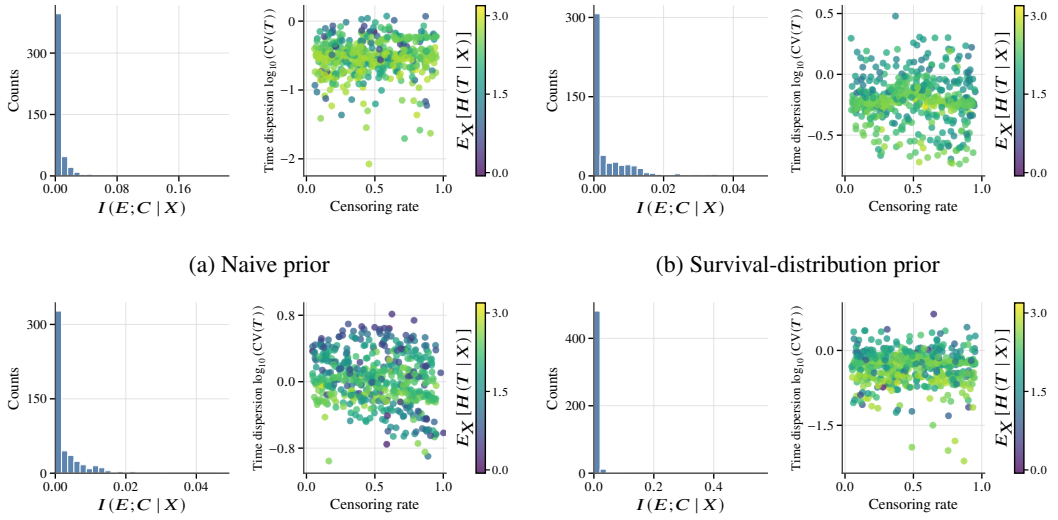
Prior Family 4: Kitchen-Sink Meta Prior. Finally, SurvivalPFN can use a meta-prior that mixes multiple complete survival priors. Let $P_1, \ldots, P_M$ denote child prior generators and let $w_1, \ldots, w_M$ be nonnegative mixture weights with $\sum_m w_m = 1$. The kitchen-sink prior samples

$$M^* \;\sim\; \mathrm{Categorical}(w_1, \ldots, w_M), \qquad \mathcal{D}_\theta^{tr} \;\sim\; P_{M^*}.$$

Equivalently, the induced prior over synthetic datasets is

$$P_{\mathrm{sink}}(\mathcal{D}) \;=\; \sum_{m=1}^{M} w_m P_m(\mathcal{D}).$$

This mixture increases prior diversity by combining direct table-output mechanisms, smooth monotone distributional mechanisms, and positive-time mixture mechanisms. In the current default configuration, the kitchen-sink prior places most mass on the direct table-output and monotone survival-distribution priors, while the exact mixture weights remain configurable.



(a) Naive prior

(b) Survival-distribution prior

(c) Mixture prior

(d) Kitchen-sink prior

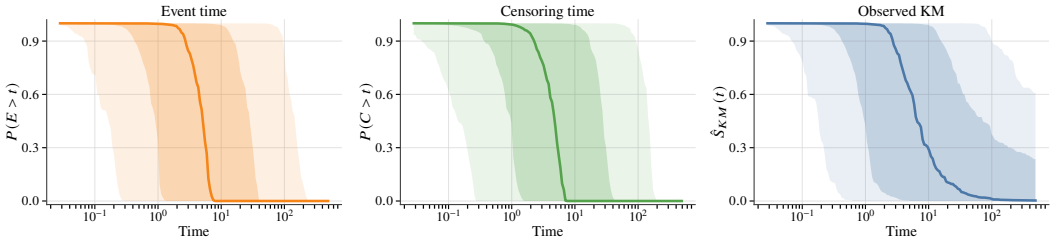
Figure 10: Prior-quality diagnostics across four synthetic prior families, with 500 sampled datasets per family. Each panel summarizes the induced dependence between event and censoring times, censoring-rate coverage, time-scale dispersion, and conditional event-time uncertainty.

Diagnostics for Synthetic DGPs. We further examine the synthetic DGPs induced by each prior family in Figures 10 and 11. These diagnostics are designed to check the two desiderata of the prior: it should remain close to the identifiable independent-censoring regime, while still covering a broad range of survival-data regimes. For each prior family, we sample 500 synthetic datasets, each with 1,024 observations, and compute dataset-level and curve-level summaries.

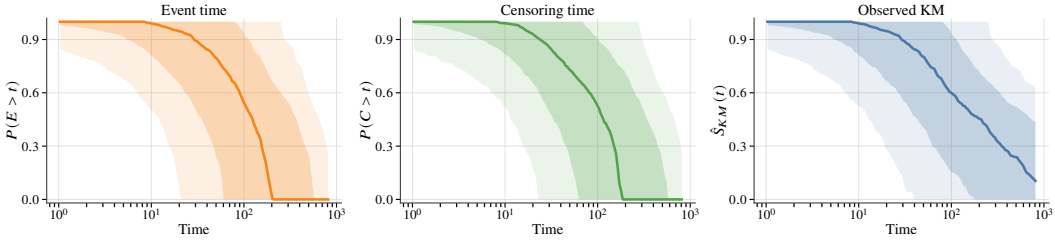
Figure 10 reports scalar diagnostics for each generated dataset. The left subpanel estimates the conditional mutual information $I(E; C \mid X)$ between the latent event time E and censoring time C after conditioning on covariates X. Since our simulator is constructed under conditional independence, values concentrated near zero provide an empirical sanity check that the generated censoring process does not introduce substantial residual event-censoring dependence beyond X. The right subpanel summarizes dataset diversity: each point is one generated dataset, with the x-axis showing the censoring rate, the y-axis showing the observed-time dispersion $\log_{10}(\mathrm{CV}(T))$, where $\mathrm{CV}(T) = \mathrm{std}(T)/|\mathrm{mean}(T)|$ is the coefficient of variation of the observed times, and the color showing the conditional observed-time entropy $\mathbb{E}_X[H(T \mid X)]$. Thus, this panel visualizes how broadly each prior covers censoring rates, relative time-scale variation, and residual outcome uncertainty after conditioning on covariates.

Several trends are apparent from these plots. First, across all four prior families, the estimated conditional mutual information $I(E; C \mid X)$ is highly concentrated near zero, suggesting that the generated datasets largely remain within the intended conditional independent censoring regime. This provides an empirical sanity check that the prior does not typically generate strongly informative censoring structures. Second, the scatter plots show that the priors cover a broad range of right-censored survival regimes. The generated datasets span nearly the full range of censoring rates, from lightly censored to heavily censored settings. They also cover different observed-time dispersion regimes, and different levels of residual conditional uncertainty, as measured by $\mathbb{E}_X[H(T \mid X)]$.

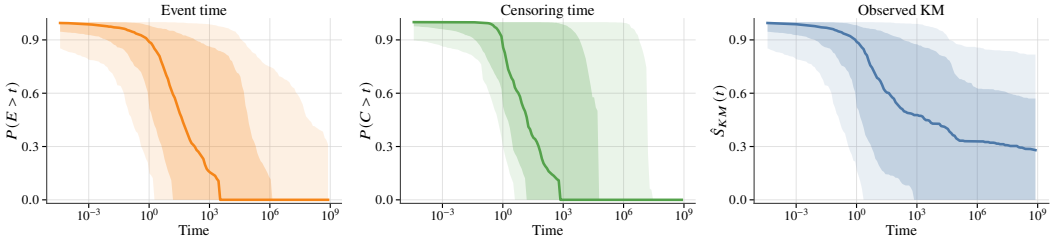
Figure 11 complements these scalar summaries with curve-level diagnostics. For each generated dataset, we compute the empirical latent event-time survival curve $P(E > t)$, the empirical latent censoring-time survival curve $P(C > t)$, and the Kaplan-Meier estimate $\widehat{S}_{\mathrm{KM}}(t)$ from the observed pairs (T, Δ). The solid line denotes the pointwise median curve across generated datasets, while the dark and light bands denote the pointwise 25th-75th and 10th-90th percentile ranges, respectively. These curves show that the priors induce a wide range of event-time, censoring-time, and observed survival shapes, including different time scales, tail behaviors, and censoring-distorted Kaplan-Meier patterns. Together, Figures 10 and 11 show that the proposed prior families generate diverse



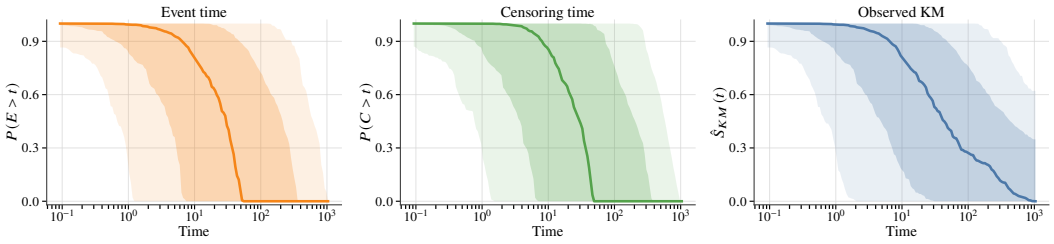
(a) Naive prior



(b) Survival-distribution prior



(c) Mixture-model prior



(d) Kitchen-sink prior

Figure 11: **Empirical distributions across 500 synthetic datasets.** Each panel shows the induced marginal event-time survival curve $P(E > t)$, censoring-time survival curve $P(C > t)$, and observed Kaplan-Meier curve $\widehat{S}_{\mathrm{KM}}(t)$. Solid lines denote the pointwise median curve across generated datasets. Dark shaded bands denote the interquartile range (25th-75th percentiles), and light shaded bands denote the 10th-90th percentile range.

survival tasks while maintaining the identifiable right-censoring structure required by our theoretical framework. These curve-level summaries further confirm that the diversity of DGPs. The empirical distributions vary substantially across generated datasets, with different decay rates, time scales, and tail behaviors.

D.2 Architecture Details

SurvivalPFN uses the same PFN-style transformer architecture as TabDPT [62] and CausalPFN [3], with only task-specific changes to the token contents and output interpretation. Each context row $(x_i, t_i, \delta_i) \in \mathcal{D}_\theta^{tr}$ is represented as a single token by combining embeddings of the covariates x_i, observed time t_i, and event indicator δ_i. Each query row is represented by embeddings of the

query covariate x_θ^* and the query indicator $\widetilde{\delta}^*$. We use linear embedding layers and omit positional encodings, so the context is treated as a set rather than an ordered sequence.

All context and query tokens are passed through a 20-layer transformer encoder with hidden dimension 384, RMS query-key normalization, and parallel SwiGLU feed-forward blocks. The attention mask follows the standard PFN design [37, 38]: context tokens attend to one another, while query tokens attend only to the context. This masking prevents information leakage across queries and allows the model to process many query instances in parallel. The output representation of each query token is projected to $L = 1024$ logits, and a softmax produces a discretized PPD over the transformed time bins:

$$q_\omega(\cdot \mid x_\theta^*, \widetilde{\delta}^*, \mathcal{D}_\theta^{tr}) \;=\; \Big[q_{\omega,\ell}(x_\theta^*, \widetilde{\delta}^*, \mathcal{D}_\theta^{tr})\Big]_{\ell=1}^{L}.$$

Depending on $\widetilde{\delta}^*$, this distribution is interpreted as an approximation to either the PPD for event time or the PPD for censoring time. The corresponding PPSD or PPCD is obtained by summing the predicted tail probability mass across bins.

The full model has approximately 20M parameters and is trained in two stages: (i) a predictive phase that follows standard predictive PFN training from Ma et al. [62], and (ii) a survival phase that optimizes the survival likelihood or cross-entropy loss. We use AdamW [49] with warmup and cosine annealing in the predictive phase, and switch to the schedule-free optimizer [14] in the survival phase. The maximum context length is 16K in the first phase and 2,048 in the second. The predictive phase is trained on four A100 GPUs for up to one week, followed by two and half days of survival-phase training on one H100 GPU.

Training is parallelized over both synthetic context and query tokens. At each gradient step, we sample B_θ independent DGPs from the prior, generate one context table for each DGP, and draw B_q query rows per datasets. These $B_\theta B_q$ query predictions are computed in a single batched forward pass, and the final loss is averaged over all DGP-query pairs. This batching strategy is identical in spirit to the parallel training procedure used in CausalPFN; see Algorithm 1 in Balazadeh et al. [3] for the full procedure.

D.3 Monotone Time Transformations

Survival times are nonnegative and often highly skewed. Directly discretizing raw time can therefore allocate too many bins to sparse tail regions or make the learned distribution sensitive to dataset-specific time scales. To address this, SurvivalPFN applies a context-fitted monotone transformation before discretization.

For each dataset $\mathcal{D} = \{(x_i, t_i, \delta_i)\}_{i=1}^N$, let $g : \mathbb{R}_+ \to \mathcal{Z}$ denote the dataset-specific transformation fitted on the observed times for the context tokens $\{t_i\}_{i=1}^{N_c}$, where N_c is the size of context tokens. The observed times and query targets for context tokens are mapped into model space as

$$\begin{aligned} z_i &= g(t_i), \\ z_e^* &= g(e^*), \\ z_c^* &= g(c^*). \end{aligned}$$

The histogram likelihood Equation 3.3 or cross-entropy loss in Equation D.2 is then evaluated in this transformed space.

To make prediction, model-space bin edges are mapped back to raw time through g^{-1}, so that the final PPSD/PPCD is reported on the original time scale. In all cases, g is monotone increasing, preserving the temporal ordering of events and censoring times.

`lognormal2normal` **Transformation.** The `lognormal2normal` transformation uses a simple parametric normalization motivated by the positivity and right-skewness of survival times. For each dataset, we first calculate the mean m and variance s^2 on the context tokens. We fit a lognormal distribution whose raw-time mean and standard deviation match m and s^2. If $T = \exp(\mu + \sigma Z)$

with $Z \sim \mathcal{N}(0,1)$, then

$$\sigma^2 \;=\; \log\!\left(1 + \frac{s^2}{m^2}\right),$$
$$\mu \;=\; \log(m) - \frac{1}{2}\sigma^2.$$

The forward transformation maps a raw time $t > 0$ to its fitted normal coordinate:

$$g_{\mathrm{LN}}(t) \;=\; \frac{\log t - \mu}{\sigma}.$$

The inverse transformation is

$$g_{\mathrm{LN}}^{-1}(z) \;=\; \exp(\mu + \sigma z).$$

Thus, fixed-width bins in model space correspond to adaptive, nonuniform bins in raw time. This transformation is smooth, bijective on $\mathbb{R}_+$, preserves time ordering, and can extrapolate beyond the largest observed time. Its main limitation is that it imposes a lognormal shape on the observed-time distribution; this parametric normalization may not allocate resolution optimally when the parametric form is incorrect.

