Uncertainty-Aware Time-to-Event Prediction using Deep Kernel Accelerated Failure Time Models

Zhiliang Wu ZHILIANG.WU@SIEMENS.COM
Ludwig Maximilians University Munich
Siemens AG, Technology, Munich

Yinchong Yang YINCHONG.YANG@SIEMENS.COM
Siemens AG, Technology, Munich

Peter A. Fasching PETER.FASCHING@UK-ERLANGEN.DE
Department of Gynecology and Obstetrics
University Hospital Erlangen, Erlangen

Volker Tresp VOLKER.TRESP@SIEMENS.COM
Ludwig Maximilians University Munich
Siemens AG, Technology, Munich

Abstract

Recurrent neural network based solutions are increasingly being used in the analysis of longitudinal Electronic Health Record data. However, most works focus on prediction accuracy and neglect prediction uncertainty. We propose Deep Kernel Accelerated Failure Time models for the time-to-event prediction task, enabling uncertainty-awareness of the prediction by a pipeline of a recurrent neural network and a sparse Gaussian Process. Furthermore, a deep metric learning based pre-training step is adapted to enhance the proposed model. Our model shows better point estimate performance than recurrent neural network based baselines in experiments on two real-world datasets. More importantly, the predictive variance from our model can be used to quantify the uncertainty estimates of the time-to-event prediction: Our model delivers better performance when it is more confident in its prediction. Compared to related methods, such as Monte Carlo Dropout, our model offers better uncertainty estimates by leveraging an analytical solution and is more computationally efficient.

1. Introduction

Since the introduction of the Electronic Health Record (EHR), an exploding amount of healthcare-related data has been collected in clinics. The physicians often become overwhelmed by data volume and data complexity and may turn to data-driven clinical decision support systems (Halpern et al., 2016; Tresp et al., 2016; Xiao et al., 2018). These solutions often provide decision support in two ways. A *prescriptive* system generates action recommendations, such as medications and therapy plans, while a *predictive* system provides physicians with a prediction of the outcome, given a decision. Such outcome could be, for instance, the adverse events related to a specific therapy or the progression-free-survival time after treatment. These predictions are often based on modeling the three-way interaction between the outcome, the patients' status, and clinical decisions recorded in historical

data. In this work, we address the prediction of treatment outcome and propose a new class of uncertainty-aware models that can communicate uncertainty to physicians. We argue that such uncertainty estimates add transparency and trustworthiness to the clinical decision support systems and encourage their application on even larger scales.

Due to the high-dimensional, sparse, and sequential nature of EHR data, simpler, more transparent white-box models often fail to capture the complex interactions between the target variable and input features. Meanwhile, recurrent neural network (RNN)-based solutions have proven capable of addressing the longitudinal aspect in EHR data (Esteban et al., 2016; Choi et al., 2017; Yang et al., 2017b; Purushotham et al., 2018; Wu et al., 2020). These models apply RNNs to aggregate historical observations to produce an individual and time-dependent representation of a patient. The last layer is then a linear map from patient representation to the target variable representing, e.g., the predicted outcome for the patient. The advantage of an RNN is twofold. First, it can handle records that vary in length from patient to patient, as in the analysis of texts with varying lengths (Mikolov et al., 2012). Second, the more advanced RNN variants such as long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) and gated recurrent unit (GRU) (Chung et al., 2014) are flexible in memorizing both long-term and short-term input features. Despite the state-of-the-art predictive performance of the neural network (NN)-based methods, these methods fail to provide reasonable uncertainty estimates of the predictions (Nguyen et al., 2015). It is easier to address this issue in classification tasks, as the predicted probability can be interpreted to reflect uncertainty, which leads to various calibration approaches, like temperature scaling (Guo et al., 2017). However, for regression tasks—like the time-to-event prediction task in our case—one has to provide a predictive distribution to address the uncertainty estimates, which is rarely considered in vanilla NN-based solutions. In healthcare applications, we would argue that uncertainty-awareness of the model is as important as point estimate performance, since the uncertainty estimates would assist the physicians in better interpreting the results from a black-box model (Begoli et al., 2019). If the model provides a high uncertainty estimate for its prediction, the physicians would be more careful about it and take a closer look at that case.

It is, however, not a trivial task to augment NNs with reliable uncertainty estimates. Currently, popular choices to do so include MC Dropout methods (Gal and Ghahramani, 2016), Bayesian neural networks (Bishop, 2006) and deep ensembles (Lakshminarayanan et al., 2017). Many variants of them involve repeated sampling procedures either during training or during inference, which could become computationally expensive for large NNs. In this work, we investigate the possibility of quantifying the predictive uncertainty by applying Gaussian Processes (GPs). As a popular class of machine learning methods, GP produces a predictive distribution instead of a single point estimate for each test sample. For small datasets, GPs have proven to be flexible and data-efficient. However, GPs' inclusion of a large training dataset inevitably introduces large storage and computational complexity (Rasmussen and Williams, 2005). With the recent advance of sparse GP techniques, the computational complexity has been largely reduced (Liu et al., 2020). Motivated by the desirable predictive distribution of GPs and the progress of sparse GPs, we propose in this work a novel approach by integrating RNNs as feature extractors into GP-based predictive models. Furthermore, we propose a deep metric learning (DML)-based pre-training method to further improve the performance of the model.

