Derivative-Based Neural Modelling of Cumulative Distribution Functions for Survival Analysis

Dominic Danks
The Alan Turing Institute
University of Birmingham

Christopher Yau
University of Manchester
University of Oxford
Health Data Research UK

Abstract

Survival models — particularly those able to account for patient comorbidities via competing risks analysis — offer valuable prognostic information to clinicians making critical decisions and represent a growing area of application for machine learning approaches. However, current methods typically involve restrictive parameterisations, discretisation of time or the modelling of only one event cause. In this paper, we highlight how general cumulative distribution functions can be naturally expressed via neural network-based ordinary differential equations and how this observation can be utilised in survival analysis. In particular, we present DeSurv, a neural derivative-based approach capable of avoiding the aforementioned restrictions and flexibly modelling competing-risk survival data in continuous time. We apply DeSurv to both single-risk and competing-risk synthetic and real-world datasets and obtain results which compare favourably with current state-of-the-art models.

1 INTRODUCTION

1.1 Background

Survival analysis, also known as time-to-event analysis, is of fundamental importance to a wide variety of applied fields. In finance it is used to estimate default risk over time (Green and Shoven, 1986; Baesens et al., 2005; Dirick et al., 2017; Blumenstock et al., 2020), whilst in manufacturing and operational settings, estimating components' time-to-failure allows for maintenance planning and yield prediction (Ma and Krings, 2008; Susto et al., 2015; Ozturk et al., 2018). However, perhaps the most prominent application of survival analysis - and that which is the focus of this paper - is prognosis prediction in healthcare settings. In this context, the problem of interest is to predict the personalised survival distribution of a patient given their recorded covariates, in turn providing prognostic value to clinicians (Figure 1). Although survival analysis has long been applied to medical data, increased availability of healthcare data, in particular via personalised electronic health records (EHRs), has led to growing interest from the machine learning community.

1.2 Motivation

In many real-world settings, patients often suffer from a number of diseases simultaneously (comorbidities) which all pose prognostic risk. To correctly capture survival behaviours in this context, these *competing risks* must be explicitly accounted for, however few survival analysis models actively address this problem. Furthermore, those that do tend to have undesirable aspects, e.g. misspecification issues (Austin et al., 2021), discrete time (Lee et al., 2018) or restrictive parameterisations (Fine and Gray, 1999).

Many of these aspects are introduced into the models due to the challenge of continuously and generally modelling the survival functions (or equivalently cumulative distribution functions) involved. In this paper, we provide a method suited to this challenge by using the fact that the governing constraints on the functions of interest are naturally expressed via deep neural network-based ordinary differential equations. We then show how this in turn provides an unrestricted approach to competing risks survival analysis.

1.3 Contribution

Established approaches to survival analysis typically either i) do not generalise to competing-risk settings, ii) require the discretisation of time or iii) involve re-

Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151.

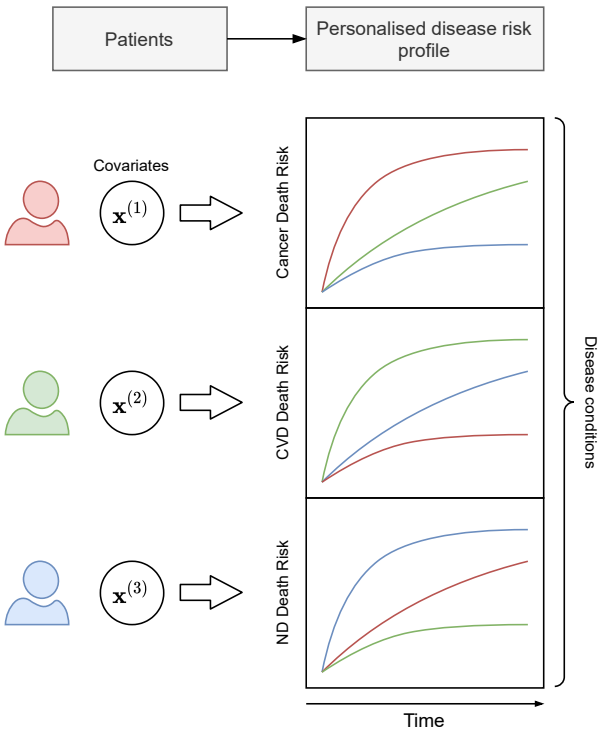

Figure 1: **Competing Risks Survival Analysis (CRSA).** Given a patient's covariates $\mathbf{x}^{(i)}$, CRSA provides a personalised risk profile for each disease event, for example death due to cancer, cardiovascular disease (CVD) or neurological disease (ND).

strictive parameterisations. In this paper, we use the fact that survival analysis is an area in which flexible models of cumulative distribution functions (CDFs) are of critical importance in order to develop a new class of deep survival analysis model capable of dealing with competing risks in continuous time.

More specifically, our contributions are to: 1) Demonstrate how general time-to-event CDFs can be intuitively modelled via neural networks (NNs) and ordinary differential equations (ODEs) (DeCDF). (Sections 2.1–2.2). 2) Demonstrate an equivalence between DeCDF and certain normalising flow models. (Section 2.3). 3) Show how DeCDF can be immediately applied to allow for continuous-time, single-risk, deep learning-based survival analysis. (Section 3). 4) Show how DeCDF can be extended to model *cumulative incidence functions* (CIFs) and therefore perform competing-risk survival analysis (DeSurv). (Section 4). 5) Demonstrate our models in the single- and competing-risk settings using synthetic and real-world examples, showing strong empirical performance whilst allowing for both competing risks and continuous time within the same framework. (Section 5).

1.4 Related Work

In recent years, healthcare-based survival analysis has seen increased attention from numerous disciplines, including machine learning. Such ML-based methods have taken a variety of approaches to attempt to improve classical methods. One common continuous-time approach for single-risk data, instigated by Faraggi and Simon (1995), is to build upon the Cox proportional hazards (CPH) model (Cox, 1972). Faraggi and Simon (1995) replaced the linear log-risk function of the standard Cox model with a simple neural network but found insignificant performance benefit. Katzman et al. (2018) revisited this approach and found that by employing improved modern deep learning machinery, their similar model (DeepSurv) could outperform the linear CPH model. Kvamme et al. (2019)'s Cox-Time model retains the Cox formalism of DeepSurv (Katzman et al., 2018) but introduces non-proportionality in the hazard function by involving time in the log-risk function. Kvamme et al. (2019) also approximate the standard Cox partial likelihood loss to allow for greater compatibility with gradient descent-based learning. Yousefi et al. (2017) and Zhu et al. (2016, 2017) apply NN-based Cox methodology to genomic and imaging data respectively. Two recent works with links to ours are Tang et al. (2021) and Ausset et al. (2021). Tang et al. (2021)'s SODEN model uses ODEs, however it uses them to parameterise the cumulative hazard function, rather than the CDF as we do. It also uses full ODE solves rather than quadrature and does not consider the treatment of competing risks. Ausset et al. (2021) consider how normalising flows can be applied to single-risk survival analysis, again using full ODE solves and not considering competing risks. Other works to apply ML methods to continuous-time single-risk survival data include piecewise-constant hazard models (Friedman, 1982; Fornili et al., 2014; Kvamme and Ørnulf Borgan, 2019), Deep Exponential Family models (Ranganath et al., 2016), and Random Forest models (Breiman, 2001; Ishwaran et al., 2008; Mogensen et al., 2012; Dietrich et al., 2016).

Many survival analysis approaches discretise time into intervals, allowing the implementation convenient use of fixed and finite model outputs to compute risk at each designated time step. For example, the Logistic Hazard (Brown, 1975), PLANN (Biganzoli et al., 1998) and Nnet-Survival (Gensheimer and Narasimhan, 2019) models parameterise the conditional hazard (the probability of an event in an interval given survival up to that interval) at each time step in this manner. Meanwhile, the (N-)MTLR (Yu et al., 2011; Fotso, 2018) and DeepHit (Lee et al., 2018) models target the probability mass function (PMF) for event occurrence directly.