`time2quantile` **Transformation.** The `time2quantile` transformation is a nonparametric, context-adaptive alternative. It maps raw time into an empirical quantile coordinate in $[0,1]$. For all the context tokens in the dataset, sort the observed times and collapse duplicates to obtain unique knots

$$0 \;=\; a_0 \;<\; a_1 \;<\; \cdots \;<\; a_K \;=\; \max_i t_i.$$

Each knot a_j is assigned its right-continuous empirical CDF value

$$q_j \;=\; \widehat{F}(a_j) \;=\; \frac{1}{N_c}\sum_{i=1}^{N_c} \mathbb{1}\{t_i \le a_j\}, \qquad q_0 \;=\; 0, \quad q_K \;=\; 1.$$

The forward transformation is the piecewise-linear interpolation of these knots:

$$g_{\mathrm{Q}}(t) \;=\; q_j + \frac{t - a_j}{a_{j+1} - a_j}(q_{j+1} - q_j), \qquad a_j \;\le\; t \;\le\; a_{j+1}.$$

Values above the largest time are mapped to 1. The inverse transformation swaps the axes and linearly interpolates from quantile space back to raw time:

$$g_{\mathrm{Q}}^{-1}(q) \;=\; a_j + \frac{q - q_j}{q_{j+1} - q_j}(a_{j+1} - a_j), \qquad q_j \;\le\; q \;\le\; q_{j+1}.$$

For this transformation, the model-space range is fixed to $[0,1]$. Uniform bins in quantile space become adaptive raw-time bins: regions with many observed times receive finer resolution, while sparse regions receive wider bins.

The `time2quantile` transformation makes no parametric assumption on the time distribution and is robust to changes in time units or monotone rescalings of raw time. However, because it is fitted from the empirical distribution of $\{t_i\}_{i=1}^{N_c}$, it is context-local and cannot resolve the tail shape beyond $\max_i t_i$; all larger times are mapped to quantile 1, with probability mass beyond this point represented only by the final residual histogram bin.

Summary. The two transformations reflect complementary design choices. The `lognormal2normal` transformation provides a smooth positive-time coordinate with parametric tail extrapolation, while `time2quantile` provides a fully context-adaptive coordinate that normalizes all tasks to the common interval $[0,1]$. Both transformations preserve time ordering and allow SurvivalPFN to allocate discretization resolution more effectively than raw-time binning.

D.4 Training Objective

SurvivalPFN is trained to predict a discretized PPD over transformed time. For a query with indicator $\widetilde{\delta}^*$, define the latent supervised target

$$r^*_\theta(\widetilde{\delta}^*) \;=\; \widetilde{\delta}^* e^*_\theta + \big(1-\widetilde{\delta}^*\big)c^*_\theta.$$

Thus, $\widetilde{\delta}^* = 1$ asks for the latent event time, whose posterior predictive tail gives the PPSD, while $\widetilde{\delta}^* = 0$ asks for the latent censoring time, corresponding to the PPCD.

Let $g_{\mathcal{D}^{tr}_\theta} : \mathbb{R}_+ \to \mathcal{Z}$ be the monotone transformation fitted from the observed context times in $\mathcal{D}^{tr}_\theta$, and let $\{\mathcal{I}_\ell\}_{\ell=1}^{L}$ denote the ordered bins in transformed-time space. Define the bin-index map

$$\kappa_{\mathcal{D}^{tr}_\theta}(r) \;=\; \ell \quad \text{if} \quad g_{\mathcal{D}^{tr}_\theta}(r) \in \mathcal{I}_\ell,$$

with boundary clipping handled by the same convention used in implementation. Equivalently, define the one-hot target

$$b_{\ell,\mathcal{D}^{tr}_\theta}(r) \;=\; \mathbb{I}\Big\{\kappa_{\mathcal{D}^{tr}_\theta}(r) = \ell\Big\}, \qquad \ell = 1,\dots,L.$$

Likelihood loss. The main SurvivalPFN checkpoint is trained with the one-hot discrete negative log-likelihood over transformed-time bins:

$$\mathrm{NLL}(r^*_\theta \,\|\, q_\omega) \;=\; -\log q_{\omega,\kappa_{\mathcal{D}^{tr}_\theta}(r^*_\theta)}\Big(x^*_\theta, \widetilde{\delta}^*, \mathcal{D}^{tr}_\theta\Big).$$

Equivalently,

$$\mathrm{NLL}(r^*_\theta \,\|\, q_\omega) \;=\; -\sum_{\ell=1}^{L} b_{\ell,\mathcal{D}^{tr}_\theta}(r^*_\theta)\log q_{\omega,\ell}\Big(x^*_\theta, \widetilde{\delta}^*, \mathcal{D}^{tr}_\theta\Big).$$

The corresponding population objective is

$$\mathcal{L}_{\mathrm{NLL}}(\omega) \;=\; \mathbb{E}_{\theta\sim\pi(\cdot)}\,\mathbb{E}_{\mathcal{D}^{tr}_\theta,\, x^*_\theta,\, e^*_\theta,\, c^*_\theta,\, \widetilde{\delta}^*}\left[-\log q_{\omega,\kappa_{\mathcal{D}^{tr}_\theta}(r^*_\theta(\widetilde{\delta}^*))}\Big(x^*_\theta, \widetilde{\delta}^*, \mathcal{D}^{tr}_\theta\Big)\right]. \tag{D.1}$$

This is the objective used by the best validation checkpoint reported in the main results.

Smoothed cross-entropy variant. We also consider a smoothed cross-entropy loss, which is used in TabDPT [62] and CausalPFN [3]. Instead of assigning all mass to one bin, the target time is converted into a narrow histogram in transformed-time space:

$$a^{(\sigma)}_{\ell,\mathcal{D}^{tr}_\theta}(r) \;=\; \int_{\mathcal{I}_\ell} \alpha_\sigma\Big(z;\, g_{\mathcal{D}^{tr}_\theta}(r)\Big)\,dz, \qquad \ell = 1,\dots,L,$$

where α_σ is a narrow smoothing density, implemented as a Gaussian centered at $g_{\mathcal{D}^{tr}_\theta}(r)$ with a predefined variance. The smoothed cross-entropy loss is

$$\mathrm{SCE}_\sigma(r^*_\theta \,\|\, q_\omega) \;=\; -\sum_{\ell=1}^{L} a^{(\sigma)}_{\ell,\mathcal{D}^{tr}_\theta}(r^*_\theta)\log q_{\omega,\ell}\Big(x^*_\theta, \widetilde{\delta}^*, \mathcal{D}^{tr}_\theta\Big). \tag{D.2}$$

As $\sigma \to 0$, the smoothed target $a^{(\sigma)}_{\ell,\mathcal{D}^{tr}_\theta}(r)$ reduces to the one-hot target $b_{\ell,\mathcal{D}^{tr}_\theta}(r)$, and the smoothed cross-entropy recovers the discrete NLL. In our experiments, this loss is treated as an alternative training option and evaluated in the ablation study.

Note. This prior-data likelihood differs from the standard right-censored survival likelihood in Section 2. The standard observed-data likelihood trains on partially observed query outcomes (t^*, δ^*) and must account for censored queries through survival-tail probabilities. In contrast, SurvivalPFN training uses simulator-provided latent query times e^*_θ or c^*_θ as supervision.

D.5 Inference Procedure

At inference time, SurvivalPFN takes an observed survival dataset $\mathcal{D}$ as context and one or more query covariates x^*. For survival prediction, we set the query indicator to $\widetilde{\delta}^* = 1$, so that the model outputs a discretized posterior predictive event-time distribution

$$q_\omega\Big(\cdot \,\Big|\, x^*, \widetilde{\delta}^* = 1, \mathcal{D}\Big) \;=\; \Big[q_{\omega,\ell}(x^*, \widetilde{\delta}^* = 1, \mathcal{D})\Big]_{\ell=1}^{L}.$$

This requires only a single forward pass and does not involve dataset-specific gradient updates, posterior sampling, or hyperparameter tuning. Let $g_{\mathcal{D}}$ denote the monotone time transformation fitted from the observed times in $\mathcal{D}$, and let $\{\mathcal{I}_\ell\}_{\ell=1}^{L}$ denote the discretized bins in the transformed time space. The predicted posterior predictive survival distribution (PPSD) is obtained by summing the probability mass assigned to bins whose raw-time support lies above t:

$$\widehat{S}_\omega(t \mid x^*, \mathcal{D}) \;=\; \sum_{\ell:\, g_{\mathcal{D}}^{-1}(\mathcal{I}_\ell) > t} q_{\omega,\ell}\Big(x^*, \widetilde{\delta}^* = 1, \mathcal{D}\Big),$$

which is implemented as a cumulative tail sum over the discretized time bins. Similarly, setting $\widetilde{\delta}^* = 0$ yields the PPCD (although we generally do not care about PPCD during inference).

E Benchmarking Details

E.1 Baseline Model Details

In this study, we compare SurvivalPFN against a board range of 20 representative survival analysis methods spanning classical statistical models, tree-based ensembles, discrete-time neural survival models, continuous-time neural survival models, and methods using TFMs. Below, we briefly summarize each baseline, including its core modeling idea and the implementation details used in our benchmark. A side-to-side overview of their methodological properties is provided in Table 2.

For each column in the table, *continuous-time* indicates whether the method explicitly parameterizes a smooth survival distribution over continuous time, without relying on post-hoc interpolation over a discrete time grid. *(Semi-)parametric* indicates whether the method specifies the hazard, density, or survival function through an explicit parametric or semi-parametric form. The *proportional hazards (PH)* assumption indicates that covariates act multiplicatively on a shared baseline hazard, so that hazard ratios between individuals are constant over time. Finally, the *ensemble/mixture* column indicates whether the method combines a finite collection of base learners, components, or experts, such as trees in an ensemble or distributions in a finite mixture model.

In particular, SurvivalMDN is marked parametric because it uses a finite mixture of Gaussians, and DSM is marked parametric because it uses a finite mixture of Weibulls; only an infinite-mixture construction would be nonparametric.

Table 2: Qualitative comparison of survival models considered in our benchmark. Columns indicate whether a method explicitly parameterizes a continuous-time (cont. time) survival distribution, imposes a (semi-)parametric form, assumes proportional hazards (PH), uses an ensemble or finite-mixture construction, or belongs to the tabular foundation model (TMF) family.

Model	**Cont. time**	**(Semi-) parametric**	**PH assump.**	**Ensemble/ mixture**	**TFM**	**Distinguishing feature**
Prior-fitted and tabular foundation models						
SurvivalPFN	✗	✗	✗	✗	✓	Novel in-context Bayesian survival estimator.
StaticSurvivalTFM	✗	✗	✗	✗	✓	Handling right-censoring with tabular foundation models.
Classical survival models						
CoxPH	✗	✓	✓	✗	✗	Semi-parametric Cox proportional hazards model.
CoxNet	✗	✓	✓	✗	✗	Regularized Cox proportional hazards model.
cSVR	✗	✗	✗	✗	✗	Censored support-vector regression baseline.
Tree-based models						
GB	✗	✓	✓	✓	✗	Gradient-boosted Cox proportional hazards model.
CWGB	✗	✓	✓	✓	✗	Censoring-weighted gradient-boosted Cox model.
RSF	✗	✗	✗	✓	✗	Random survival forest for nonlinear effects and interactions.
Neural discrete-time models						
DeepHit	✗	✗	✗	✗	✗	Neural discrete-time model emphasizing discrimination.
DeepSurv	✗	✓	✓	✗	✗	Neural Cox proportional hazards model.
MTLR	✗	✗	✗	✗	✗	Multi-task logistic regression for discrete-time survival.
Nnet-survival	✗	✗	✗	✗	✗	Discrete-time neural survival model.
CoxTime	✗	✓	✗	✗	✗	Neural Cox model with time-varying effects.
IWSG	✗	✗	✗	✗	✗	Explicitly models the censoring mechanism.
CQRNN	✗	✗	✗	✗	✗	Quantile-regression survival baseline.
BNN-MTLR	✗	✗	✗	✗	✗	Bayesian neural extension of MTLR.
Neural continuous-time models						
DSM	✓	✓	✗	✓	✗	Finite parametric (Weibull) mixture components.
SumoNet	✓	✗	✗	✗	✗	Continuous survival-function via automatic differentiation.
SurvivalMDN	✓	✓	✗	✓	✗	Finite parametric (Gaussian) mixture components.
DeepAFT–Weibull	✓	✓	✓	✗	✗	Continuous Weibull accelerated failure time model.
DeepAFT–Log-logistic	✓	✓	✗	✗	✗	Continuous log-logistic accelerated failure time model.