Apart from DeepHit (Lee et al., 2018), the aforemen-

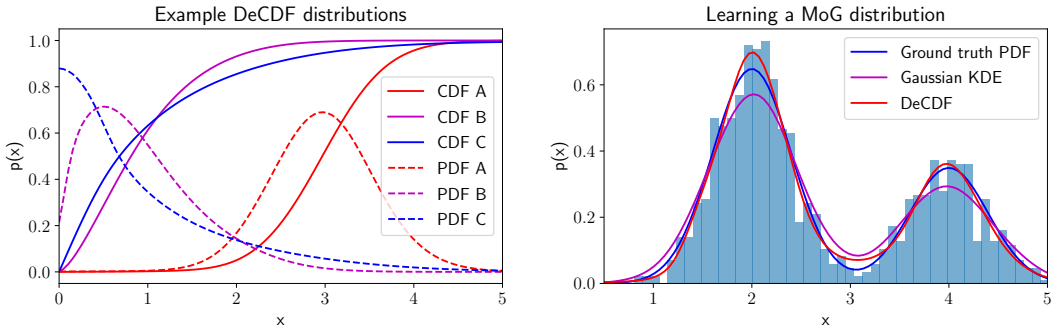

Figure 2: **Modelling Distributions with DeCDF.** DeCDF allows flexible modelling of general time-to-event distributions. Left: Example CDFs and corresponding PDFs modelled by DeCDF. Right: An example of DeCDF learning to model a rich distribution from data, in this case a Mixture of Gaussians (MoG) distribution.

tioned models do not consider competing risks. Furthermore, proper treatment of competing risks is not typical in the wider survival analysis literature, and it has been noted that greater attention should be to this aspect of survival data (Koller et al., 2012). Whilst it is technically possible to apply single-risk methods to competing-risk data by considering only one event type a valid event and other event types as censoring, notable bias is introduced (Austin et al., 2016). One approach to perform competing risks analysis is the Fine-Gray subdistribution hazard model (Fine and Gray, 1999; Austin et al., 2016), however, similarly to the standard Cox model, this involves restrictive assumptions on the hazard's functional form and its covariant dependence. Furthermore, the model has problematic aspects such as outputting potentially invalid overall time-to-failure CDFs capable of exceeding 1 (Austin et al., 2021). Some recent works have addressed competing risks from a modern ML point of view. For example, DMGPSA (Alaa and van der Schaar, 2017) expresses patient hitting times with respect to each event type via Deep Multi-Task Gaussian Processes (Damianou and Lawrence, 2013). Meanwhile, DeepHit (Lee et al., 2018) addresses competing risks by explicitly parameterising the joint PMF over failure time and event type using a large softmax-based neural network architecture.

The objective of our work (DeSurv) is to provide a natural, continuous time-based model that permits flexible modelling of survival functions whilst also allowing for both single- and competing-risk settings. It can therefore be thought of as an extension of DeepHit to incorporate continuous time, or as an extension of a method such as DeepSurv to incorporate competing risks.

2 NEURAL CDF MODELLING

In this section, we describe how a cumulative distribution function can be naturally represented in terms of neural networks and ordinary differential equations.

2.1 Problem Setup

Let T be a random variable representing time to failure with support $\mathbb{R}_{\geq 0}$ and denote its probability density function (PDF) and cumulative distribution function (CDF) by $f(t)$ and $F(t)$ respectively. $F(t)$ is a valid CDF if it satisfies

1. $F(0) = 0$
2. $F(t + a) \geq F(t) \quad \forall a > 0$
3. $\lim_{t\to\infty} F(t) = 1$.

Condition 1 can be considered an initial condition, whilst condition 2 is a monotonicity condition. Both are naturally expressed in the language of ordinary differential equation (ODE) initial value problems (IVPs). More specifically, specifying $F(0) = 0$ as an initial condition and enforcing $\frac{\mathrm{d}F}{\mathrm{d}t} \geq 0$ in an IVP will ensure condition 1 and 2 are satisfied. The simultaneous satisfaction of condition 3 is a matter of practitioner choice.

2.2 DeCDF

Our approach, which we denote DeCDF, is to model an underlying strictly monotonic function $u(t)$ as an integral, i.e. to express its derivative in terms of t only, and then apply a suitable transformation to $u(t)$ to obtain $F(t)$. Perhaps the most immediate ML-aware example of this is to let $g(t)$ be a neural network with positive range and set $F(t) = H(u(t))$, where

$$\frac{\mathrm{d}u}{\mathrm{d}t} = g(t); \; H(x) = \tanh x \quad \text{and} \quad u(0) = 0. \tag{1}$$

Within this formalism, $u(t)$ can be computed using Gauss-Legendre (GL) quadrature of order n (we use

Table 1: **Survival Datasets.** Details of the datasets used to demonstrate model performance.

Dataset		Observed	Censored	Features	Event time		Censoring time	
					Mean	Max	Mean	Max
METABRIC		1103 (58%)	801 (42%)	9	100	355	160	337
SUPPORT		6036 (68%)	2837 (32%)	14	6.85	64.8	35.3	67.6
Synth	Cause 1	5699 (19%)	15000 (50%)	12	0.080	0.988	0.043	1.28
	Cause 2	9301 (31%)			0.086	1.11		
SEER	BC	116170 (19%)	442685 (74%)	23	57.8	311	112.1	311
	CVD	39639 (7%)			95.4	311		

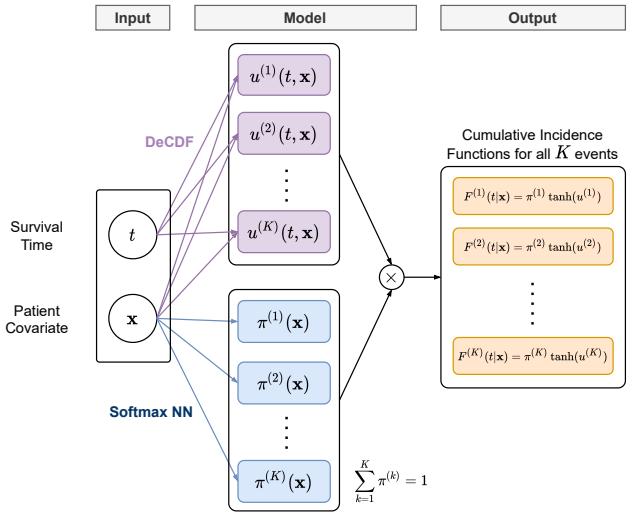

Figure 3: **DeSurv Model Architecture.** DeSurv flexibly models the cumulative incidence function of each event by performing elementwise multiplication between the outputs of a DeCDF block and a softmax neural network block. In the absence of competing risks, only the DeCDF block is required.

$n = 15$ throughout) via

$$u(t) = \frac{t}{2} \sum_{i=1}^{n} w_i g\left(\frac{t}{2}(u_i + 1)\right),$$

with u_i, w_i the order n GL quadrature nodes and weights respectively, computed via Legendre polynomials (see e.g. Press et al. (2007) for details). This computation has defined computational complexity and can be readily backpropagated and parallelised for efficient implementation in a variety of frameworks. Example outputs of DeCDF are shown in the left panel of Figure 2, whilst the right panel demonstrates DeCDF's ability to estimate a Mixture of Gaussians distribution from the displayed histogram data. Further details of this experiment can be found in Appendix A.

2.3 Normalising Flow Viewpoint

The form of (1) arises intuitively when considering how to model a CDF in terms of its derivatives, however it is natural to ask which functions could be used in place of tanh and whether there are any shortcomings in the generality of this specification. In fact, tanh can be replaced with any strictly increasing CDF on $\mathbb{R}_{\geq 0}$ and the method is able to model any well-behaved time-to-event distribution. This can be seen by considering the problem from the perspective of normalising flows.

Normalising flows (Rezende and Mohamed, 2016; Kobyzev et al., 2020; Papamakarios et al., 2021) model rich distributions of a random variable X by considering X to be the result of applying a learnt transformation $\mathcal{T}$ to an underlying random variable Z which follows a simple distribution. This in turn defines the density of X in terms of the density of Z and $\mathcal{T}$. When performing normalising flow analysis, one chooses to model $\mathcal{T}$ (the generative direction) or $\mathcal{T}^{-1}$ (the normalising direction). Modelling $\mathcal{T}$ allows for straightforward sampling of X, whilst modelling $\mathcal{T}^{-1}$ allows for straightforward density evaluation of X. It has been shown (see e.g. Papamakarios et al. (2021)) that normalising flows are capable of capturing arbitrarily complex X distributions using arbitrarily simple Z distributions.

To elucidate the link between normalising flows and DeCDF, first note that (1) implies a density on T of

$$p_t(t) = \frac{\mathrm{d}F}{\mathrm{d}t} = H'(u(t))\frac{\mathrm{d}u}{\mathrm{d}t} = H'(u(t))g(t). \quad (2)$$

Now let $p_z(z)$ denote the PDF of the random variable Z with support $\mathbb{R}_{\geq 0}$ and let $\mathcal{T}^{-1}(t) = u(t)$, with $u(t)$ as in (1). Applying change of variables provides the density on T as

$$p_t(t) = p_z(u(t))\frac{\mathrm{d}u}{\mathrm{d}t} = p_z(u(t))g(t). \quad (3)$$

Comparing (2) and (3), we see that $H'(u(t))$, that is the derivative of the transformation function in the DeCDF context, can be seen as the base distribution

Table 2: **METABRIC Performance Metrics.**

Method	$\mathbf{C}_{\text{td}}$	IBS	INBLL	MAE	RMSE
Cox	0.621 ± 0.035	0.176 ± 0.009	0.523 ± 0.021	75.60 ± 4.86	87.76 ± 4.86
Cox-CC	0.634 ± 0.012	0.177 ± 0.007	0.522 ± 0.017	74.15 ± 4.54	86.99 ± 4.10
DeepSurv	0.634 ± 0.019	0.175 ± 0.006	0.519 ± 0.014	74.45 ± 4.65	87.36 ± 4.98
Cox-Time	0.664 ± 0.019	0.175 ± 0.005	0.518 ± 0.016	66.59 ± 2.90	78.60 ± 3.07
DeepHit	$\mathbf{0.673 \pm 0.013}$	0.188 ± 0.006	0.549 ± 0.013	88.3 ± 5.05	100.11 ± 4.90
DeSurv	0.660 ± 0.022	$\mathbf{0.172 \pm 0.008}$	$\mathbf{0.509 \pm 0.020}$	$\mathbf{54.76 \pm 2.14}$	$\mathbf{65.5 \pm 2.44}$

PDF in a normalising flow viewpoint. $H(x)$ can therefore be taken to be any strictly increasing CDF function on $\mathbb{R}_{\geq 0}$ and (1) remains valid. Meanwhile, the underlying monotonic function, $u(t)$, which is transformed by H in the DeCDF context can be seen as $\mathcal{T}^{-1}$ in normalising flow language. This means that $u(t)$ could be modelled via established normalising flow invertible functions to achieve similar functionality to our approach. We return to this point in the discussion as a recommendation for future work. Furthermore, the above relationship demonstrates the general expressivity of DeCDF and hence its suitability for flexible time-to-event distribution modelling.

3 DeCDF FOR SINGLE-RISK SURVIVAL ANALYSIS

3.1 Problem Setting

Survival analysis in the presence of a single risk considers data which for N subjects takes the form $\mathcal{D} = \{s^{(i)}, \mathbf{x}^{(i)}, k^{(i)}\}_{i=1}^N$, where $s^{(i)}$ denotes time-to-event, $\mathbf{x}^{(i)}$ denotes subject covariates and $k^{(i)}$ denotes event type. In the single-risk setting, $k^{(i)} \in \{\varnothing, 1\}$, with $k^{(i)} = 1$ implying the event of focus (e.g. death) occurred at $s^{(i)}$ and $k^{(i)} = \varnothing$ implying right censorship, that is the subject being lost to follow up, often for an unknown reason. Given training data $\mathcal{D}_{\text{train}} = \{s^{(i)}, \mathbf{x}^{(i)}, k^{(i)}\}_{i=1}^N$, the goal is to learn a model which is capable of providing accurate *personalised* prognostic cumulative distribution functions of the survival time for test subjects. That is, given covariates of a test subject, $\mathbf{x}$, be able to provide an accurate estimate of $F(t|\mathbf{x}) = P(T \leq t|\mathbf{x})$.

3.2 Applying DeCDF

Given DeCDF's ability to model CDFs, we can directly utilise its methodology for this modelling purpose by simply adding $\mathbf{x}$ as an additional input to the model. More specifically, we let $g_\phi(\mathbf{x}, t)$ be a neural network with parameters ϕ and positive output range (enforced via softplus) and set $\hat{F}(t|\mathbf{x}) = \tanh(u(t|\mathbf{x}))$, where

$$\frac{\mathrm{d}u}{\mathrm{d}t} = g_\phi(\mathbf{x}, t) \quad \text{with} \quad u(0) = 0.$$

Note that, by the argument of Section 2.3, tanh could be replaced by any valid CDF on $\mathbb{R}_{\geq 0}$.

The negative log-likelihood for data $\mathcal{D} = \{s^{(i)}, \mathbf{x}^{(i)}, k^{(i)}\}_{i=1}^N$ and parameters ϕ is

$$\mathcal{L} = -\sum_{i=1}^{N} \Bigg[\mathbb{1}\left(k^{(i)} = \varnothing\right) \log\left(1 - \hat{F}(s^{(i)}|\mathbf{x}^{(i)})\right) + \mathbb{1}\left(k^{(i)} \neq \varnothing\right) \log\left(\hat{F}'(s^{(i)}|\mathbf{x}^{(i)})\right) \Bigg].$$

We use this as our training loss and apply minibatch gradient descent via Adam (Kingma and Ba, 2014) to optimise the neural network parameters ϕ.

4 EXTENDING TO COMPETING RISKS: DeSurv

4.1 Problem Setting

In the competing risks setting, survival data again takes the form $\mathcal{D} = \{s^{(i)}, \mathbf{x}^{(i)}, k^{(i)}\}_{i=1}^N$, but now $k^{(i)} \in \{\varnothing, 1, 2, \ldots, K\}$ for K event types of interest. Typically, these event types code for "death due to cause k". For example, in a study of cancer patients, some participants may suffer from comorbidities such as cardiovascular disease (CVD) and may die due to these *competing risks* rather than cancer itself.

Whilst the overall survival time CDF, $F(t|\mathbf{x})$, is of interest in the competing risks context, the quantity of greatest interest is the cumulative incidence function (CIF), $F_k(t|\mathbf{x})$, associated with each event type. This is defined as $F_k(t|\mathbf{x}) = P(T \leq t, k|\mathbf{x})$, and therefore encodes the probability of experiencing event k before time t *accounting for the presence of competing risks*. Marginalising out the event type provides the overall survival time CDF as $F(t|\mathbf{x}) = \sum_{k=1}^K F_k(t|\mathbf{x})$.

4.2 DeSurv

DeCDF can be adapted to the competing risks setting by noting that each cumulative incidence function is a scaled CDF as $F_k(t|\mathbf{x}) = P(T \leq t, k|\mathbf{x}) = P(T \leq t|k, \mathbf{x})P(k|\mathbf{x})$ where the conditional CDF, $P(T \leq t|k, \mathbf{x})$, which we denote $\tilde{F}_k(t|\mathbf{x})$, is a valid

Table 3: **SUPPORT Performance Metrics.**

Method	$\mathbf{C}_{\text{td}}$	IBS	INBLL	MAE	RMSE
Cox	0.562 ± 0.005	0.211 ± 0.003	0.609 ± 0.006	8.05 ± 0.17	10.86 ± 0.31
Cox-CC	0.600 ± 0.007	0.200 ± 0.004	0.588 ± 0.013	8.11 ± 0.16	10.85 ± 0.29
DeepSurv	0.603 ± 0.006	0.199 ± 0.004	0.585 ± 0.013	8.09 ± 0.17	10.85 ± 0.29
Cox-Time	0.612 ± 0.008	0.200 ± 0.004	0.586 ± 0.010	7.94 ± 0.21	10.84 ± 0.32
DeepHit	0.619 ± 0.010	0.216 ± 0.002	0.622 ± 0.004	20.37 ± 1.08	21.73 ± 1.14
DeSurv	$\mathbf{0.622 \pm 0.010}$	$\mathbf{0.199 \pm 0.003}$	$\mathbf{0.583 \pm 0.010}$	$\mathbf{7.86 \pm 0.17}$	$\mathbf{10.75 \pm 0.22}$

CDF and $P(k|\mathbf{x})$, which we denote $\pi_k(\mathbf{x})$, satisfies $\sum_{k=1}^{K} \pi_k(\mathbf{x}) = 1$. The CIF can therefore be captured by simultaneously modelling $\{\tilde{F}_k\}_{k=1}^{K}$ via a DeCDF approach and $\{\pi_k(\mathbf{x})\}_{k=1}^{K}$ via a neural network with softmax activation. We refer to this model as DeSurv. The architecture for DeSurv is shown in Figure 3.

The negative log-likelihood for data $\mathcal{D} = \{s^{(i)}, \mathbf{x}^{(i)}, k^{(i)}\}_{i=1}^{N}$ and parameters ϕ is

$$\mathcal{L} = -\sum_{i=1}^{N} \left[\mathbb{1}\left(k^{(i)} = \varnothing\right) \log\left(1 - \sum_{k=1}^{K} F_k(s^{(i)}|\mathbf{x}^{(i)})\right) + \mathbb{1}\left(k^{(i)} \neq \varnothing\right) \log\left(F'_{k^{(i)}}(s^{(i)}|\mathbf{x}^{(i)})\right) \right].$$

As in the single risk case, we take this to be our loss and optimise via Adam (Kingma and Ba, 2014) to learn the parameters ϕ of the neural networks describing the $\{\tilde{F}_k\}_{k=1}^{K}$ and $\{\pi_k\}_{k=1}^{K}$ mappings.

5 EXPERIMENTS

In this section, we demonstrate the behaviour of DeSurv and other related approaches on real and synthetic datasets from single-risk and competing-risk settings. We utilise the same neural network architectures for each method as far as possible and provide hyperparameter settings in the supplementary material. Train/Validation/Test splits are 64%/16%/20% in all cases. Table entries are provided as mean $\pm$ std and are calculated by re-running experiments (including data partitioning) in their entirety 10 times, each time with a new random seed drawn from a seeded RNG. The computational requirements of all involved algorithms were relatively low, with all experiments able to run within hours on a 2.7 GHz Quad-Core i7 processor. A complete PyTorch-based (Paszke et al., 2019) implementation of DeSurv can be found at `https://github.com/djdanks/DeSurv`.

5.1 Single-Risk Survival Analysis

We evaluate our model in the single-risk setting alongside various single-risk benchmarks discussed in section 1.4 on two anonymised real-world medical datasets: METABRIC and SUPPORT.

5.1.1 Datasets

The METABRIC dataset contains gene expression profiles and clinical covariates of breast cancer (BC) patients for the purposes of BC subgroup identification. We follow the practice of Katzman et al. (2018), and utilise 9 features, namely 4 gene indicators (*MKI67*, *EGFR*, *PGR*, and *ERBB2*) and 5 clinical covariates (hormone treatment indicator, radiotherapy indicator, chemotherapy indicator, ER-positive indicator and age at diagnosis). SUPPORT (Knaus et al., 1995) is a larger dataset derived from a prognostic study of seriously ill patients and contains 14 covariates. We prepare the dataset as in Katzman et al. (2018). Further dataset details are provided in Table 1.