1. **SurvivalPFN**. SurvivalPFN is the model we developed in this paper. See description in Section 3 and Appendix D.
2. **StaticSurvivalTFM** [47]. StaticSurvivalTFM is a framework that converts any binary classifier into a survival predictor; we provide methodological details in Appendix H. For a fair comparison with SurvivalPFN, we use TabDPT [62] as the backbone binary classifier in the main benchmark. We further study the effect of replacing this backbone with MITRA [107] (the reported best version in Kim et al. [47]), with results reported in Appendix G.4.
3. **Cox proportional hazards model (CoxPH)** [12]. CoxPH is the classical semi-parametric proportional-hazards model, estimating covariate effects through the Cox partial likelihood while leaving the baseline hazard unspecified. We used the `scikit-survival` implementation [78], which produces risk scores and can recover survival curves using an estimated baseline hazard [6].
4. **Elastic-net Cox proportional hazards model (CoxNet)** [95]. CoxNet regularizes the Cox partial likelihood with an elastic-net penalty, making Cox-style modeling more stable in high-dimensional settings. We used the `scikit-survival` implementation [78], with the same baseline hazard estimator as CoxPH.
5. **Censored Support Vector Regression (cSVR)** [79]. cSVR extends support-vector regression to right-censored outcomes by treating censored observations through inequality constraints or ranking-style losses. Specifically, we use the `FastSurvivalSVM` method in `scikit-survival` [78]. Because cSVR only outputs a scalar regression time rather than a full survival distribution, it cannot produce valid values for evaluation metrics that require an entire survival curve or event-time distribution.
6. **Gradient-boosted Cox model (GB)** [87]. GB fits an additive risk model by gradient boosting under a Cox partial-likelihood objective. We used the `scikit-survival` implementation [78], which yields a boosted risk score and survival estimates through the fitted baseline hazard as CoxPH.
7. **Component-wise gradient-boosted Cox model (CWGB)** [41]. CWGB is a component-wise variant of gradient-boosted Cox regression, where weak learners update individual covariate components under a Cox-style objective. We used the `scikit-survival` implementation [78].
8. **Random Survival Forests (RSF)** [42]. RSF is an ensemble of survival trees that partitions the feature space using survival-specific split criteria and averages terminal-node survival estimates across trees. We use the `scikit-survival` implementation [78].
9. **DeepHit** [55]. DeepHit is a discrete-time neural survival model that directly parameterizes the probability mass function over event-time bins. We used the `pycox` implementation; its objective combines a likelihood term with a CI-like ranking term, explicitly encouraging discrimination.
10. **DeepSurv** [46]. DeepSurv replaces the linear predictor in CoxPH with a neural network while retaining the Cox partial-likelihood objective and proportional-hazards structure. We reimplemented it following the public PyTorch implementation (`https://github.com/czifan/DeepSurv.pytorch`) and add baseline hazard estimation as in CoxPH.
11. **Multi-task Logistic Regression (MTLR)** [104, 19]. MTLR parameterizes the survival distribution over a discrete time grid using a sequence of dependent logistic regressors. We reimplemented MTLR following the public PyTorch implementation (`https://github.com/mkazmier/torchmtlr`).
12. **Nnet-survival** [5, 23]. Nnet-survival models discrete-time conditional hazards with a neural network and trains by a binary cross-entropy-style survival likelihood over time intervals. We used the `pycox` implementation, converting predicted discrete hazards into survival curves for distributional evaluation.
13. **CoxTime** [54]. CoxTime generalizes neural Cox regression by allowing the relative risk to depend on both covariates and time, thereby relaxing the proportional-hazards assumption. We used the `pycox` implementation and recovered survival curves from the learned time-dependent risk function and the same baseline hazard estimator as CoxPH.

14. **Inverse-Weighted Survival Games (IWSG)** [32]. IWSG explicitly models both the failure-time and censoring distributions through an inverse-probability-of-censoring-weighted (IPCW) game objective. We reimplemented it from the official IWSG codebase (`https://github.com/rajesh-lab/Inverse-Weighted-Survival-Games`).
15. **Censored Quantile Regression Neural Network (CQRNN)** [74]. CQRNN directly predicts event-time quantiles under censoring, providing a distribution-free way to represent time-to-event uncertainty. We reimplemented it following the official CQRNN codebase (`https://github.com/TeaPearce/Censored_Quantile_Regression_NN`) and converted the predicted quantile function into a monotone survival curve when distributional metrics (IBS and D-calibration) were required.
16. **Bayesian Neural Network Multi-task Logistic Regression (BNN-MTLR)** [82]. BNN-MTLR extends MTLR with Bayesian neural-network uncertainty, producing a PPD over discrete survival curves. We reimplemented it from the official BNN-MTLR codebase (`https://github.com/shi-ang/BNN-ISD`).
17. **Deep Survival Machines (DSM)** [69]. DSM parameterizes the event-time distribution as a finite mixture of Weibull components, with neural networks producing mixture weights and component parameters. We reimplemented DSM using the code in `auton-survival` package [70].
18. **Survival Monotonic Network (SuMoNet)** [88]. SuMoNet models continuous-time survival distributions with monotonic neural networks, using automatic differentiation to obtain valid densities from the learned cumulative distribution function. We reimplemented it following the official SuMoNet codebase (`https://github.com/MrHuff/Sumo-Net`).
19. **Survival Mixture Density Network (SurvivalMDN)** [33]. SurvivalMDN models the event-time distribution using a finite mixture density network, providing a flexible but still finite-dimensional parametric mixture representation. We reimplemented it from the official SurvivalMDN codebase (`https://github.com/XintianHan/Survival-MDN`).
20. **Neural Weibull accelerated failure-time model (DeepAFT-Weibull)** [72]. DeepAFT-Weibull uses a neural network to parameterize a Weibull accelerated failure-time distribution for right-censored data. We reimplemented the model following the paper.
21. **Neural log-logistic accelerated failure-time model (DeepAFT-Loglogistic)** [72]. DeepAFT-Loglogistic similarly uses a neural network to parameterize a log-logistic accelerated failure-time distribution. We reimplemented the model following the paper.

E.2 Unified Time Grid for Consistent Evaluation

All methods in our benchmark are evaluated through a common prediction object. After fitting, each model is converted into a survival-probability matrix

$$\widehat{\mathbf{S}} \;=\; \left[\widehat{S}_i(t_j)\right]_{i=1,\dots,n_{\text{test}};\, j=1,\dots,m}, \qquad \mathcal{G} \;=\; \{t_1 < \cdots < t_m\},$$

where $\widehat{S}_i(t_j)$ denotes the predicted survival probability for test individual i at time t_j. All metrics are then computed from $(\mathcal{G}, \widehat{\mathbf{S}})$. Thus, differences across methods enter only through how their native time representation is constructed before prediction.

To avoid evaluation leakage, all time grids are defined using only the data available during training; test-set event times are never used to define the evaluation support.

For models that require a predefined binning for making discrete-time survival function prediction, including MTLR, DeepHit, Nnet-survival, BNN-MTLR and StaticSurvivalTFM, we use an event-time quantile grid. Let

$$\mathcal{T}_E \;=\; \{T_i : \delta_i = 1,\; i \in \mathcal{D}^{tr}\}$$

denote the uncensored event times in the fitting data. When the number of bins is not specified, we set

$$K \;=\; \left\lceil \sqrt{|\mathcal{T}_E|} \right\rceil,$$

and define the discrete support by the unique empirical quantiles

$$\mathcal{G}_{\text{disc}} \;=\; \text{unique}\left\{ Q_{\mathcal{T}_E}\left(\frac{k-1}{K-1}\right) :\; k = 1, \ldots, K \right\},$$

where $Q_{\mathcal{T}_E}$ is the empirical quantile function. This grid provides the native discretization on which discrete-time models are trained, with each bin contains the exact same number of uncensored instances.

For DeepHit and Nnet-survival, the implementation requries that the first bin location must smaller than the smallest time in the data. Therefore, we additionally shift the first bin location:

$$b_1 \;\leftarrow\; \max\left\{ \min_{i \in \mathcal{D}^{tr}} \{T_i - \epsilon\},\; 0 \right\}, \qquad \epsilon = 10^{-5}.$$

Continuous-time models do not require a training-time discretization. Nevertheless, curve-based evaluation still requires a finite query set. For these methods, we define the evaluation support from the observed fitting durations:

$$\mathcal{G}_{\text{cont}} \;=\; \{0\} \cup \text{unique}\{T_i :\; i \in \mathcal{D}^{tr}\}, \tag{E.1}$$

with duplicate zeros removed if necessary. For quantile-output models such as CQRNN, predicted quantile functions are first converted to survival probabilities on the same support before metric computation.

This protocol preserves each model class's natural time parameterization: discrete-time methods learn on event-time quantile bins, whereas continuous-time methods are queried on the empirical training-time support.

E.3 Hyperparameter Tuning

We tune hyperparameters only for neural-network baselines whose performance depends on optimizer and architecture choices. Classical `scikit-survival` models (CoxPH, CoxNet, cSVR, GB, CWGB, and RSF), StaticSurvivalTFM, and SurvivalPFN are kept fixed at their benchmark settings.

For each outer split r, hyperparameter selection is performed using only the training-side data,

$$\mathcal{D}^{tr+val}_{(r)} \;=\; \mathcal{D}^{tr}_{(r)} \cup \mathcal{D}^{val}_{(r)},$$

and the test set is never used for either tuning or discretization. For each model m, we sample $R = 10$ configurations from its search space Λ_m and estimate their performance by shuffled $F = 5$-fold cross-validation on $\mathcal{D}^{tr+val}_{(r)}$, using the outer split seed for reproducibility. The selected configuration is

$$\lambda^*_{m,(r)} \;=\; \arg\min_{\lambda \in \{\lambda_1, \ldots, \lambda_R\}} \frac{1}{F} \sum_{f=1}^{F} \mathcal{L}^{val}_{(f)}(\lambda; m),$$

where $\mathcal{L}^{val}_{(f)}$ denotes the model-specific objective loss on the validation on fold f. Configurations that fail to complete training are treated as infeasible and assigned infinite validation loss. After selection, model m is refit on $\mathcal{D}^{tr+val}_{(r)}$ using $\lambda^*_{m,(r)}$, and evaluated once on the held-out test split.

The full model-specific search spaces are summarized in Table 3. All remaining optimization settings are fixed across tuning trials: AdamW optimizer, batch size 256, ReLU activations, no normalization layer, early stopping, and a maximum budget of $10{,}000$ epochs. For models that require an explicit learning-rate floor, we set

$$\eta_{\min} \;=\; 10^{-3}\eta,$$

where η is the tuned learning rate. Model-specific constants not included in Table 3 are fixed throughout tuning.

Table 3: Hyperparameter search spaces for tuned neural baselines. The table defines the shared base space and model-specific extensions.

Models	**Hyperparameter meanings**	**Search space**
Base hyperparameters		

DeepHit
DeepSurv
MTLR
Nnet-survival
CoxTime
DeepAFT-Weibull
DeepAFT-Loglogistic
	`lr`: learning rate;
	`weight_decay`: weight decay;
	`neurons`: hidden architecture;
	`dropout`: dropout probability
		{
		`lr`: $\{10^{-4}, 10^{-3}, 10^{-2}\}$;
		`weight_decay`: $\{10^{-3}, 10^{-2}, 10^{-1}\}$;
		`neurons`: $\{[64], [64, 32], [64, 64, 16], [32], [32, 16], [32, 32, 16], [16], [16, 8], [8], []\}$;
		`dropout`: $\{0.0, 0.4, 0.6\}$
		}

Model-specific hyperparameters		
CQRNN	`n_quantiles`: number of predicted quantile levels.	{ Base; `n_quantiles`: $\{9, 19, 39\}$ }
SumoNet	`neurons_alter`: hidden architecture for the censoring branch.	{ Base; `neurons_alter` = `neurons` }
IWSG	`neurons_alter`: hidden architecture for the censoring branch.	{ Base; `neurons_alter` = `neurons` }
SurvivalMDN	`n_mixtures`: number of mixture components.	{ Base; `n_mixtures`: $\{3, 5, 10, 50\}$ }
DSM	`n_mixtures`: number of mixture components.	{ Base; `n_mixtures`: $\{3, 5, 10, 50\}$ }
BNN-MTLR	`pi`: prior mixture probability.	{ Base; `pi`: $\{0.2, 0.5, 0.8\}$ }

E.4 Evaluation Metrics

We evaluate the model's performance on the held-out test set $\mathcal{D}^{te}$. For a fitted model, let $\widehat{S}_{E|X}(u \mid x_i)$ denote the predicted event-time survival function for subject i. When a point prediction is required, we use the predicted median survival time

$$\widehat{e}_i \;=\; \inf\Big\{u : \widehat{S}_{E|X}(u \mid x_i) \le 1/2\Big\},$$

and define the corresponding risk score as

$$\widehat{r}_i \;=\; -\widehat{e}_i,$$

so that higher risk corresponds to shorter predicted survival.

Concordance Index. Discrimination measures whether a model correctly orders subjects by risk. We use Harrell's concordance index [34], computed over comparable pairs in which subject i experiences an observed event before subject j:

$$\mathrm{CI} \;=\; \frac{\sum_{i=1}^{n}\sum_{j\neq i}\delta_i\,\mathbb{1}[t_i < t_j]\,\mathbb{1}\big[\widehat{r}_i > \widehat{r}_j\big]}{\sum_{i=1}^{n}\sum_{j\neq i}\delta_i\,\mathbb{1}[t_i < t_j]}.$$

Higher CI values indicate better performance.

Integrated Brier Score. The Brier score evaluates probabilistic accuracy at a target time u. Because the event status at u may be unknown for subjects censored before u, we use inverse-probability-of-censoring weighting (IPCW) [27, 90]. Let $\widehat{S}_C$ denote the Kaplan-Meier estimate of the marginal censoring survival function, estimated from the training-side data. The IPCW Brier score is

$$\mathrm{BS}(u) \;=\; \frac{1}{n}\sum_{i=1}^{n}\left[\frac{\delta_i\mathbb{1}[t_i \le u]\cdot\widehat{S}_{E|X}(u \mid x_i)^2}{\widehat{S}_C(t_i)} + \frac{\mathbb{1}[t_i > u]\left(1-\widehat{S}_{E|X}(u \mid x_i)\right)^2}{\widehat{S}_C(u)}\right].$$

The integrated Brier score averages this quantity over an evaluation horizon τ:

$$\mathrm{IBS} \;=\; \frac{1}{\tau}\int_0^{\tau} \mathrm{BS}(u)\,du. \tag{E.2}$$

In our experiments, τ is chosen from the training-side event-time support, so that the evaluation horizon is not determined by the held-out test set. Lower IBS indicates better performance.

Mean Absolute Error. To evaluate point prediction of event time, we use the mean absolute error based on pseudo-observations (MAE-PO) [81]. For uncensored subjects, the observed event time t_i can be used directly. For censored subjects, it constructs a pseudo-observation $\widetilde{e}_i$ that estimates the subject's contribution to the marginal Kaplan-Meier survival curve, and then weights the corresponding error by a confidence weight w_i. The resulting error takes the form

$$\text{MAE-PO} \;=\; \frac{\sum_{i=1}^{n} w_i \,|\widehat{e}_i - \widetilde{e}_i|}{\sum_{i=1}^{n} w_i}.$$

This produces an event-time error metric that can use both uncensored and censored test subjects. Lower MAE value indicates better performance.