5.1.2 Metrics

We evaluate the models using the five metrics detailed below. We evaluate the metrics using the `pycox` library (Kvamme et al., 2019).

Time-dependent Concordance Index (Antolini et al., 2005): We define the Cause-specific Time-dependent Concordance Index for event k so that the metric is immediately applicable to the competing risks setting. Replacing CIF with CDF in what follows provides the single-risk version. Consider two patients i and j. Suppose patient i experiences event $k^{(i)}$ at time $s^{(i)}$ and that patient j outlives patient i so that $s^{(i)} < s^{(j)}$. Then, the time-dependent concordance represents the probability that the predicted CIF of patient i for event $k^{(i)}$ exceeds the same event's predicted CIF for patient j. Intuitively, it argues that if the model is good, the predicted cumulative risk at time $s^{(i)}$ should be greater for the patient who experiences an event at that time than for the patient that remains alive beyond $s^{(i)}$. The associated formal definition and empirical computation of this metric is presented in Appendix B.

Integrated Brier Score (IBS): The Brier score (BS) is a metric which measures the ability of a classifier to predict binary outcomes. Given N binary observations

Table 4: **Competing Risks Concordance Values.**

	SEER		Synthetic	
Method	$\mathbf{C_{td}}^{(BC)}$	$\mathbf{C_{td}}^{(CVD)}$	$\mathbf{C_{td}}^{(1)}$	$\mathbf{C_{td}}^{(2)}$
DeepHit	$\mathbf{0.833 \pm 0.002}$	0.873 ± 0.002	0.611 ± 0.007	0.571 ± 0.004
DeSurv	0.826 ± 0.002	$\mathbf{0.879 \pm 0.002}$	$\mathbf{0.629 \pm 0.009}$	$\mathbf{0.587 \pm 0.005}$

Table 5: **Additional Competing Risks Metrics.**

Dataset	Method	$\text{RMSE}^{(1)}$	$\text{RMSE}^{(2)}$	BCE	Acc.
SEER	DeepHit	97.86 ± 6.28	103.65 ± 5.77	1.330 ± 0.004	75.01 ± 0.83
	DeSurv	$\mathbf{47.46 \pm 0.53}$	$\mathbf{76.82 \pm 1.17}$	$\mathbf{1.150 \pm 0.006}$	$\mathbf{79.73 \pm 0.33}$
Synthetic	DeepHit	0.0956 ± 0.0030	0.1032 ± 0.0013	1.396 ± 0.001	$\mathbf{62.02 \pm 1.04}$
	DeSurv	$\mathbf{0.0944 \pm 0.0022}$	$\mathbf{0.1013 \pm 0.0016}$	$\mathbf{1.389 \pm 0.003}$	61.78 ± 0.91

$y_i \in \{0, 1\}$ and N predictions $\hat{p}_i$ from a binary classifier for $p(y_i = 1)$, the BS is $\sum_i (y_i - \hat{p}_i)^2 / N$. In the survival analysis setting, a Brier score, $\text{BS}(t)$ can be obtained for N patients at time t by considering the Brier score of the binary variable U_i which is 1 for patients who live beyond t and 0 for those that experience an event before t (see Appendix B for a full mathematical definition). Numerically integrating $\text{BS}(t)$ over the range of test times and dividing by the interval provides the Integrated Brier Score (IBS) we use. Smaller values indicate better classification performance (Graf et al., 1999).

Integrated Negative Binomial Log-Likelihood (INBLL): Following on from the above IBS discussion definition, we can define the negative log-likelihood of the variable $U = \{U_i\}_{i=1}^N$ and integrate this over time (see Appendix B for a full mathematical definition) to obtain the Integrated Negative Binomial Log-Likelihood (INBLL), a metric for which smaller values indicate better classification performance (Kvamme et al., 2019).

Mean Absolute Error (MAE) and Root Mean Square Error (RMSE): These represent the MAE and RMSE between the mean of a model's patient-specific time-to-event distribution and that patient's observed event time calculated across all (non-censored) patients.

5.2 Competing Risks Survival Analysis

We evaluate DeSurv in the competing risks setting using the methodology of Section 4 on real and synthetic data alongside the established deep learning-based competing risks model DeepHit (Lee et al., 2018), which provides discretised time output.

5.2.1 Datasets

The Surveillance, Epidemiology, and End Results Program (SEER) dataset contains anonymised survival data from cancer patients in the United States. We extract breast cancer (BC) patients from the database and select those patients that either i) were right censored, ii) died from BC or iii) died from cardiovascular disease (CVD) (see Appendix C for further details), yielding data with censorship and two competing risks (see Table 1).

We also generate a synthetic dataset of the style of that in Lee et al. (2018) with $N = 30,000$ examples drawn from the generative process

$$\mathbf{x}_1^{(i)}, \mathbf{x}_2^{(i)}, \mathbf{x}_3^{(i)} \sim U[\mathbf{0}, \mathbf{1}]$$
$$T_k^{(i)} \sim \text{Exp}\left(\left(\boldsymbol{\gamma}_3^T \mathbf{x}_3^{(i)}\right)^2 + \boldsymbol{\gamma}_k^T \mathbf{x}_k^{(i)}\right) \equiv \text{Exp}\left(\lambda_k^{(i)}\right),$$

for $k = 1, 2$ with $(\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2, \boldsymbol{\gamma}_3) = (\mathbf{0.5}, \mathbf{2}, \mathbf{1})$ and applied random censoring to 50% of examples (see Table 1).

5.3 Results Discussion

Tables 2 & 3 show that DeSurv outperforms alternative single-risk methods overall on both the METABRIC and SUPPORT datasets. DeepHit does demonstrate a higher concordance than DeSurv on METABRIC and a comparable concordance on SUPPORT, however its performance is notably poorer with respect to other metrics, suggesting limited calibration. Visual inspection of the survival functions associated with DeepHit relative to those associated with the performant Cox-Time and DeSurv models confirms this (see Figure 4). In particular, it can be seen that whilst the models generally agree on the ranking of patient risk (reflected in concordance values), the survival functions associated with DeepHit exhibit a lack of patient-level variation relative to those of DeSurv and Cox-Time.

This highlights the importance of looking beyond solely concordance values during model evaluation and comparison.

DeSurv also extends successfully to the competing-risk setting, outperforming DeepHit on the synthetic dataset and performing competitively with DeepHit with respect to concordance on SEER. Visualisations of SEER patient CIFs suggest that DeepHit's tendency to lack trajectory separation is also present in the competing risk case, whereas DeSurv captures a variety of risk patterns (Figure 5). Additional visual comparisons of survival functions and CIFs across the models and datasets are provided in Appendix D. Further insight can be gained via the additional evaluation metrics shown in Table 5, namely the RMSE for each of the competing risks as well as the binary cross entropy (BCE) and accuracy (Acc.) values associated with the task of predicting which event type occurred for those that experienced an event. DeSurv again demonstrates a strong calibration performance relative to DeepHit, particularly on SEER, despite demonstrating similar concordance performance, again highlighting the importance of considering a variety of metrics when evaluating overall survival model performance.

6 DISCUSSION

We have shown how general time-to-event CDFs can be modelled naturally in the language of derivative-based models, yielding a general model for such functions, DeCDF. We have also demonstrated how DeCDF can be applied immediately to single-risk survival analysis and developed a novel DeCDF-based survival analysis method, DeSurv, capable of flexibly modelling competing-risk survival data in continuous time. DeSurv can be seen as an extension of DeepHit (Lee et al., 2018) to continuous time or an extension of work such as DeepSurv (Katzman et al., 2018) and Cox-Time (Kvamme et al., 2019) to the competing-risk setting.

Our work also highlights a link between DeCDF and normalising flows, namely that the ODE solution and transformation function in DeCDF can be thought of as a normalisation direction transformation and base CDF respectively in normalising flow language (see Section 2.3). In this paper, we drew upon this equivalence to reinforce the expressivity of DeCDF. A natural extension of our work would be to explore this link further, for example by investigating the effect of flow-based transformation parameterisation and base distribution on performance. Another possible avenue for further work would be to investigate alternative ODE-based CDF parameterisations, an example of which is provided in Appendix E. It would also be of interest to apply DeCDF to CDF-based problems in other application settings.

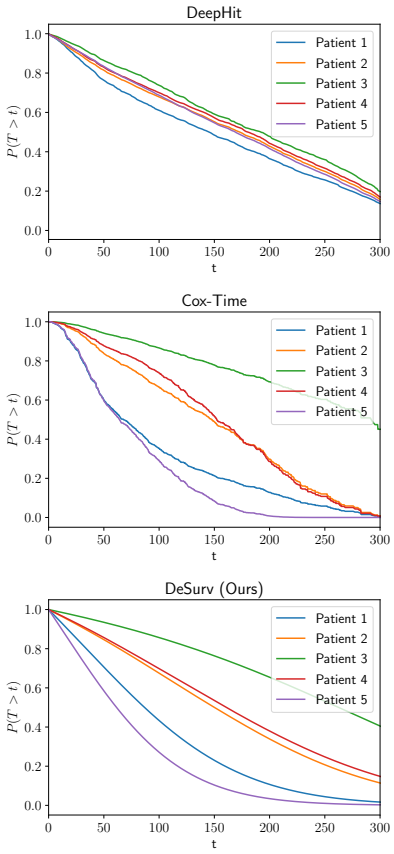

Figure 4: **METABRIC Survival Functions.** 25-year survival trajectories for five random test patients according to three of the considered models. Note that whilst the models agree on patient risk ranking, DeepHit exhibits limited trajectory separation relative to DeSurv and Cox-Time.