Distribution Calibration. Distribution calibration (D-calibration) evaluates whether the predicted survival distribution is calibrated over the full time axis [30]. For an uncensored subject, define the probability integral transform value

$$u_i \;=\; \widehat{S}_{E|X}(t_i \mid x_i).$$

If the predicted survival distributions are calibrated, then $\{u_i : \delta_i = 1\}$ should follow a standard uniform distribution on $[0, 1]$. In practice, we partition the probability range $[0, 1]$ into 10 equal-width bins. Uncensored subjects contribute to the bin containing $\widehat{S}_{E|X}(t_i \mid x_i)$. For censored subjects, the event time is only known to satisfy $e_i > t_i$, so their contribution is "blurred" over the portion of the probability scale consistent with this information, namely $\left[0, \widehat{S}_{E|X}(t_i \mid x_i)\right]$. D-calibration is then assessed by the chi-square statistic over the numbers in all the bins. We report the chi-square statistic for each experiments, a smaller statistic indicates less evidence against distribution calibration.

Log-rank Reliability Test. We also use a log-rank goodness-of-fit test to assess whether predicted event times are statistically aligned with the observed time-to-event data. The test compares the observed test sample

$$\mathcal{A}_{\mathrm{obs}} \;=\; \{(t_i, \delta_i)\}_{i=1}^{n}$$

with the predicted event-time sample

$$\mathcal{A}_{\mathrm{pred}} \;=\; \left\{\left(\widehat{e}_i, \widehat{\delta}_i = 1\right)\right\}_{i=1}^{n},$$

where predicted median survival times are treated as uncensored event-time predictions. We report the log-rank statistic for each experiments, a smaller statistic indicate closer agreement between the predicted event-time distribution and the observed test outcomes.

All metrics are computed using the `SurvivalEVAL` package [83].

E.5 Compute Environment and Runtime Protocol

All benchmark experiments were executed on a Slurm-managed GPU cluster. Each experiment was submitted as an independent job with a fixed resource budget: one NVIDIA L40S GPU (48 GB memory), one CPU core from Intel Xeon Gold 6448Y, 16 GB of system memory, and a maximum wall-clock time of 72 hours. The same resource allocation was used across models and datasets for fair evaluation.

The 72-hour wall-clock limit includes all computation required for a benchmark run, including model fitting, hyperparameter tuning when enabled, final refitting, prediction on the held-out test set, and metric computation. If a run did not complete successfully within this budget, we treated the corresponding model-dataset run as failure. Failed runs were deemed as the worst (or equally worst if multiple models failed on this dataset) in the ranking procedure described in Appendix E.6.

E.6 Performance Reporting Protocol

We evaluate each model on repeated random train/validation/test splits. Each dataset is evaluated over $R = 10$ repetitions. The experiment seed is set from 0 to 9 to these repetition, to support data split and model initialization (if needed).

For each split r, the dataset is divided into a 70% $\mathcal{D}_{(r)}^{tr+val}$ and a 30% $\mathcal{D}_{(r)}^{te}$. We compute performance scores on the held-out test set via the metrics described in Appendix E.4. We report the mean and standard deviation across 10 split.

For dataset-level comparisons, each model is ranked separately within each dataset and metric. Rank 1 is assigned to the best value, with ties receiving the minimum tied rank. If a model does not produce a valid result for a given dataset-metric pair, it is assigned one rank below the worst completed method for that pair.

The cSVR baseline produces only a scalar event-time prediction rather than a full survival distribution. Consequently, distributional metrics such as IBS and D-calibration are not applicable to cSVR on any dataset.

Across the full benchmark, we evaluated 21 models on 71 datasets, yielding $21 \times 71 = 1491$ model-dataset runs. Among these, 39 runs failed to produce valid results, corresponding to a failure rate of 2.62%. We summarize the failures below.

- **CoxPH** failed on 21 datasets: MCLcleaned, AML_Bull, NSBCD, gse4335, micro.censure, gse1992, nki70, stagec, gse3143, BMT, cancer, DLBCL, DBCD, zinc, BCCardiotox, vlbw, PDM, actg, METABRIC, AIDS, and hdfail. In all cases, the failure was caused by a singular Hessian matrix of the Cox partial log-likelihood, which prevented the Newton-Raphson optimizer from computing a valid update.[2]
- **DeepSurv** failed on 2 dataset, MCLcleaned and PDM. The fitted baseline hazard produced a degenerate survival curve equal to 1 at all time points, resulting in identical predictions for all test instances.
- **CoxTime** failed on 3 dataset, AML_Bull, DBCD, and FRTCS, because the predicted survival matrix contained NaN values.
- **IWSG** failed on 5 datasets: SUPPORT, MIMIC-IV_all, hdfail, SEER_brain, and SEER_liver. In all cases, the model did not complete training within the 72-hour wall-clock limit.
- **DSM** failed on 3 datasets: FRTCS, NPC, and MIMIC-IV_all. For FRTCS and NPC, the predicted survival matrix contained NaN values. For MIMIC-IV_all, the run did not complete within the 72-hour wall-clock limit.
- **SurvivalMDN** failed on 3 datasets: SEER_liver, SEER_brain, and hdfail. On hdfail, the run exceeded the available 48 GB GPU memory. On SEER_liver and SEER_brain, the model did not complete within the 72-hour wall-clock limit.
- **DeepAFT-Weibull** failed on 1 dataset, FRTCS, because the model did not converge during training.
- **DeepAFT-Loglogistic** failed on 1 dataset, FRTCS, because the model did not converge during training.

Let $\mathrm{rank}_{m,\mathcal{D},q}$ denote the rank of model m on dataset $\mathcal{D}$ for metric q. For each model and metric, we summarize performance by the median rank across datasets,

$$\widetilde{r}_{m,q} \;=\; \mathrm{median}_{\mathcal{D}}\, \mathrm{rank}_{m,\mathcal{D},q}\,.$$

To quantify uncertainty in this median rank, we use a nonparametric bootstrap over datasets. Specifically, for each bootstrap replicate $b = 1, \ldots, B$, we sample datasets with replacement from the benchmark collection, recompute the median rank,

$$\widetilde{r}_{m,q}^{(b)} \;=\; \mathrm{median}_{\mathcal{D}\in\mathcal{B}^{(b)}}\, \mathrm{rank}_{m,\mathcal{D},q},$$

[2]This issue can often be mitigated by adding regularization. However, because we include CoxNet as the regularized Cox baseline, we intentionally keep CoxPH as the conventional unregularized Cox model.

and report the 95% bootstrap confidence interval as

$$\left[Q_{0.025}\left(\{\widetilde{r}_{m,q}^{(b)}\}_{b=1}^{B}\right), Q_{0.975}\left(\{\widetilde{r}_{m,q}^{(b)}\}_{b=1}^{B}\right)\right].$$

The overall rank is computed by pooling ranks over all evaluation metrics before applying the same median-rank and bootstrap procedure.

F Benchmark Dataset Details

We construct a large collection of real-world survival datasets from two sources. First, we use datasets from the `SurvSet` package [16]. We exclude datasets that are longitudinal, contain competing risks rather than a single event of interest. After applying these criteria, 67 `SurvSet` datasets remain. Second, we collect 24 additional survival datasets from textbooks, recent papers, and public software packages, restricting attention to datasets with at most 100,000 samples. This yields a total of 91 real-world datasets, which, to our knowledge, constitutes one of the largest survival-analysis benchmark collections considered in a single study. Summary statistics for all datasets are provided in Table 4.

Among the 91 datasets, we designate 20 datasets as validation datasets for selecting the SurvivalPFN checkpoint during training.[3] The remaining 71 datasets are held out for the final benchmark and are not used for checkpoint selection. For each validation dataset, we use a deterministic split with 70% of samples as the in-context training set and 30% as the inference set. At each training epoch, the current SurvivalPFN checkpoint is evaluated on all 20 validation datasets. We select the checkpoint with the best weighted average integrated Brier score,

$$\begin{aligned}\text{Weighted-IBS}(\theta) &= \frac{\sum_{\mathcal{D}} \sqrt{N}\ \text{IBS}_{\mathcal{D}}(\theta)}{\sum_{\mathcal{D}} \sqrt{N}} \\ \theta^* &= \arg\min_{\theta} \text{Weighted-IBS}(\theta),\end{aligned}$$

where $\text{IBS}_{\mathcal{D}}(\theta)$ is the IBS of checkpoint θ on the inference split of dataset $\mathcal{D}$, calculated using Equation E.2. The square-root weighting gives larger datasets more influence while avoiding domination by the largest cohorts.

The validation datasets were chosen to cover a broad range of empirical regimes. In particular, we considered sample size, censoring rate, and the tail survival probability estimated by the Kaplan-Meier curve, $\widehat{S}_{\text{KM}}(t_{\max})$.

Table 4: Summary of survival datasets used for checkpoint selection and held-out benchmarking. Features (cat. feat.) reports the total number of covariates, with the number of categorical covariates in parentheses.

Dataset	Number of samples	Features (cat. feat.)	Missing features	Missing rate	Censoring rate	$\widehat{S}(t_{\max})$
Validation datasets for checkpoint selection						
ovarian	26	4 (4)	0	0.0%	53.8%	49.7%
glioma	37	4 (4)	0	0.0%	37.8%	34.9%
leukemia	42	3 (2)	0	0.0%	28.6%	18.9%
pharmacoSmoking	125	16 (16)	0	0.0%	28.8%	28.8%
d.oropha.rec	192	24 (24)	0	0.0%	27.6%	16.5%
Pbc3	349	19 (19)	4	0.4%	82.5%	63.4%
retinopathy	394	11 (11)	0	0.0%	60.7%	53.1%
Rossi	432	7 (5)	0	0.0%	73.6%	73.6%
phpl04K8a	442	21 (21)	0	0.0%	46.6%	0.0%
prostate	502	25 (25)	4	0.2%	29.5%	23.8%

Continued on next page

[3] SurvivalPFN is not trained or fine-tuned on these validation datasets, nor on any other real-world survival dataset; they are used only for checkpoint selection.

Dataset	Number of samples	Features (cat.)	Missing feat.	Missing rate	Censoring rate	$\widehat{S}(t_{\max})$
uis	628	14 (14)	3	0.6%	19.1%	16.6%
grace	1,000	5 (5)	0	0.0%	67.6%	63.5%
rdata	1,040	6 (6)	0	0.0%	47.4%	38.1%
TRACE	1,878	6 (6)	0	0.0%	49.0%	43.9%
Aids2	2,839	12 (12)	0	0.0%	38.0%	5.8%
UnempDur	3,241	6 (6)	0	0.0%	38.7%	10.5%
smarto	3,873	34 (34)	16	4.8%	88.1%	72.5%
dataDIVAT1	5,943	16 (16)	0	0.0%	83.6%	0.0%
oldmort	6,495	14 (14)	0	0.0%	69.7%	0.0%
prostateSurvival	14,294	6 (6)	0	0.0%	94.4%	81.4%
Held-out test datasets for benchmarking						
vdv	78	4,705 (0)	0	0.0%	56.4%	56.4%
Bergamaschi	82	10 (10)	0	0.0%	65.9%	58.3%
larynx	90	4 (3)	0	0.0%	44.4%	29.7%
MCLcleaned	92	574 (0)	0	0.0%	30.4%	0.0%
lupus	95	5 (2)	1	0.2%	70.5%	36.8%
AML_Bull	115	6,283 (0)	0	0.0%	42.6%	28.5%
NSBCD	115	549 (0)	0	0.0%	67.0%	57.4%
gse4335	115	12,815 (20)	1	0.0%	67.0%	57.4%
micro.censure	117	81 (81)	0	0.0%	77.8%	42.4%
gse1992	124	15,541 (11)	1	0.0%	71.8%	45.4%
cgd	128	23 (23)	0	0.0%	65.6%	47.6%
veteran	137	8 (8)	0	0.0%	6.6%	0.0%
nki70	144	76 (76)	0	0.0%	66.7%	48.0%
stagec	146	18 (18)	2	0.4%	63.0%	55.8%
burn	154	13 (13)	0	0.0%	68.8%	48.0%
gse3143	158	8,663 (3)	0	0.0%	68.4%	41.9%
BMT	187	39 (24)	11	1.1%	54.5%	53.3%
WPBC	198	32 (1)	1	0.1%	76.3%	65.3%
Melanoma	205	5 (5)	0	0.0%	72.2%	64.5%
hepatoCellular	227	43 (43)	26	33.6%	57.3%	51.2%
cancer	228	28 (28)	4	1.0%	27.6%	5.0%
NCCTG	228	8 (1)	6	3.7%	27.6%	5.0%
DLBCL	240	7,339 (0)	0	0.0%	42.5%	26.8%
mgus	241	12 (12)	4	8.7%	6.6%	1.8%
e1684	284	3 (3)	0	0.0%	31.0%	28.3%
DBCD	295	4,919 (0)	0	0.0%	73.2%	61.0%
HFCR	299	11 (5)	0	0.0%	67.9%	57.6%
ova	358	11 (11)	0	0.0%	25.7%	22.2%
diabetes	394	4 (4)	0	0.0%	60.7%	53.0%
chop	412	3,833 (0)	0	0.0%	60.4%	39.7%
PBC	418	17 (7)	12	14.5%	61.5%	35.3%
zinc	431	20 (20)	0	0.0%	81.2%	79.0%
Unemployment	452	7 (7)	0	0.0%	43.4%	5.8%
whas500	461	17 (17)	0	0.0%	61.8%	0.0%
cost	518	13 (13)	0	0.0%	22.0%	22.0%
BCCardiotox	531	25 (15)	22	5.7%	89.8%	69.0%
GBM	591	10 (7)	3	4.5%	17.1%	0.0%
vlbw	617	41 (41)	5	1.7%	82.7%	0.0%
GBSG2	686	8 (2)	0	0.0%	56.4%	34.3%
FRTCS	697	13 (13)	2	0.1%	89.7%	57.2%
kidney_transplant	863	4 (3)	0	0.0%	83.8%	72.4%
dataOvarian1	912	162 (162)	0	0.0%	40.4%	11.9%
colon	929	12 (12)	1	0.2%	51.3%	45.5%
credit	1,000	31 (20)	1	0.6%	30.0%	5.9%

Continued on next page

Dataset	Number of samples	Features (cat.)	Missing feat.	Missing rate	Censoring rate	$\widehat{S}(t_{\max})$
PDM	1,000	8 (5)	0	0.0%	60.3%	0.0%
LeukSurv	1,043	29 (29)	0	0.0%	15.7%	4.4%
actg	1,151	17 (17)	0	0.0%	91.7%	90.1%
COVID	1,422	9 (2)	0	0.0%	90.5%	1.1%
WHAS	1,638	6 (4)	0	0.0%	57.9%	35.7%
dataDIVAT2	1,837	4 (4)	0	0.0%	68.3%	31.1%
scania	1,931	8 (8)	0	0.0%	43.8%	15.9%
churn	1,958	19 (10)	0	0.0%	52.4%	24.4%
METABRIC	1,981	79 (73)	0	0.0%	55.2%	11.6%
AIDS	2,139	22 (14)	0	0.0%	24.4%	0.0%
NACD	2,396	48 (33)	0	0.0%	36.4%	12.5%
rott2	2,982	12 (12)	0	0.0%	57.3%	26.5%
divorce	3,371	4 (4)	0	0.0%	69.4%	55.7%
acath	3,504	3 (3)	1	11.9%	33.4%	0.0%
NWTCO	4,028	6 (5)	0	0.0%	85.8%	84.9%
dataDIVAT3	4,267	16 (16)	0	0.0%	94.4%	84.7%
Framingham	4,699	17 (17)	2	0.1%	68.7%	60.5%
NPC	6,449	9 (3)	0	0.0%	80.8%	75.6%
Dialysis	6,805	72 (72)	0	0.0%	76.4%	58.7%
FLCHAIN	7,871	23 (2)	1	0.7%	72.5%	68.2%
MSKCC	8,130	206 (199)	2	0.0%	70.3%	34.8%
SUPPORT	9,105	31 (11)	10	4.0%	31.9%	24.1%
employee	11,991	16 (9)	0	0.0%	83.4%	50.8%
MIMIC-IV_all	38,520	91 (6)	0	0.0%	66.7%	0.0%
hdfail	52,422	88 (88)	0	0.0%	94.5%	56.1%
SEER_brain	73,703	10 (4)	0	0.0%	40.1%	26.6%
SEER_liver	82,841	14 (1)	0	0.0%	37.6%	18.0%

We briefly describe below the 24 datasets that are not taken from `SurvSet`. For the `SurvSet` datasets, please refer to Drysdale [16].

- *leukemia*: The *leukemia* dataset records remission survival for 42 patients with sex, treatment assignment, and log white blood cell count [51].
- *larynx*: The *larynx* cancer dataset follows 90 male patients from first treatment until death or study end, with disease stage, age, and diagnosis year [45].
- *lupus*: The *lupus* dataset records survival times and diagnostic features for systemic lupus erythematosus patients [64].
- *BMT*: The Bone Marrow Transplant (*BMT*) Children dataset describes pediatric patients with malignant and nonmalignant hematologic diseases who underwent unrelated-donor allogeneic hematopoietic stem cell transplantation [94].
- *NCCTG*: The *NCCTG* lung cancer dataset follows advanced lung cancer patients with physician-rated and patient-rated performance scores [59].
- *HFCR*: The Heart Failure Clinical Records (*HFCR*) dataset contains follow-up outcomes and clinical measurements for 299 heart failure patients [8].
- *Rossi*: The *Rossi* dataset follows 432 Maryland prison releasees for one year to study recidivism after financial-aid treatment assignment [91].
- *BCCardiotox*: *BCCardiotox* follows HER2+ breast cancer patients treated with potentially cardiotoxic therapies and records time to cancer therapy-related cardiac dysfunction [77].
- *GBM*: The TCGA glioblastoma multiforme (*GBM*) dataset contains clinical survival information for primary brain tumor patients [96].
- *kidney_transplant*: The *kidney_transplant* dataset records post-transplant death times with recipient age, gender, and race [50].

- *credit*: The `PySurvival` [20] credit-risk dataset adapts the German Credit data to model the time until a loan is fully repaid.
- *PDM*: The `PySurvival` [20] predictive-maintenance (*PDM*) dataset models the time until an industrial machine breaks.
- *COVID*: The COVID-19 Asian discharge dataset models time to hospital discharge for patients with COVID-19 diagnosis [53].
- *WHAS*: The Worcester Heart Attack Study (*WHAS*) follows acute myocardial infarction patients after hospital admission [40].
- *churn*: The `PySurvival` [20] churn dataset models when SaaS customers stop their monthly subscription.
- *METABRIC*: *METABRIC* profiles primary breast tumors with long-term clinical follow-up for breast-cancer prognosis [13].
- *AIDS*: ACTG 175 records HIV-infected adults randomized to nucleoside monotherapy or combination therapy [31].
- *NACD*: The Northern Alberta Cancer Dataset contains clinical records for several cancer sites, which used to predict the death from cancer onset [104].
- *NPC*: The nasopharyngeal carcinoma dataset follows patients from Sun Yat-sen University Cancer Center for progression-free survival after radiotherapy with or without chemotherapy [97]. Original training and validation cohorts are combined.
- *MSKCC*: The MSK-IMPACT cohort contains targeted sequencing and clinical annotations from more than 10,000 advanced cancer cases [105].
- *MIMIC-IV_all*: *MIMIC-IV_all* is derived from the MIMIC-IV critical care database, which contains de-identified electronic health records for patients admitted to intensive care units [43]. We use the all-cause mortality cohort curated by Qi et al. [81], restricting to patients who survived at least 24 hours after ICU admission.
- *employee*: The `PySurvival` [20] employee-retention dataset models when employees leave a company using HR and workload attributes.
- *SEER_brain*: *SEER_brain* is a brain cancer cohort derived from the Surveillance, Epidemiology, and End Results (SEER) Program registry. The task is to predict time from cancer diagnosis to death or censoring, using the cohort curated by Qi et al. [84].
- *SEER_liver*: *SEER_liver* is the corresponding liver cancer cohort from SEER, with the same time-to-death prediction target and curation protocol [84].

G Additional Experimental Results

G.1 RQ1: Predictive Performance

Figure 6 has demonstrate the overall performance across 71 benchmark datasets. The intervals reflect uncertainty in the estimated median rank across the benchmark collection, rather than the variability of ranks themselves. In this appendix, we further analyze the benchmark results by stratifying datasets according to sample size and censoring rate.

The sample-size stratification reveals that SurvivalPFN is particularly effective in the small-data regime. On small datasets (Figure 12), SurvivalPFN is clearly separated from most baselines in overall rank and performs strongly across all five metrics, including IBS, CI, D-calibration, MAE, and Log-Rank. Traditional statistical survival methods and tree-based methods performance in the second tier, while neural-network-based methods generally performance the worst.

For medium-sized datasets (Figure 13), SurvivalPFN remains among the strongest methods overall, with especially competitive performance on IBS, Log-Rank. However, the gap to the best baselines becomes smaller, and several nerual-network-based methods (*e.g.*, MTLR, SurvivalMDN) become competitive on the overall performance and also on individual metrics such as CI or D-calibration.

On large datasets (Figure 14), the advantage of SurvivalPFN decreases further: its overall, IBS, and CI ranks move leftward compared with the small-data setting, although it remains competitive on MAE and Log-Rank. This trend suggests a size-regime trade-off. As more observations become

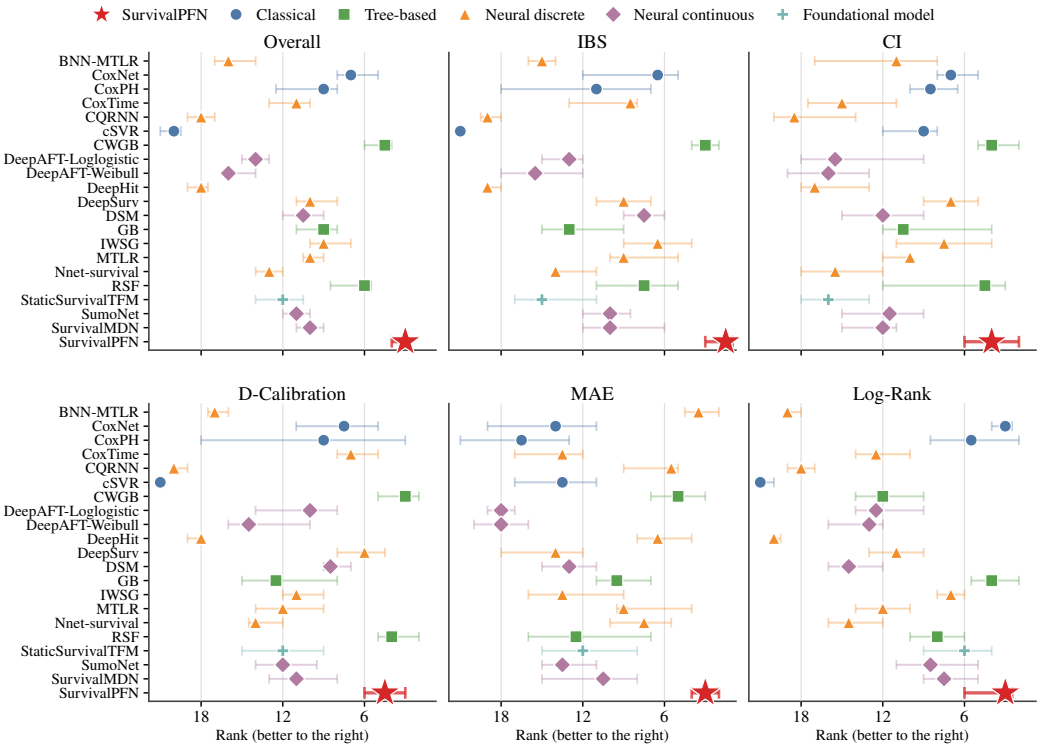


Figure 12: **Model ranks across 24 small-size and non-high-dimensional datasets ($d < N < 500$).** Points/stars denote median ranks across datasets, with horizontal bars showing 95% bootstrap confidence intervals for the median rank.

available, deep-learning-based estimators can benefit more directly from the larger sample size, whereas SurvivalPFN is constrained by finite context length and the need to summarize large tables during inference.

In contrast, the censoring-rate stratification shows substantially more stable behavior. Across low-, medium-, and high-censoring subsets (Figures 15-17), SurvivalPFN remains in the top group overall and retains strong performance on IBS, MAE, and Log-Rank. This consistency is encouraging because censoring affects the available event-time information and can differentially impact discrimination, calibration, and distributional accuracy metrics. The results suggest that the synthetic prior used to train SurvivalPFN provides useful robustness across censoring regimes. Although the strongest baseline varies across metrics and censoring levels, no single baseline matches SurvivalPFN's overall consistency across dataset strata, censoring strata, and evaluation metrics.

The current SurvivalPFN architecture uses row tokenization with a fixed 100 input width: one input position is reserved for the query indicator, leaving support for at most 99 covariates (to be more precise, $k_D \leq \min(99, n_{\text{context}}, d)$). For datasets with more than 99 features, we therefore fit TruncatedSVD using only the training/context covariates, retain 99 components, and apply the same transformation to the held-out query covariates before in-context inference. As this unsupervised compression step may discard weak prognostic signals, we do not anticipate SurvivalPFN to excel in high-dimensional datasets.

To test the boarder of the model, we evaluate the SurvivalPFN performance over the high-dimentional datasets. To our surprise, even though SurvivalPFN is not trained with high-dimensional priors nor the validation set (for selecting the best checkpoint) contain high-dimensional dataset, SurvivalPFN achieves the third-best overall median rank and remains particularly competitive for IBS and D-calibration. These results suggest that SurvivalPFN can retain useful performance after dimensionality reduction, but they do not establish native scalability to arbitrarily high-dimensional inputs. The fixed 99-feature representation remains an architectural limitation. The revised version will explicitly acknowledge this and identify scaling-up the model as important future directions.

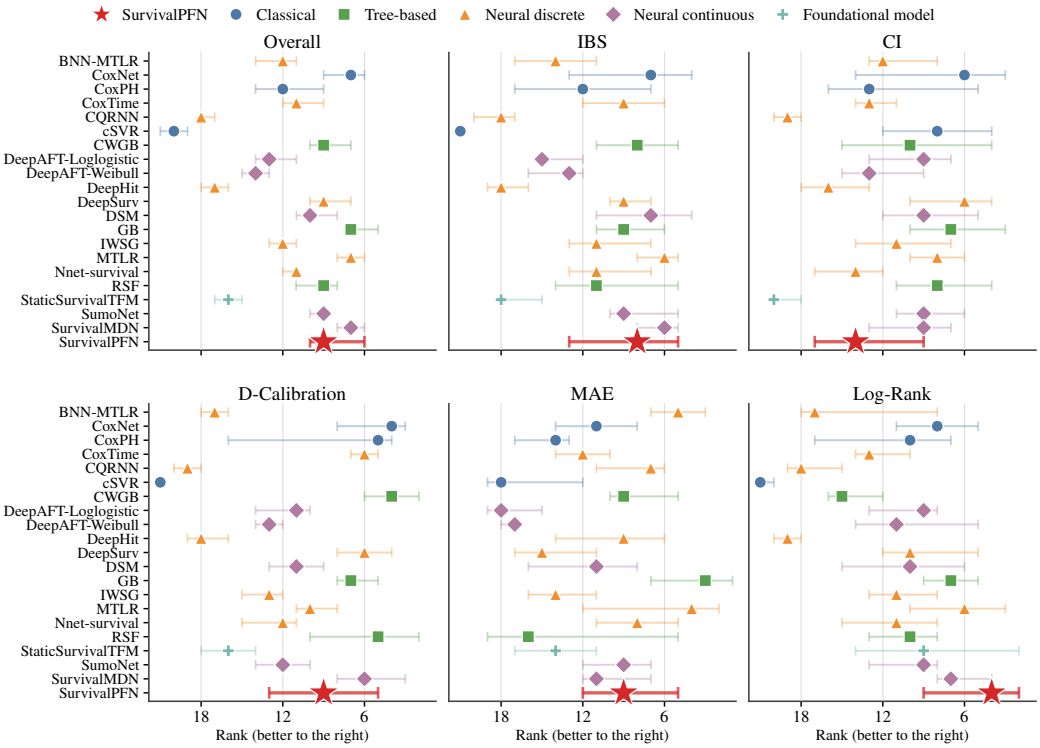


Figure 13: **Model ranks across 27 medium-size datasets** ($500 \leq N < 5000$)**.** Points/stars denote median ranks across datasets, with horizontal bars showing 95% bootstrap confidence intervals for the median rank.