We also note that whilst we and others stress the importance of predictive performance of survival analysis models in prognostic settings, it is also of interest to understand how the personalised survival functions output by our model and others such as DeepHit (Lee et al., 2018) depend on covariates and hence which covariates are most associated with disease prognosis. The required sensitivity analysis for this work can be readily carried out on DeSurv and other deep learning models via automatic differentiation and would also represent a valuable contribution.

A key potential use case for deep survival analysis models such as DeSurv is to provide prognostic information to clinicians making critical decisions. Applying poorly performing models in this context would clearly adversely affect patient care and would represent a particularly negative social consequence. Hence, significant validation testing including human-computer

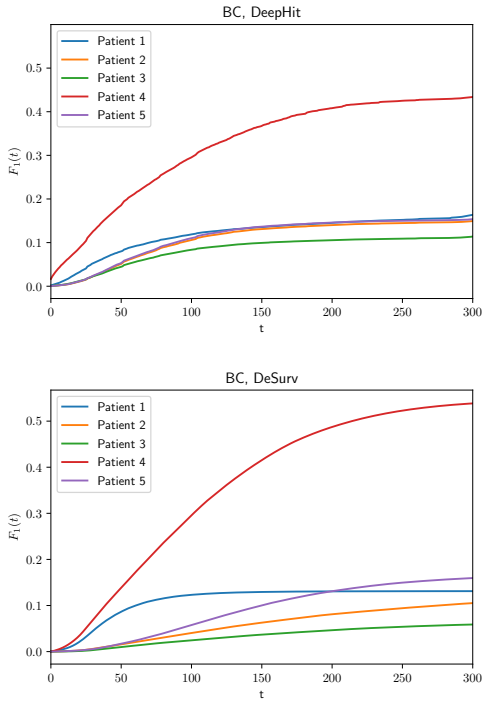

Figure 5: **SEER CIFs.** 25-year cumulative incidence functions for five random test patients according to DeepHit and DeSurv. As in the single-risk case (Figure 4), the models generally agree on patient risk rankings, however DeepHit exhibits reduced trajectory separation relative to DeSurv.

interaction elements would have to be performed on any model hoping to deploy in a clinical setting. Additionally, whilst a significant benefit of models such as DeSurv is their ability to learn complex dependencies between patient covariates and event risk profiles from data, this can also act as a limitation. In particular, if trained blindly on well-curated retrospective data, these models will inherit any biases which may be present, meaning they may not generalise well to the everyday clinical setting. When training deployment-stage models, training should be carried out in a bias-aware way in order to avoid systemic algorithmic inequality such as the racial bias recently shown to exist in a U.S. healthcare management algorithm (Obermeyer et al., 2019).

Acknowledgements

DD is supported by a Doctoral Studentship from the Alan Turing Institute (EPSRC Grant Ref: EP/N510129/1). CY is supported by an EPSRC Turing AI Acceleration Fellowship (Grant Ref: EP/V023233/1). CY acknowledges support from the NIHR Programme for Artificial Intelligence for Multiple Long-Term Conditions (Grant Ref: NIHR202632).

References

Ahmed M. Alaa and M. van der Schaar. 2017. Deep multi-task gaussian processes for survival analysis with competing risks. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Laura Antolini, Patrizia Boracchi, and Elia Biganzoli. 2005. A time-dependent discrimination index for survival data. *Statistics in Medicine*, 24(24):3927–3944.

Guillaume Ausset, Tom Ciffreo, François Portier, Stephan Clémençon, and Timothée Papin. 2021. Individual survival curves with conditional normalizing flows. *2021 IEEE 8th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 1–10.

Peter C. Austin, Douglas S. Lee, and Jason P. Fine. 2016. Introduction to the analysis of survival data in the presence of competing risks. *Circulation*, 133(6):601–609.

Peter C. Austin, Ewout W. Steyerberg, and Hein Putter. 2021. Fine-gray subdistribution hazard models to simultaneously estimate the absolute risk of different event types: Cumulative total failure probability may exceed 1. *Statistics in Medicine*.

B Baesens, T Van Gestel, M Stepanova, D Van den Poel, and J Vanthienen. 2005. Neural network survival analysis for personal loan data. *Journal of the Operational Research Society*, 56(9):1089–1098.

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

Gabriel Blumenstock, Stefan Lessmann, and Hsin-Vonn Seow. 2020. Deep learning for survival and competing risk modelling. *Journal of the Operational Research Society*, 0(0):1–13.

Leo Breiman. 2001. Random forests. *Machine Learning*, 45(1):5–32.

Charles C. Brown. 1975. On the use of indicator variables for studying the time-dependence of parameters in a response-time model. *Biometrics*, 31(4):863–872.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. 2018. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

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

Andreas Damianou and Neil D. Lawrence. 2013. Deep Gaussian processes. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, volume 31 of *Proceedings of Machine Learning Research*, pages 207–215, Scottsdale, Arizona, USA. PMLR.

Stefan Dietrich, Anna Floegel, Martina Troll, Tilman Kühn, Wolfgang Rathmann, Anette Peters, Disorn Sookthai, Martin von Bergen, Rudolf Kaaks, Jerzy Adamski, Cornelia Prehn, Heiner Boeing, Matthias B Schulze, Thomas Illig, Tobias Pischon, Sven Knüppel, Rui Wang-Sattler, and Dagmar Drogan. 2016. Random survival forest in practice: a method for modelling complex metabolomics data in time to event analysis. *International Journal of Epidemiology*, 45(5):1406–1420.

Lore Dirick, Gerda Claeskens, and Bart Baesens. 2017. Time to default in credit scoring using survival analysis: a benchmark study. *Journal of the Operational Research Society*, 68(6):652–665.

David Faraggi and Richard Simon. 1995. A neural network model for survival data. *Statistics in Medicine*, 14(1):73–82.

Jason P. Fine and Robert J. Gray. 1999. A proportional hazards model for the subdistribution of a competing risk. *Journal of the American Statistical Association*, 94(446):496–509.

Marco Fornili, Federico Ambrogi, Patrizia Boracchi, and Elia Biganzoli. 2014. Piecewise exponential artificial neural networks (peann) for modeling hazard function with right censored data. In *Computational Intelligence Methods for Bioinformatics and Biostatistics*, pages 125–136, Cham. Springer International Publishing.

Stephane Fotso. 2018. Deep neural networks for survival analysis based on a multi-task framework.

Michael Friedman. 1982. Piecewise Exponential Models for Survival Data with Covariates. *The Annals of Statistics*, 10(1):101 – 113.

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

Amir Gholami, Kurt Keutzer, and George Biros. 2019. Anode: Unconditionally accurate memory-efficient gradients for neural odes.

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

Jerry Green and John B. Shoven. 1986. The effects of interest rates on mortgage prepayments. *Journal of Money, Credit and Banking*, 18(1):41–59.

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

Jared L. Katzman, Uri Shaham, Alexander Cloninger, Jonathan Bates, Tingting Jiang, and Yuval Kluger. 2018. Deepsurv: personalized treatment recommender system using a cox proportional hazards deep neural network. *BMC Medical Research Methodology*, 18(1):24.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. Cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

William A. Knaus, Frank E. Harrell, Joanne Lynn, Lee Goldman, Russell S. Phillips, Alfred F. Connors, Neal V. Dawson, William J. Fulkerson, Robert M. Califf, Norman Desbiens, Peter Layde, Robert K. Oye, Paul E. Bellamy, Rosemarie B. Hakim, and Douglas P. Wagner. 1995. The support prognostic model: Objective estimates of survival for seriously ill hospitalized adults. *Annals of Internal Medicine*, 122(3):191–203.

Ivan Kobyzev, Simon Prince, and Marcus Brubaker. 2020. Normalizing flows: An introduction and review of current methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, page 1–1.

Michael T. Koller, Heike Raatz, Ewout W. Steyerberg, and Marcel Wolbers. 2012. Competing risks and the clinical community: irrelevance or ignorance? *Statistics in Medicine*, 31(11-12):1089–1097.

Håvard Kvamme, Ørnulf Borgan, and Ida Scheel. 2019. Time-to-event prediction with neural networks and cox regression. *Journal of Machine Learning Research*, 20(129):1–30.

Håvard Kvamme and Ørnulf Borgan. 2019. Continuous and discrete-time survival prediction with neural networks.

Changhee Lee, William Zame, Jinsung Yoon, and Mihaela van der Schaar. 2018. Deephit: A deep learning approach to survival analysis with competing risks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Zhanshan Ma and Axel W. Krings. 2008. Survival analysis approach to reliability, survivability and prognostics and health management (phm). In *2008 IEEE Aerospace Conference*, pages 1–20.