G.2 RQ2: Computational Efficiency

For the runtime comparison in Figure 1, datasets with at least one failed run are excluded from this runtime aggregation, so every method is compared on the same set of completed datasets. This prevents methods from appearing artificially faster because they crashed, timed out, or failed to produce valid predictions on more demanding datasets. Predictive-performance rankings still follow the failure handling protocol in Appendix E.6.

G.3 RQ3: Sensitivity to Training-Set Size

Figure 19 extends the training-ratio analysis to 12 additional datasets. Across these datasets, increasing the training ratio improves all methods, with IBS decreasing and CI increasing most sharply when moving from very small context sizes to moderate context sizes. SurvivalPFN remains stable across ratios and is frequently among the best methods on both metrics, especially in low-data regimes.

G.4 RQ4: Compare with General Tabular Foundational Models

This section compares the SurvivalPFN, StaticSurvialTFM with other general TFMs. Specifically, we includes the most advanced TFMs including: TabPFN v2.5 [28], TabICL v2 [85], MITRA [107], and TabDPT [62] regressors.

For the these TFM regression baselines, we train only on uncensored training examples because these models cannot natively deal with right-censoring datasets. TabPFN and TabICL are used as quantile regressors: they predict event-time quantiles, which are monotonized and converted into survival curves for distributional evaluation. MITRA and TabDPT are used as point regressors: they predict a single event time per test subject (just like cSVR). Since point predictions do not define a full survival distribution, distributional metrics such as IBS and D-calibration are not calculated and ranked as the

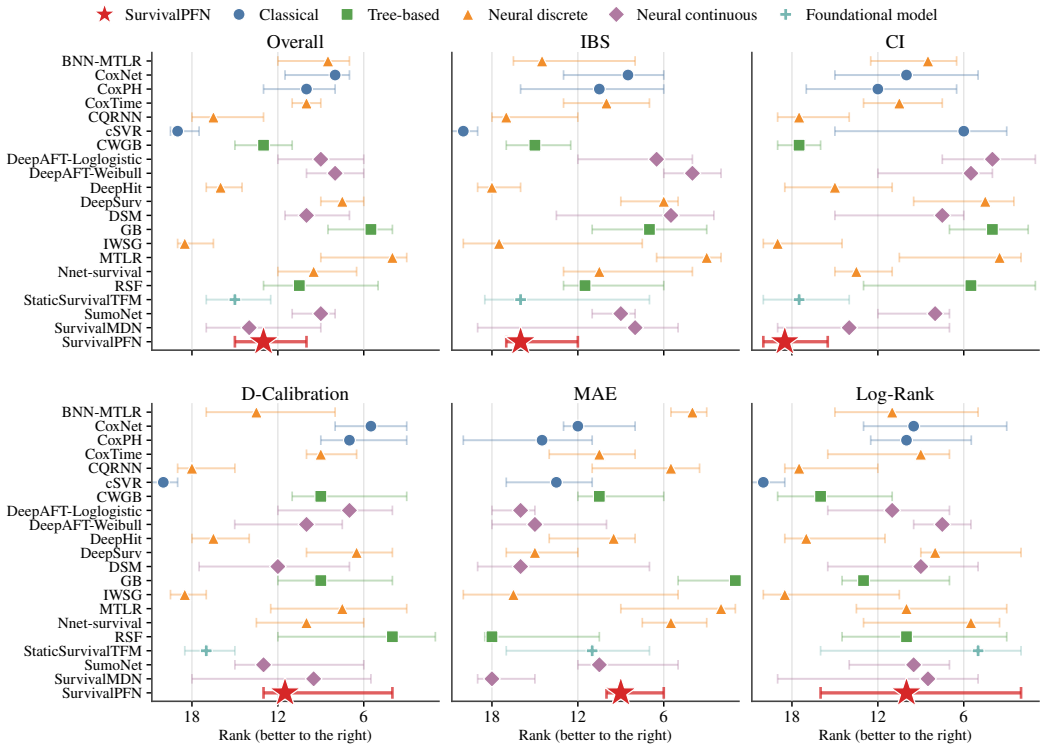


Figure 14: **Model ranks across 10 large-size datasets ($N \geq 5000$).** Points/stars denote median ranks across datasets, with horizontal bars showing 95% bootstrap confidence intervals for the median rank.

worst among all the methods, while CI, MAE, and log-rank style comparisons are computed from the predicted event times.

For StaticSurvialTFM [47], it is a static fomula that can convert any classifier to survival predictor. We instantiate this static formulation with TabDPT and MITRA classifier backbones, predict failure probabilities over the cutoff grid, convert them to survival probabilities, and enforce monotone survival curves. We choose TabDPT to match with the model architecture of SurvivalPFN (for a fair comparison). We include MITRA as it is the best performing backbone described in Kim et al. [47].

The results present here uses the same evaluation protocal as described in Appendix G.1. Figure 20 expands the comparison in Figure 8 by including both general TFMs and the StaticSurvivalTFM wrapper instantiated with TabDPT and MITRA. Overall, SurvivalPFN remains the strongest and most consistent method: it achieves the best aggregate rank and ranks first or near-first across nearly all metrics. The largest gains appear for IBS and D-calibration, where SurvivalPFN clearly outperforms both direct TFM baselines and StaticSurvivalTFM variants, indicating better probabilistic survival estimation and calibration. SurvivalPFN also performs best on CI and log-rank, showing that its advantage extends beyond distributional accuracy to risk ranking and group separation.

The StaticSurvivalTFM performance is really sensitive to the backbone TFM model – which aligns the findings in [47]. Using MITRA as the backbone improve over using TabDPT, especially for CI and log-rank, confirming that survival-specific label construction is helpful. However, their performance is less stable across metrics: StaticSurvivalTFM (MITRA) is competitive on MAE, CI, but does not match SurvivalPFN on IBS, D-calibration and Log-rank; StaticSurvivalTFM (TabDPT) performs well on log-rank but is weaker on other. In contrast, the direct regression baselines – TabPFN, TabICL, MITRA, and TabDPT – are consistently worse, despite being strong general tabular predictors.

These results support the main conclusion of **RQ4**: survival prediction benefits from a foundation model trained with survival-specific supervision and censoring-aware synthetic tasks, rather than relying only on generic tabular in-context learning.

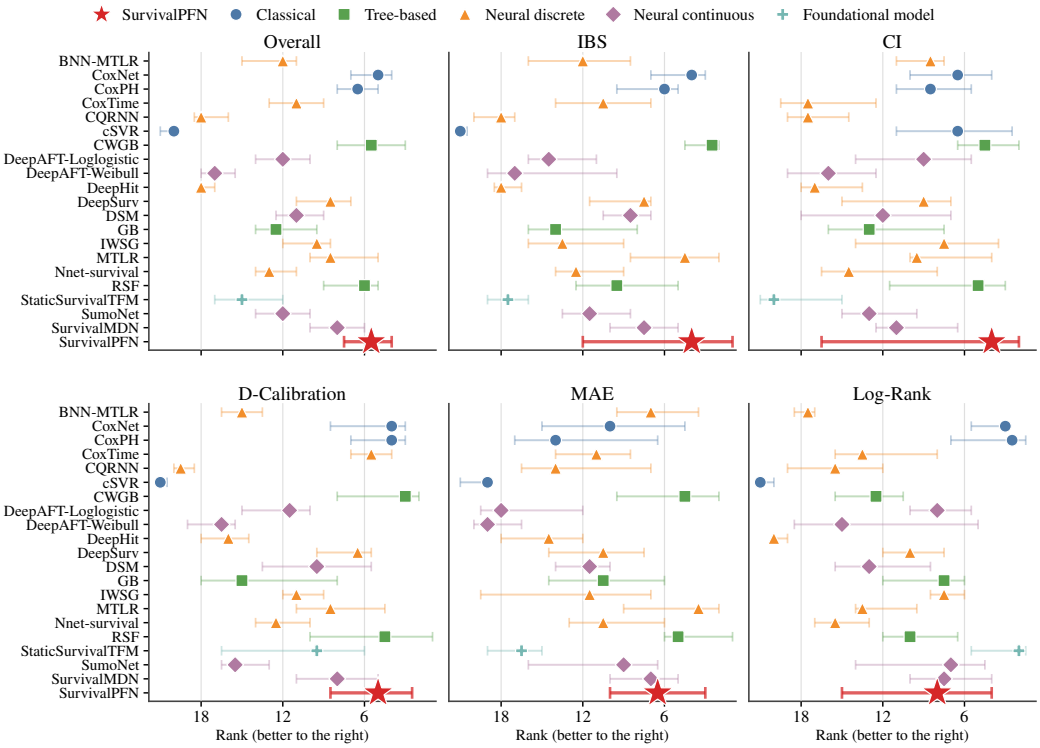


Figure 15: **Model ranks across 13 low-censoring-rate datasets (censoring rate $<$ 33%).** Points/stars denote median ranks across datasets, with horizontal bars showing 95% bootstrap confidence intervals for the median rank.

G.5 RQ5: Evidence of Using Context

Figure 21 presents the individual dataset changes, ordered by decreasing CI loss and aligned across the IBS and CI panels. This view reveals differences between the effects on discrimination and overall survival prediction error. For example, FRTCS exhibits the largest CI decrease, approximately 0.546, whereas churn shows the largest relative IBS increase, approximately 73.0%. Conversely, hdfail shows an IBS improvement despite a decrease in CI. The aligned panels therefore expose metric-specific responses that aggregate summaries can obscure. Together, the dataset-level patterns support the interpretation that SurvivalPFN's predictions depend on the covariate–outcome association supplied by its context.

G.6 RQ6: Ablation Studies

SurvivalPFN Ablation Configurations. Table 5 summarizes the SurvivalPFN variants used in the ablation study. Each row corresponds to one pretrained checkpoint and is defined by six configuration choices.

Predictive Pretraining specifies whether the model is initialized from the predictive PFN-style pretraining phase before survival-specific training. ✓ means that the model first undergoes the general predictive pretraining stage described in Appendix D.2, and is then further trained with the survival phase. A value of ✗ means that survival-phase training starts without this predictive initialization. This ablation tests whether generic PFN-style in-context predictive pretraining improves downstream survival prediction.

Prior specifies the synthetic survival prior used to generate the right-censored pretraining tasks, as described in Appendix D.1. We consider four possible prior families – the *naive prior*, the *survival-distribution prior*, the *mixture prior* and the *kitchen-sink prior*.

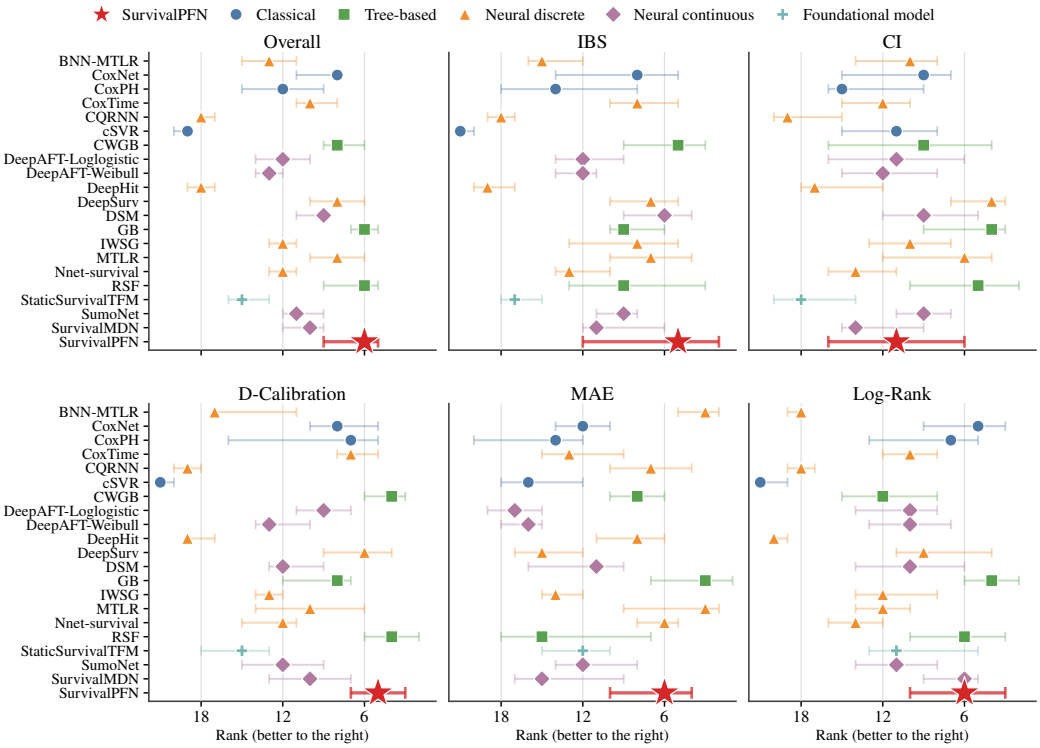


Figure 16: **Model ranks across 29 medium-censoring-rate datasets (censoring rate $\geq$ 33% and $<$ 67%).** Points/stars denote median ranks across datasets, with horizontal bars showing 95% bootstrap confidence intervals for the median rank.

Time transformation specifies the monotone transformation applied to event and censoring times before discretization, as described in Appendix D.3. We try the `lognormal2normal` and the `time2quantile` transformations.

Loss specifies how the discretized predictive distribution is trained in transformed-time space, as described in Appendix D.4. The *NLL* setting uses a one-hot discrete negative log-likelihood, assigning all target mass to the bin containing the latent target time. The *CE* setting uses the smoothed histogram cross-entropy loss, where the latent target time is converted into a narrow Gaussian-smoothed histogram over bins.

Query schedule specifies how the query indicator $\widetilde{\delta}^*$ is selected during training, following Section 3. We try the *event-only*, *both*, and *random* strategies.

Variable train ratio specifies whether the ratio between context/training samples and query/inference samples is varied during synthetic pretraining. A value of ✓ means that this ratio is randomized across synthetic tasks, exposing the model to different amounts of context information and encouraging robustness to varying downstream train/test splits. A value of ✗ means that the ratio is fixed at 70%/30% during training.