Ulla B. Mogensen, Hemant Ishwaran, and Thomas A. Gerds. 2012. Evaluating random forests for survival analysis using prediction error curves. *Journal of Statistical Software, Articles*, 50(11):1–23.

Ziad Obermeyer, Brian Powers, Christine Vogeli, and Sendhil Mullainathan. 2019. Dissecting racial bias in an algorithm used to manage the health of populations. *Science*, 366(6464):447–453.

Samet Ozturk, Vasilis Fthenakis, and Stefan Faulstich. 2018. Assessing the factors impacting on the reliability of wind turbines via survival analysis—a case study. *Energies*, 11(11).

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. 2021. Normalizing flows for probabilistic modeling and inference.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. 2007. *Numerical Recipes 3rd Edition: The Art of Scientific Computing*, 3 edition. Cambridge University Press, USA.

Christopher Rackauckas, Yingbo Ma, Vaibhav Dixit, Xingjian Guo, Mike Innes, Jarrett Revels, Joakim Nyberg, and Vijay Ivaturi. 2018. A comparison of automatic differentiation and continuous sensitivity analysis for derivatives of differential equation solutions. *CoRR*, abs/1812.01892.

Christopher Rackauckas and Qing Nie. 2017. Differentialequations.jl – a performant and feature-rich ecosystem for solving differential equations in julia. *The Journal of Open Research Software*, 5(1).

Rajesh Ranganath, Adler Perotte, Noémie Elhadad, and David Blei. 2016. Deep survival analysis.

Danilo Jimenez Rezende and Shakir Mohamed. 2016. Variational inference with normalizing flows.

Gian Antonio Susto, Andrea Schirru, Simone Pampuri, Seán McLoone, and Alessandro Beghi. 2015. Machine learning for predictive maintenance: A multiple classifier approach. *IEEE Transactions on Industrial Informatics*, 11(3):812–820.

Weijing Tang, Jiaqi Ma, Qiaozhu Mei, and Ji Zhu. 2021. Soden: A scalable continuous-time survival model through ordinary differential equation networks.

Safoora Yousefi, Fatemeh Amrollahi, Mohamed Amgad, Chengliang Dong, Joshua E. Lewis, Congzheng Song, David A. Gutman, Sameer H. Halani, Jose Enrique Velazquez Vega, Daniel J. Brat, and Lee A. D. Cooper. 2017. Predicting clinical outcomes from large scale cancer genomic profiles with deep survival models. *Scientific Reports*, 7(1):11707.

Chun-Nam Yu, Russell Greiner, Hsiu-Chin Lin, and Vickie Baracos. 2011. Learning patient-specific cancer survival distributions as a sequence of dependent regressors. In *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc.

Xinliang Zhu, Jiawen Yao, and Junzhou Huang. 2016. Deep convolutional neural network for survival analysis with pathological images. In *2016 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 544–547.

Xinliang Zhu, Jiawen Yao, Feiyun Zhu, and Junzhou Huang. 2017. Wsisa: Making survival prediction from whole slide histopathological images. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6855–6863.

Supplementary Material: Derivative-Based Neural Modelling of Cumulative Distribution Functions for Survival Analysis

A CDF AND PDF ESTIMATION FROM DATA VIA DeCDF

Figure 2 of the main paper visualises the ability of DeCDF to represent varied time-to-event distributions. Here we present in more detail how the figure was obtained.

A.1 Distribution Estimation From Data

In the univariate case considered in Figure 2, DeCDF with a parameter setting ϕ takes in x and outputs a valid CDF $F_\phi(x)$. The density associated with this CDF is $f_\phi(x) = \frac{\mathrm{d}F_\phi}{\mathrm{d}x}$ which, assuming the setup in Section 2.2 of the main text, can be calculated as $f_\phi(x) = (1 - F_\phi(x)^2)g_\phi(x)$, where $g_\phi(x)$ represents the underlying positive-output NN (see Section 2.2 of the main text).

Given N independent and identically distributed samples $\mathcal{D} = \{x_i\}_{i=1}^N$ from a continuous distribution $\mathcal{P}$, the negative log-likelihood under the DeCDF model is

$$\mathcal{L}(\phi; \mathcal{D}) = -\left[\sum_{i=1}^{N} \log\left(1 - F_\phi(x_i)^2\right) + \log\left(g_\phi(x_i)\right)\right]. \tag{4}$$

Minimising this with respect to ϕ provides an estimate for the distribution $\mathcal{P}$.

A.2 Figure 2, Left Panel

To generate this figure, 1000 samples were drawn from each of 3 distributions, namely A) a Gaussian with $\mu = 3, \sigma = 0.6$, B) a Weibull with $k = 1.5, \lambda = 1$, and C) a Weibull with $k = 1, \lambda = 1$. The parameters of DeCDF were trained using the loss above to begin to emulate these distributions. The aim of this procedure was not to optimise for and test DeCDF's ability to emulate distributions, but rather to show some example outputs which can arise from the model.

A.3 Figure 2, Right Panel

To generate this figure, 1000 samples were drawn from the following (Mixture of Gaussians) generative process

$$z_i \sim \text{Bernoulli}(p);\ x_i \sim \mathcal{N}(\mu_{z_i}, \sigma_{z_i}^2),$$

with $p = 0.35, \mu_0 = 2, \mu_1 = 4$ and $\sigma_1 = \sigma_2 = 0.4$. The loss in (4) was then optimised for 1000 iterations using Adam (Kingma and Ba, 2014) with a 10^{-3} learning rate to obtain the DeCDF model with the density shown in the right panel of Figure 2. The displayed Gaussian Kernel Density Estimate (KDE) is obtained via the default use of the `scipy.stats.gaussian_kde` function.

B EVALUATION METRIC DETAILS

In Section 5.1.2 of the main paper, we introduce the metrics we use for survival analysis model evaluation. Here we provide the mathematical definitions of the metrics.

B.1 Time-Dependent Concordance Index

If patient i experiences event $k^{(i)}$ at time $s^{(i)}$ and patient j outlives patient i so that $s^{(i)} < s^{(j)}$, the time-dependent concordance represents the probability that the predicted CIF of patient i for event $k^{(i)}$ exceeds the same event's predicted CIF for patient j. The time-dependent concordance for event k, denoted $C_{\text{td}}^{(k)}$, is therefore formally defined via

$$P\left(\hat{F}_k\left(s^{(i)} \mid \mathbf{x}^{(i)}\right) > \hat{F}_k\left(s^{(i)} \mid \mathbf{x}^{(j)}\right) \mid s^{(i)} < s^{(j)}, k^{(i)} = k\right).$$

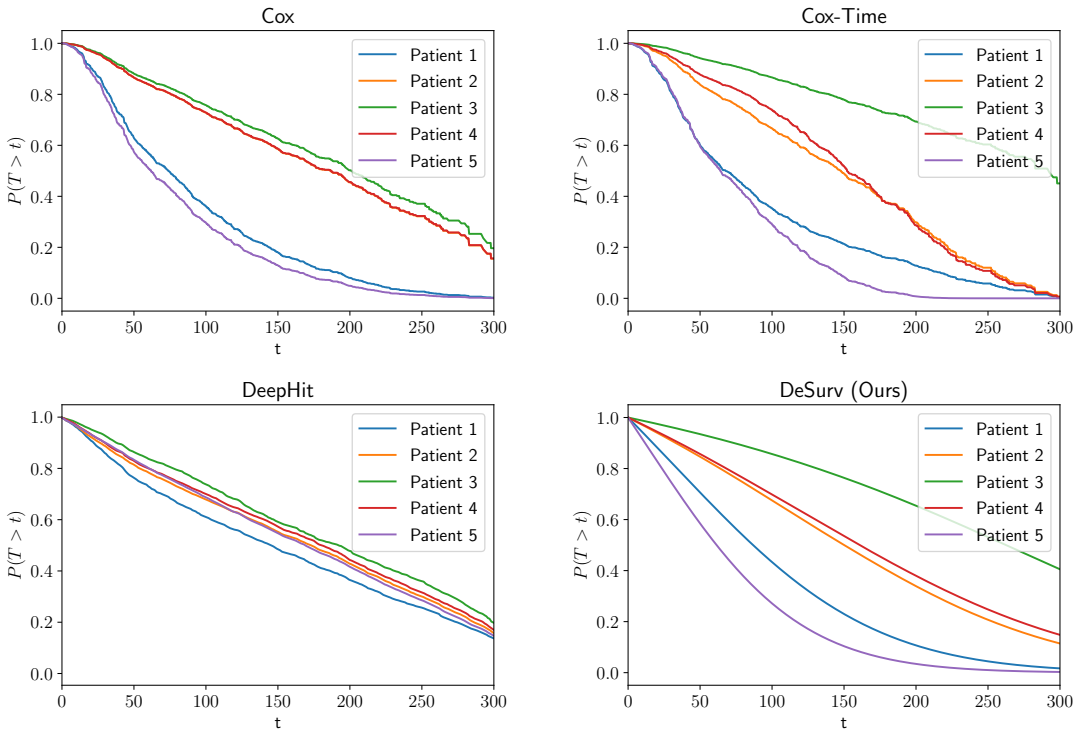

Figure 6: **METABRIC Survival Functions.** 25-year survival trajectories for five random test patients according to four of the considered models. (Appendix D)

In practice, it is empirically estimated via

$$C_{\text{td}}^{(k)} \approx \frac{\sum_{i \neq j} A_{k,i,j} \cdot \mathbb{1}\left(\hat{F}_k\left(s^{(i)} \mid \mathbf{x}^{(i)}\right) > \hat{F}_k\left(s^{(i)} \mid \mathbf{x}^{(j)}\right)\right)}{\sum_{i \neq j} A_{k,i,j}},$$

where $A_{k,i,j} \coloneqq \mathbb{1}\left(k^{(i)} = k, s^{(i)} < s^{(j)}\right)$.