Together, these choices define a full configuration space of $2 \times 4 \times 2 \times 2 \times 3 \times 2 = 192$ possible variants (corresponding respectively to predictive pretraining, prior family, time transformation, loss, query schedule, and variable train ratio). Exhaustively training all variants would be computationally expensive, so we selectively evaluate the representative configurations in Table 5. The model marked with † is the best validation run and is used as the default SurvivalPFN checkpoint elsewhere in the paper.

Ablation results. Figure 22 summarizes the rank of each SurvivalPFN configuration. The selected checkpoint, v01, achieves the strongest overall behavior: it attains the best median rank across metrics among the evaluated configurations, and is particularly strong on distributional metrics,

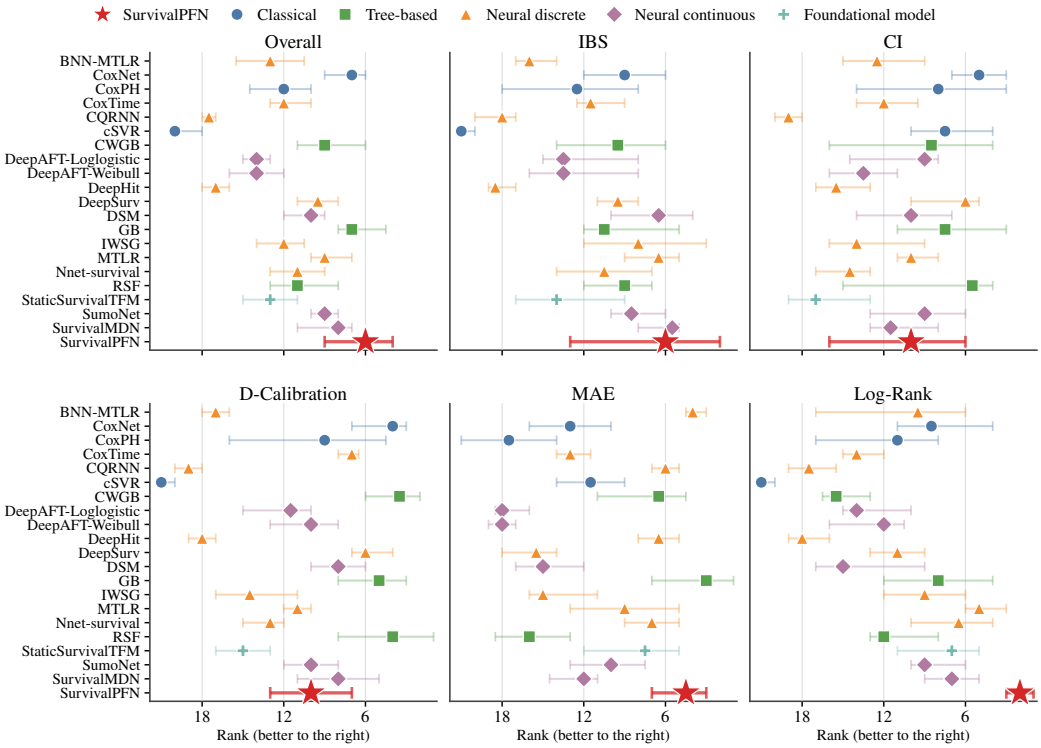


Figure 17: **Model ranks across 29 high-censoring-rate datasets (censoring rate $\geq$ 67%).** Points/stars denote median ranks across datasets, with horizontal bars showing 95% bootstrap confidence intervals for the median rank.

ranking best on IBS and D-calibration. This configuration uses predictive pretraining, the survival-distribution prior, the `lognormal2normal` transformation, the one-hot NLL objective, the Both query schedule, and a fixed train/query ratio. Its strong IBS and D-calibration performance suggests that this combination is especially effective for learning calibrated posterior predictive survival distributions, which is the primary target of SurvivalPFN.

Several trends emerge from the ablation. First, the survival-distribution prior is consistently stronger than the naive prior under otherwise similar settings. This suggests that directly modeling flexible positive-time distributions provides a more useful synthetic pretraining signal than treating generic tabular outputs as raw survival times. The kitchen-sink prior performs competitively but does not clearly dominate the survival-distribution prior, this might indicating that simply increasing prior diversity is not sufficient; the match between the prior family and the survival-prediction target also matters.

Second, the `lognormal2normal` transformation is preferred in this set of experiments. The clearest comparison is between v01 and v02, replacing `lognormal2normal` with `time2quantile` substantially worsens the overall rank and degrades all five metric-specific ranks. This pattern suggests that the smooth positive-time coordinate and tail extrapolation provided by `lognormal2normal` are useful for transferring across heterogeneous real-world time scales, whereas the empirical quantile transform may lose information about tail behavior.

Finally, predictive pretraining is generally helpful but not uniformly decisive in this limited ablation set. Similarly, varying the train/query ratio does not show a clear monotonic benefit in the evaluated subset.

H Concurrent Works on Tabular Foundation Models for Survival Analysis

We discuss two concurrent approaches that adapt TFMs to right-censored survival prediction.

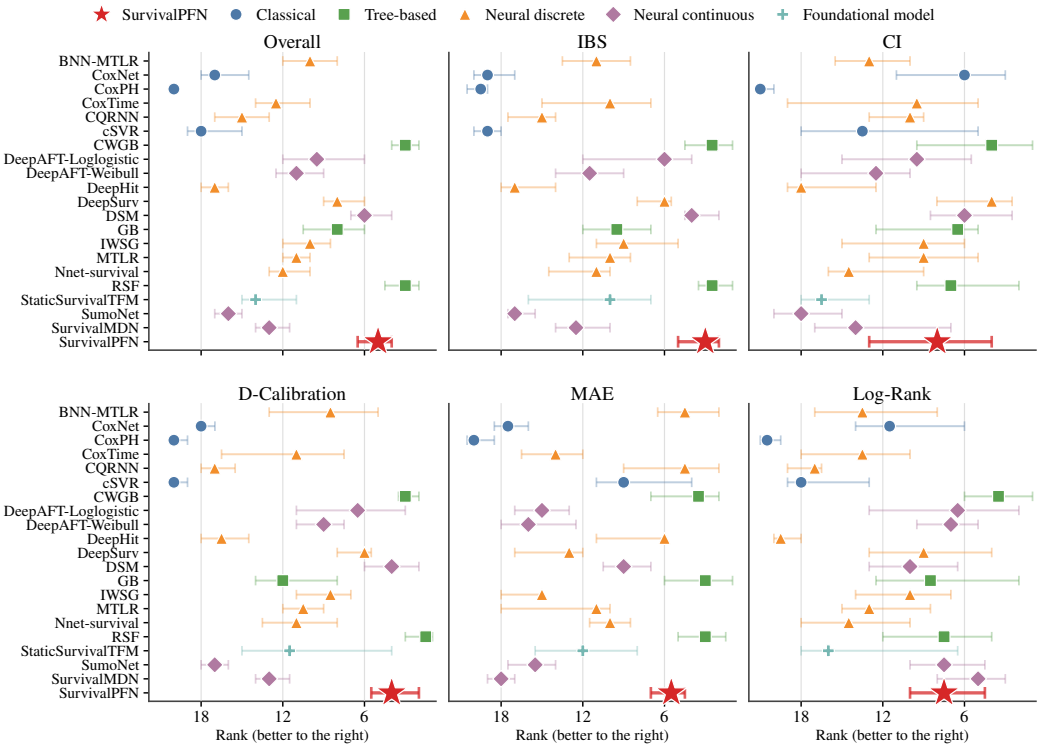


Figure 18: **Model ranks across 10 small-size and high-dimensional datasets ($N < 500, N < d$).** Points/stars denote median ranks across datasets, with horizontal bars showing 95% bootstrap confidence intervals for the median rank.

Classification-Based Framework with Off-the-Shelf TFMs. Kim et al. [47] propose a conversion from survival analysis to binary classification, allowing existing TFMs to be used without survival-specific pretraining. Given predefined discretization points

$$0 = t_0 < t_1 < \cdots < t_{K-1},$$

they define time-indexed binary labels

$$Y_{i,k} = \mathbb{1}(T_i \leq t_k),$$

so that each original tuple (x_i, t_i, δ_i) is expanded into multiple classification examples indexed by k. Under right censoring, labels after the censoring time are treated as missing; equivalently, their binary cross-entropy objective is evaluated only when $t_k < C_i$. Under conditional independent censoring and positivity, they show that minimizing the population binary cross-entropy loss recovers the true failure probabilities,

$$p(x, t_k) = \Pr(T \leq t_k \mid X = x),$$

and hence the survival probabilities $S(t_k \mid x) = 1 - p(x, t_k)$. This formulation is attractive because it can immediately use strong off-the-shelf TFMs such as MITRA, TabPFN v2.5, and TabICL v2 [107, 28, 85], without retraining a survival-specific model.

The main limitation is that this reduction increases the effective context size. Each subject produces up to $K - 1$ time-indexed classification examples, so a dataset with N subjects becomes an expanded context of order $N(K - 1)$. This is manageable for small datasets, but can exceed the input limits of current TFMs on medium or large survival datasets, requiring subsampling and potentially discarding observed survival information. The method also inherits limitations of discrete-time classification, including dependence on the time grid and the need for post-hoc monotonicity correction of predicted survival curves. In our experiments, we include the static version of this approach as StaticSurvivalTFM.

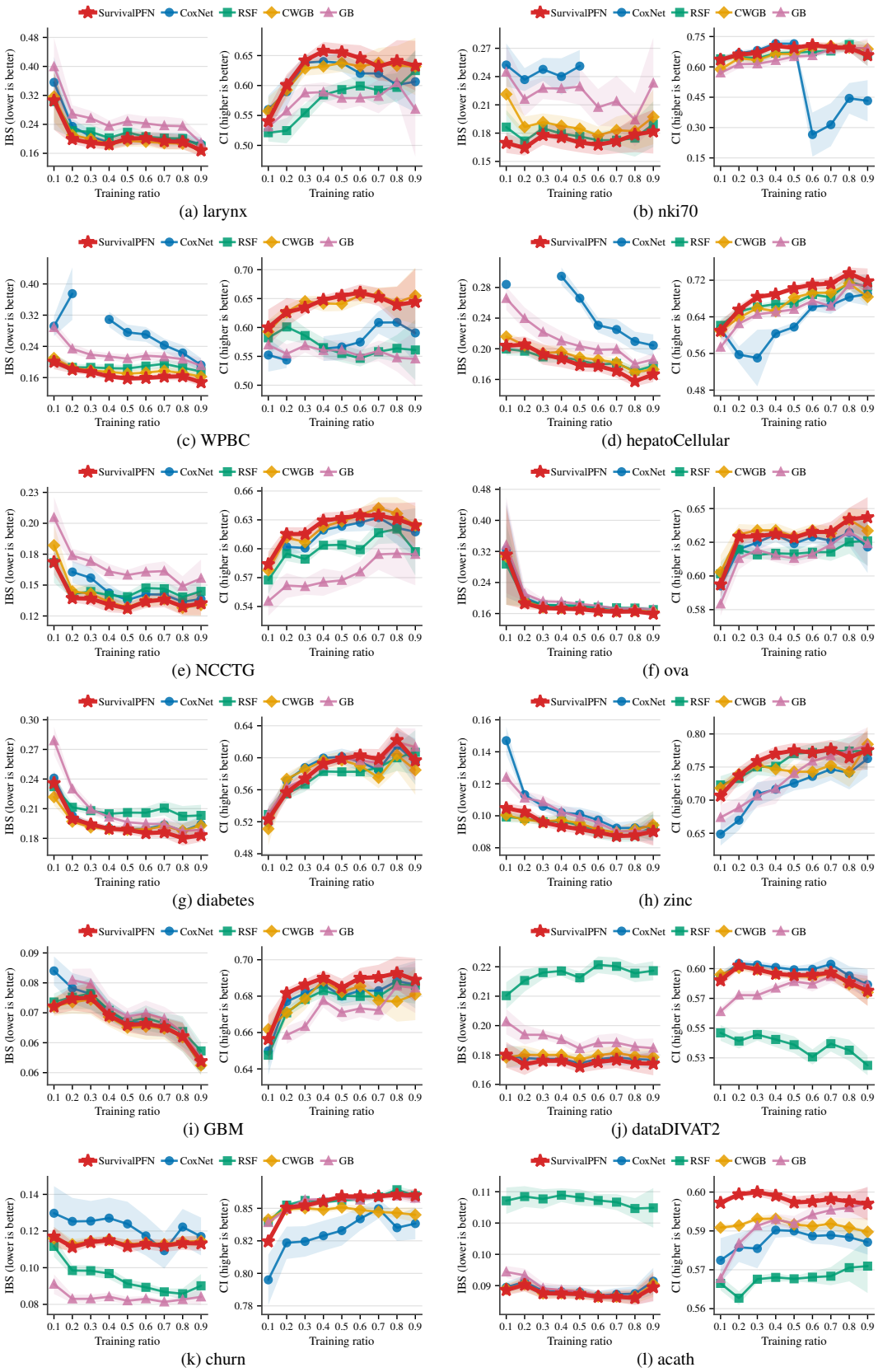


Figure 19: Sensitivity to the training/context ratio across selected 12 datasets.