B.2 Integrated Brier Score (IBS)

At time t, let U_i denote the random variable which takes value 1 if $s^{(i)} \leq t$, i.e. if patient i has experienced an event, and 0 if $s^{(i)} > t$, i.e. if patient i is alive at t. The Brier Score (defined in the main paper) associated with this variable over all patients $i = 1, \ldots, N$ is

$$\text{BS}_0(t) = \frac{1}{N} \sum_{i=1}^{N} \left[\left(1 - \hat{F}\left(t \mid \mathbf{x}^{(i)}\right)\right)^2 \mathbb{1}\left\{s^{(i)} \leq t, k^{(i)} = 1\right\} + \hat{F}\left(t \mid \mathbf{x}^{(i)}\right)^2 \mathbb{1}\left\{s^{(i)} > t\right\} \right].$$

In the presence of random censoring the terms are reweighted as

$$\text{BS}(t) = \frac{1}{N} \sum_{i=1}^{N} \left[\frac{\left(1 - \hat{F}\left(t \mid \mathbf{x}^{(i)}\right)\right)^2 \mathbb{1}\left\{s^{(i)} \leq t, k^{(i)} = 1\right\}}{\hat{G}\left(s^{(i)}\right)} + \frac{\hat{F}\left(t \mid \mathbf{x}^{(i)}\right)^2 \mathbb{1}\left\{s^{(i)} > t\right\}}{\hat{G}(t)} \right],$$

where $\hat{G}$ is the Kaplan-Meier estimate of the censoring survival function $P(C > t)$ (see Graf et al. (1999) for details). The IBS is obtained via

$$\text{IBS} = \frac{1}{t_2 - t_1} \int_{t_1}^{t_2} \text{BS}(s)\, \mathrm{d}s,$$

with the calculation performed using `pycox` with 1000 time point evaluations.

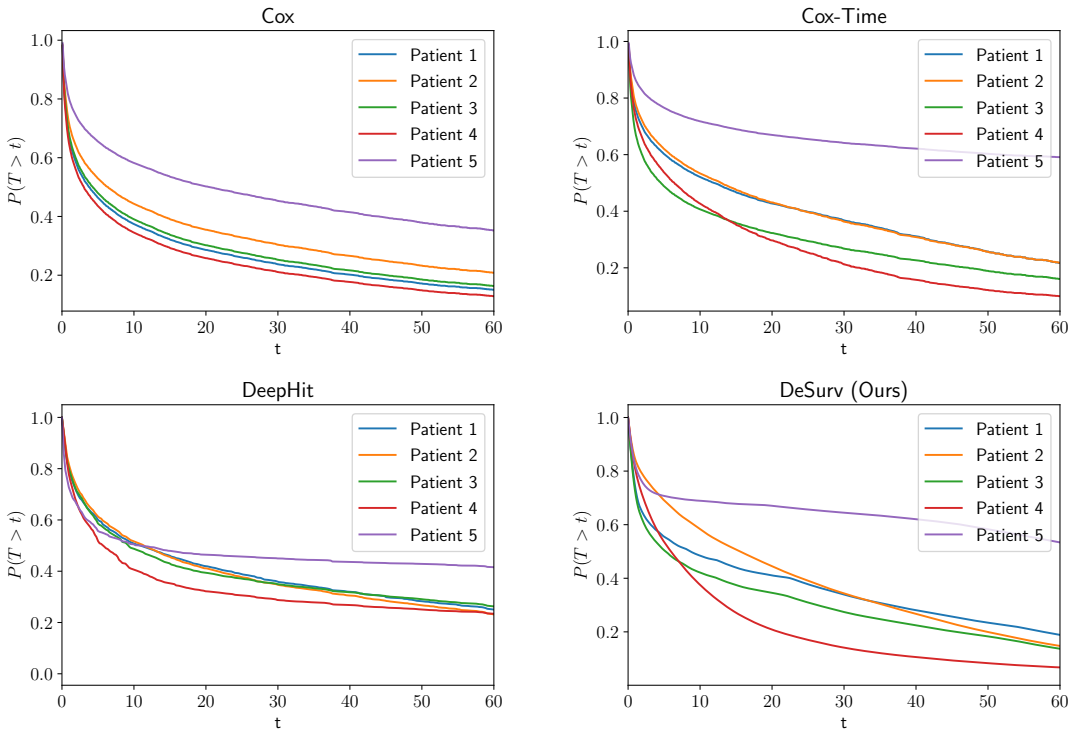

Figure 7: **SUPPORT Survival Functions.** Five-year survival trajectories for five random test patients according to four of the considered models. (Appendix D)

B.3 Integrated Negative Binomial Log-Likelihood (INBLL)

The ($1/N$-scaled) negative log-likelihood for the random variable $U = \{U_i\}_{i=1}^N$ (defined above during the IBS definition) at time t is

$$\text{NBLL}_0(t) = -\frac{1}{N}\sum_{i=1}^{N}\left[\log\left[\hat{F}\left(t \mid \mathbf{x}^{(i)}\right)\right]1\left\{s^{(i)} \leq t, k^{(i)} = 1\right\} + \log\left[1 - \hat{F}\left(t \mid \mathbf{x}^{(i)}\right)\right]\mathbb{1}\left\{s^{(i)} > t\right\}\right].$$

Applying the same censoring weighting as previously provides

$$\text{NBLL}(t) = -\frac{1}{N}\sum_{i=1}^{N}\left[\frac{\log\left[\hat{F}\left(t \mid \mathbf{x}^{(i)}\right)\right]\mathbb{1}\left\{s^{(i)} \leq t, k^{(i)} = 1\right\}}{\hat{G}\left(s^{(i)}\right)} + \frac{\log\left[1 - \hat{F}\left(t \mid \mathbf{x}^{(i)}\right)\right]\mathbb{1}\left\{s^{(i)} > t\right\}}{\hat{G}(t)}\right].$$

This can be integrated as in the IBS case as

$$\text{INBLL} = \frac{1}{t_2 - t_1}\int_{t_1}^{t_2}\text{NBLL}(s)\,\mathrm{d}s$$

to obtain the INBLL metric.

B.4 Mean Absolute Error (MAE)

We define the MAE to be

$$\text{MAE} \coloneqq \frac{\sum_i \left|s^{(i)} - \hat{s}^{(i)}\right| \cdot \mathbb{1}\left(k^{(i)} = 1\right)}{\sum_i \mathbb{1}\left(k^{(i)} = 1\right)},$$

where $s^{(i)}$ and $\hat{s}^{(i)}$ denote the observed event time and mean of the predicted survival distribution respectively for patient i.

B.5 Root Mean Square Error (RMSE)

We define the RMSE to be

$$\text{RMSE} := \sqrt{\frac{\sum_i \left(s^{(i)} - \hat{s}^{(i)}\right)^2 \cdot \mathbb{1}\left(k^{(i)} = 1\right)}{\sum_i \mathbb{1}\left(k^{(i)} = 1\right)}},$$

where $s^{(i)}$ and $\hat{s}^{(i)}$ denote the observed event time and mean of the predicted survival distribution respectively for patient i.

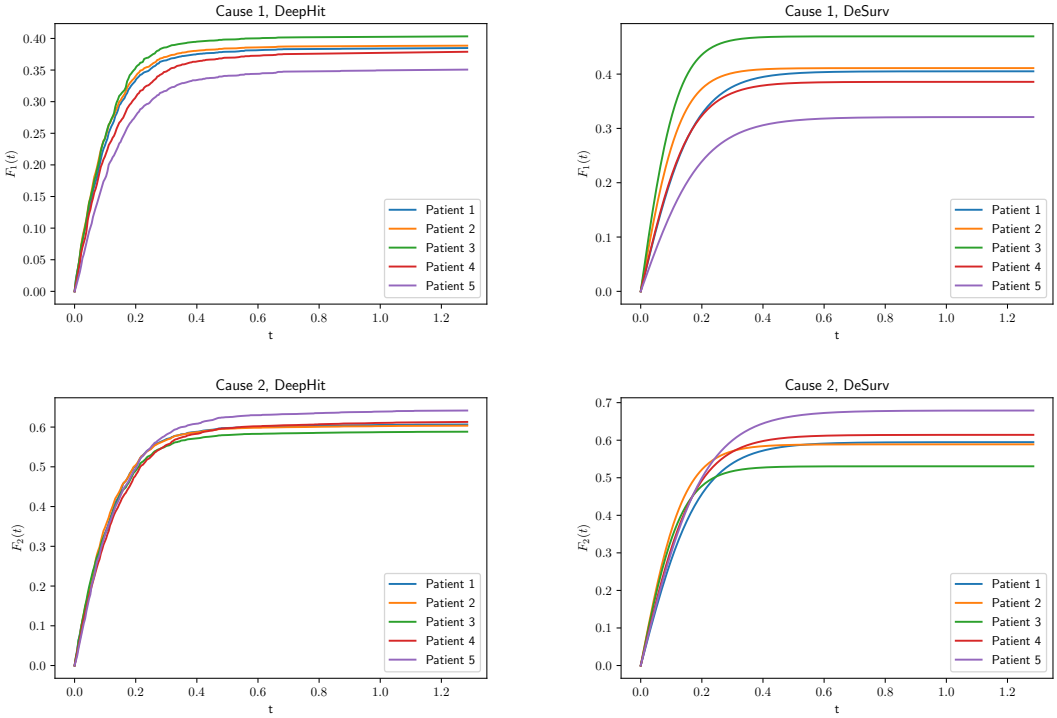

Figure 8: **Synthetic Cumulative Incidence Functions.** Cumulative incidence functions for five random test subjects according to DeepHit and DeSurv. (Appendix D)