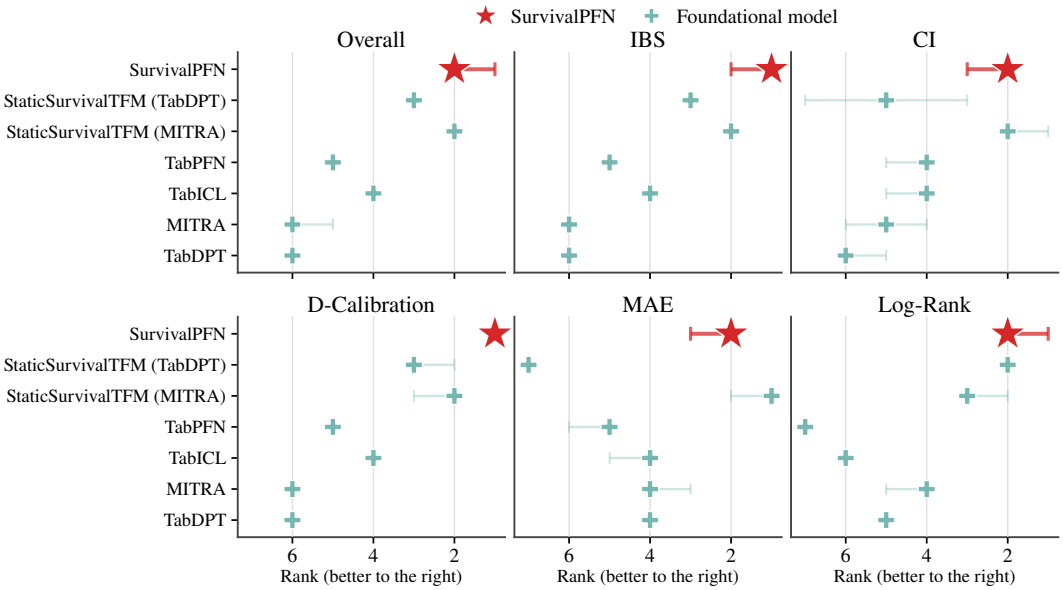


Figure 20: **Compare SurvivalPFN with general TFMs across 61 benchmark datasets, excluding high-dimensional datasets.** Plotting conventions follow Figure 6.

Table 5: **SurvivalPFN ablation configurations.** Each row corresponds to one pretrained SurvivalPFN variant. The internal checkpoint-path column from the experiment log is omitted. "Surv.-dist." denotes the survival-distribution prior. NLL denotes the one-hot discrete negative log-likelihood over transformed-time bins; CE denotes the smoothed histogram cross-entropy loss.

Model	Predictive Pretrain	Prior	Time Transformation	Loss	Query Schedule	Variable Train Ratio
v01†	✓	Surv.-dist.	`lognormal2normal`	NLL	Both	✗
v02	✓	Surv.-dist.	`time2quantile`	NLL	Both	✗
v03	✗	Surv.-dist.	`time2quantile`	CE	Random	✓
v04	✓	Surv.-dist.	`time2quantile`	CE	Random	✓
v05	✓	Surv.-dist.	`time2quantile`	NLL	Random	✓
v06	✓	Surv.-dist.	`lognormal2normal`	NLL	Random	✓
v07	✓	Kitchen-sink	`lognormal2normal`	CE	Random	✓
v08	✓	Kitchen-sink	`lognormal2normal`	NLL	Event-only	✓
v09	✓	Surv.-dist.	`lognormal2normal`	CE	Random	✓
v10	✓	Surv.-dist.	`lognormal2normal`	NLL	Event-only	✓
v11	✓	Naive	`lognormal2normal`	CE	Random	✓
v12	✓	Naive	`lognormal2normal`	NLL	Event-only	✓
v13	✗	Naive	`lognormal2normal`	CE	Random	✓
v14	✗	Naive	`lognormal2normal`	NLL	Event-only	✓

†Best validation run; this checkpoint is used as the default SurvivalPFN model elsewhere in the paper.

Survival-Specific Prior-Fitted In-Context Learning. Seletkov et al. [92] propose Survival In-Context (SIC), a survival-specific prior-fitted model trained on synthetic right-censored datasets. Their data generator first samples covariates and latent risk variables (η_1, η_2) from structural causal models (SCMs). Event times are then generated using the extended-hazard model

$$h(t \mid x) \;=\; h_0(te^{\eta_1})e^{\eta_2},$$

which yields

$$T \;=\; e^{-\eta_1} H_0^{-1}\big(e^{\eta_1-\eta_2}(-\log U)\big)\,, \qquad U \;\sim\; \mathrm{Unif}(0,1),$$

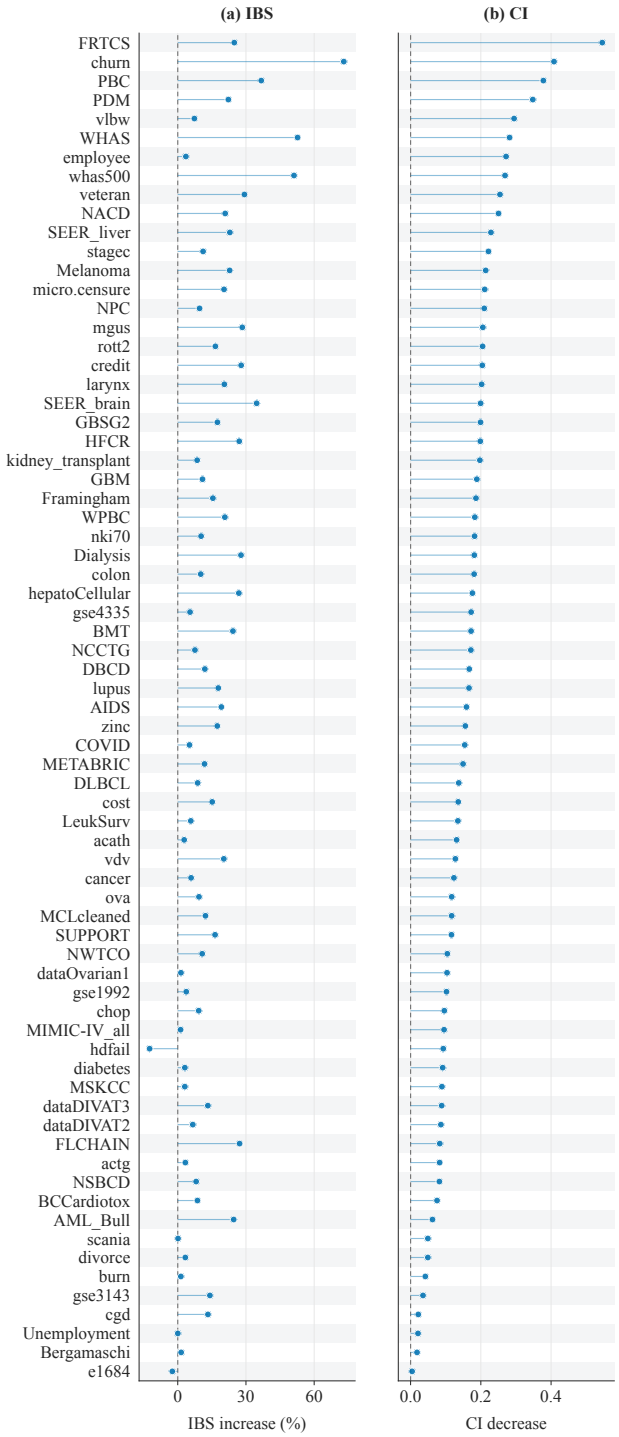


Figure 21: **Dataset-level sensitivity to context outcome permutation.** Relative IBS increases (a) and absolute CI decreases (b) are shown for all 71 datasets, ordered by decreasing CI loss. Each point is computed from the standard and permuted condition means over ten paired repetitions. Positive values indicate deterioration; dashed vertical lines indicate no change.

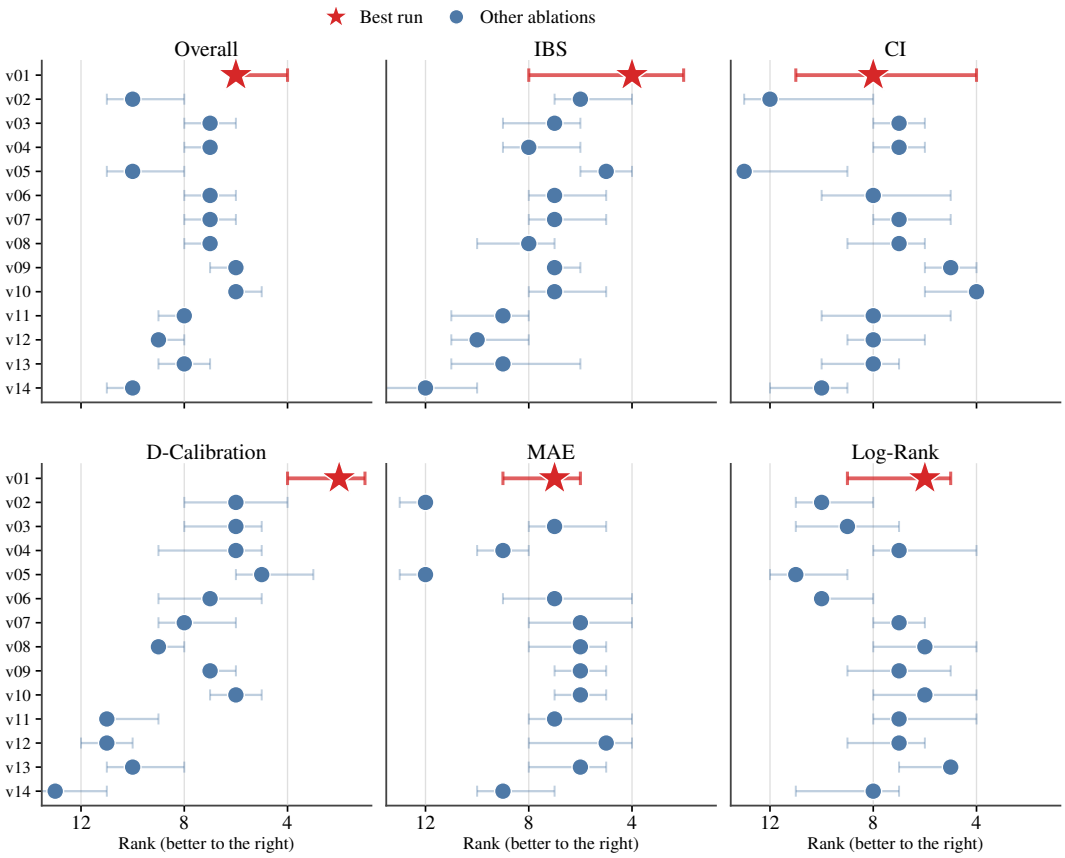


Figure 22: Ablation study over SurvivalPFN training configurations. Each row corresponds to one pretrained SurvivalPFN variant, with v01 marked as the selected best validation run and used as the default checkpoint elsewhere in the paper. Plotting conventions follow Figure 6.

where H_0^{-1} is chosen from a set of parametric baseline families such as Weibull, lognormal, log-logistic, Gompertz, and Birnbaum-Saunders distributions. Censoring is generated by random censoring assumption – not dependent on covariates and event times.

Architecturally, SIC builds on TabICL, adds a time-event embedding, and uses a DeepHit-style discrete-time survival head Lee et al. [55] trained with a likelihood-plus-ranking loss.

SIC is closely related to SurvivalPFN in that both methods pretrain an in-context model specifically for survival prediction. However, the two approaches differ substantially in prior design, which is central to the PFN paradigm. SIC's prior is based on a parametric extended-hazard construction and only random censoring, whereas SurvivalPFN uses a broader family of identifiable right-censored DGPs, including random censoring and covariate-dependent censoring mechanisms satisfying conditional independence. SurvivalPFN also avoids committing to explicit parametric hazard or survival families, allowing the event and censoring distributions to be generated by more flexible stochastic neural mechanisms. Finally, SurvivalPFN is accompanied by a posterior-predictive consistency guarantee for identifiable survival priors, whereas SIC only provides empirical evidence.

The empirical scope also differs: SIC evaluates on a smaller benchmark with one main metric and a limited set of baselines, while our study evaluates on 71 held-out datasets, five metrics, and 20 baselines. Since SIC has not released public model weights or code, we cannot include it in our direct empirical comparison.

NeurIPS Paper Checklist

1. **Claims**

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [Yes]

Justification: The abstract and introduction accurately scope the paper as amortized Bayesian survival prediction for identifiable right-censored data, and the contribution list matches the method, theory, benchmark described in Sections 3, Figure 6, and Appendix E.6.

Guidelines:

- The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes]

Justification: Section 6 explicitly discusses the main limitations, including the reliance on conditionally independent censoring for identifiability and the reduced scalability of PFN-style models on larger tables. These limitations are further supported by the identifiability discussion in Appendix B, the stratified benchmark results in Appendix G.1, and the compute/failure protocol in Appendices E.5–E.6.

Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The main consistency claim is stated informally in Proposition 3.1 and formalized in Appendix C, with the required assumptions in Assumption C.1 and Definition C.1. The appendix provides complete proofs for Proposition C.1 and Corollary C.1, and Appendix B gives the supporting identifiability and non-identifiability arguments for right-censoring mechanisms.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The paper provides the model architecture and training procedure in Appendix D, the benchmarked methods and hyperparameter tuning protocol in Appendices E.1-E.3, and the evaluation metrics and reporting protocol in Appendices E.4-E.6. The real-world datasets, validation/test split design, compute setting, and released code/model weights are described in Appendix F, Appendix E.5, and the released code.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We upload the SurvivalPFN training code, best model weights, and a `scikit-learn`-style API in the released code, and the datasets used for evaluation are publicly available and summarized in Appendix F. The reproduction-relevant details are documented through the model, hyperparameter, metric, compute, and benchmark protocol descriptions in Appendices E.1-E.6.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The training details for SurvivalPFN are specified in Appendices D.1-D.4 and Appendix D.2, including the prior generation process, architecture, loss, optimizer, batching, and inference procedure. The experimental protocol further specifies the dataset split strategy, repeated runs, metric computation, hyperparameter tuning, compute limits, and held-out evaluation in Appendices E.3-E.6.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Figures 7 and 19 reports mean $\pm$ standard deviation over 10 train/test splits for each dataset, and Figures 6, 8, 12-17 reports 95% bootstrap confidence intervals for the median rank across datasets. Appendix G.1 further explains the bootstrap procedure.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix E.5 specifies the compute resources used for pretraining, checkpoint selection, and benchmarking. The computational-efficiency comparison in Figure 1 and the timeout/failure protocol in Appendix E.6 further clarify the practical compute costs needed to reproduce the experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research involves no crowdsourcing, human-subject experiments, or deployment on individual-level clinical decision-making systems.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.

- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Section 6 discusses potential positive impacts of improved survival prediction in healthcare and other high-impact domains. It also discusses potential negative impacts and misuse risks through the method's limitations, especially invalid use under dependent censoring, and deployment in settings where assumptions may not hold.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The released assets only contain code, model weights, and synthetic-prior training procedures for tabular survival prediction.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The paper cites the original sources for the real-world survival datasets, baseline implementations, and prior-data fitted network components in Appendices F, E.1, and D.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.

- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigor, or originality of the research.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.