C EXPERIMENTAL DETAILS

C.1 Dataset Preprocessing

As noted in the main paper, no additional preprocessing was performed on the METABRIC and SUPPORT datasets beyond that described in Katzman et al. (2018).

C.1.1 SEER

We extract data from the SEER database corresponding to “Female malignant breast cancers". From this data, we extract patients with time-to-event measurements who either i) are alive as of the last measurement, ii) experienced death attributable to their cancer diagnosis, or iii) died due to “Diseases of heart", which we refer to as cardiovascular disease (CVD). We utilise all present informative variables, providing 23 (as noted in other works using (previous verions of) SEER, such as DeepHit (Lee et al., 2018)) variables including age, race, grade, laterality, histology codes, treatment information and tumour size (see e.g. this SEER dictionary for example SEER features). Missing values were mode imputed.

C.2 Model Hyperparameters

All neural networks had two 32-unit hidden layers with ReLU activations. The Adam optimiser (Kingma and Ba, 2014) was used with $\beta_1 = 0.9, \beta_2 = 0.999$, $\alpha \in \{10^{-2}, 10^{-3}\}$ and batch size was 32 in all cases. As a discrete-time method, DeepHit requires a time discretisation scheme. In each case, we discretised the time frame into 300 equal intervals, leading to the output layer of DeepHit having $300K$ output nodes for K competing risks. In the loss

of DeepHit, we set a relative weight of 4:1 between the likelihood and ranking components respectively and set $\sigma = 0.1$ following Lee et al. (2018).

D EXAMPLE INFERRED SURVIVAL FUNCTIONS AND CIFS

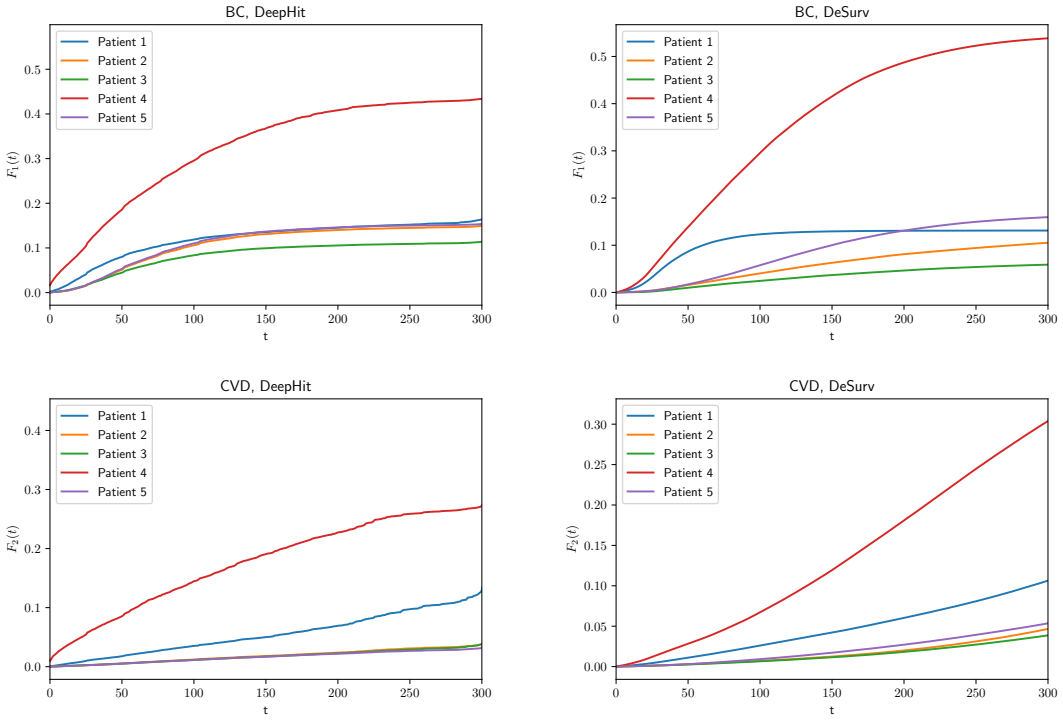

Figure 9: **SEER Cumulative Incidence Functions.** 25-year cumulative incidence functions for five random test patients according to DeepHit and DeSurv. (Appendix D)

To gain insight into model behaviour, we visualise inferred survival functions and CIFs on each dataset.

Figures 6 and 7 show the survival functions of five random test patients in the METABRIC and SUPPORT datasets respectively according to the trained Cox, Cox-Time, DeepHit and DeSurv models. Note in each case that the models generally agree on the relative order of patient risk, however the trajectories are less separated by DeepHit than other models. Note also the similarity between DeSurv's trajectories and those found via Cox-Time, arguably the prevailing modern continuous time single-risk survival analysis approach at the time of writing.

Figures 8 and 9 show the cumulative incidence functions (CIFs) for five random test patients in the synthetic and SEER datasets respectively. As in the single-risk case, the models generally agree on the risk orderings of the patients, but DeSurv demonstrates increased patient-dependent trajectory separation.

E AutoCDF

In Section 2.2 of the main paper, we present DeCDF as the natural way to employ derivative-based methodology to CDF modelling. However, as stated in the discussion, there exist other derivative-based approaches which could be investigated.

One immediate suggestion, which we refer to as AutoCDF due to the autonomous nature of the governing ODE, is to let $g(F)$ be a function with range $\mathbb{R}_{>0}$ (e.g. a neural network with appropriate output activation) and let

$$\frac{\mathrm{d}F}{\mathrm{d}t} = (1 - F)g(F) \quad \text{with} \quad F(0) = 0. \tag{5}$$

The main drawback of this approach compared to DeCDF is that the computation of $F(t = t^*)$ via (5) requires a full ODE solve (not just t-domain quadrature) from $t = 0$ to $t = t^*$. Whilst not infeasible, for $g(F)$ a neural

Table 6: **Effect of Increased Number of Discretisation Points on Synthetic Data DeepHit Performance.**

Method	$\mathbf{C_{td}}^{(1)}$	$\mathbf{C_{td}}^{(2)}$
DeepHit (n=300)	0.611 ± 0.009	0.570 ± 0.004
DeepHit (n=1000)	0.627 ± 0.007	0.583 ± 0.004
DeepHit (n=10000)	0.626 ± 0.009	0.584 ± 0.006
DeSurv (10000)	$\mathbf{0.637 \pm 0.010}$	$\mathbf{0.592 \pm 0.006}$

network ODE solves can introduce implementational and computational complexity. Using adaptive ODE solvers results in a forward operation with an *a priori* unknown computational complexity which is not readily parallelisable. Furthermore, we will require gradients of this forward operation, and whilst these can be estimated using the adjoint method reintroduced by Chen et al. (2018), this approach does not necessarily compute good approximate gradients (Gholami et al., 2019). Moreover, popular open-source implementations do not typically offer both robust neural network functionality and fully autodiff-aware ODE solvers able to reliably calculate gradients in all settings (see e.g. Rackauckas et al. (2018) for method comparisons). The primary framework at the time of writing which attempts this is Julia's SciML initiative (Rackauckas and Nie, 2017) and is under ongoing development. These issues must be considered if modelling via (5) is performed.

F ADDITIONAL SYNTHETIC DATA EXPERIMENT

In Section 5 and Table 4 of the main paper, DeepHit demonstrated limited discriminative performance on the synthetic data. As the synthetic data contains many subjects with small hitting times, one may expect that performance should improve as the number of discretisation points is increased. To investigate this, we repeat the experiment associated with Table 4 with $n = 1000$ and $n = 10000$ discretisation points (Table 6). Here it can be seen that increasing the number of discretisation points improves performance. However, performance appears to saturate for large n, and with 10000 discretisation points DeepHit still exhibits a lower concordance than DeSurv (for comparison we reevaluate DeSurv here using 10000 evaluation points when computing the CIFs for concordance). Note that whilst DeSurv's neural network output dimensions are always K ($= 2$ here), DeepHit's are nK, hence for $n = 10000$ in Table 6, 20000 output dimensions are required.