In the context of time-to-event prediction, our proposed model is closely related to the Accelerated Failure Time (AFT) models (Prentice, 1978; Kalbfleisch and Prentice, 2002). As one of the well-known models in survival analysis, the AFT models have shown promising results, especially when, in an application, the direct prediction of survival time is more important than hazard estimation. With our proposed method, we have augmented the uncertainty estimates in the AFT models.

Generalizable Insights about Machine Learning in the Context of Healthcare

Neural networks offer powerful modeling ability to learn from EHR data, but often neglect the uncertainty estimates in the predictions. Leveraging state-of-the-art sparse GPs, we propose a method to integrate the uncertainty into the NN-based solutions for time-to-event prediction tasks. Experiments on two real-world datasets show that the resulting model can 1) scale very well to large datasets; 2) deliver improved performance regarding point estimates; 3) offer reasonable predictive variances, which reflect the confidence of the predictions and enhance the calibration of the model. The uncertainty estimates in our model can help establish a trustworthy relationship with physicians since it expresses higher confidence in more accurate predictions and vice versa.

2. Related Work

Predictive Modeling with EHRs To better capture the time-dependent information of patients, many works have been proposed to model longitudinal EHR data, ranging from earlier statistical methods like *landmarking* (Van Houwelingen, 2007), *Joint Models* (Rizopoulos, 2011; Hickey et al., 2016), to more recent methods like *Bayesian Nonparametric Dynamic Survival* (Bellot and Schaar, 2020). Meanwhile, RNN-based approaches have proved to be very successful both for discrete medical events and continuous time-series data. Esteban et al. (2016) applied sequence-to-sequence RNN models to predict discrete medical events of patients suffering from kidney failure. Yang et al. (2017b) proposed many-to-one RNN models to deal with discrete medical events and predict the therapy decision for breast cancer. At the same time, Choi et al. (2017) applied models of similar structure for the early detection of heart failure onset. For continuous time-series data, multiple readings of individual signals are usually aggregated to reduce the high-resolution patient data so that they can be better consumed by the neural networks, e.g., heart rate and blood pressure in ICU time-series data (Johnson et al., 2016). With the aggregation method on the ICU time series data, Purushotham et al. (2018) provided RNN-based benchmarks for the mortality prediction, length-of-stay prediction, and ICD-9 code group prediction. Following similar data pre-processing steps, Wu et al. (2020) presented RNN-based models to learn the optimal treatment strategies for administering intravenous fluids and vasopressors. In this work, we include EHR data with discrete medical events as well as the dataset with continuous time-series measurements for the time-to-event prediction tasks to validate our proposed model.

Survival Analysis with Neural Networks and Gaussian Processes As most popular approaches in survival analysis are based on generalized linear models, they have been extended to nonlinear models, including NNs and GPs. Saul (2016) proposed chained GPs

to model multiple parameters of the log-logistic likelihood in AFT models through the latent functions of GPs. In addition, many GP-based methods are proposed to enhance the Cox proportional hazards (CPH) model, including the Bayesian semi-parametric model (Fernández et al., 2016) and deep multi-task Gaussian process DMGP (Alaa and van der Schaar, 2017). For NN-based approaches, Yang et al. (2017a) combines tensorized RNN model with the AFT model to predict progression-free survival (PFS) time. Kvamme et al. (2019) proposes an extension of CPH models, *CoxTime*, by parametrizing the relative risk function with NNs as well as modeling interactions between covariates and time. However, most of these works focus on capturing the non-linearity between covariates to improve the performance of point estimates. The uncertainty perspective of the prediction is rarely addressed. More recently, Chen (2020) proposed Deep Kernel Survival Analysis to learn kernel functions for the conditional Kaplan-Meier estimator and enables subject-specific survival time prediction intervals. The uncertainty is quantified by the prediction intervals. With an emphasis on uncertainty-awareness, we explore in this work the time-to-event prediction tasks with the AFT models using a combination of RNNs and GPs.

Exact Gaussian Process and Scalable Variational Gaussian Process with Neural Networks Since the capacity of GPs grows with available training data, many works have proposed possible solutions for both exact GP and sparse GP. Based on the efficient GP inference using Blackbox Matrix-Matrix multiplication from Gardner et al. (2018), Wang et al. (2019) realized exact GP training on over a million training samples by taking advantage of multi-GPU parallelization. Meanwhile, various works have been proposed to approximate the original GPs to save computational cost, including some early efforts, such as the Bayesian committee machine (BCM) (Tresp, 2000), the Nyström methods (Williams and Seeger, 2001), the Fully Independent Training Conditional (FITC) Approximation (Snelson and Ghahramani, 2006), Variational Free Energy (VFE) (Titsias, 2009), the more recent Scalable Variational Gaussian Process (SVGP) (Hensman et al., 2013) and Parametric Predictive Gaussian Process (PPGP) Regressor (Jankowiak et al., 2020) (more details see Sec. 3.2). In addition, the idea of combining sparse GPs with neural networks also received much attention, where the Deep Kernel Learning (DKL) (Wilson et al., 2016) and GP hybrid deep networks (GPDNN) (Bradshaw et al., 2017) are the ones most related to our work.

3. Methods

This section first discusses the RNN-based feature extractors to learn representations from the patients' static and sequential information. The resulting latent representations are used as inputs for the time-to-event prediction task. We will elaborate on our proposed model, which combines the GP-based models with AFT models. Afterward, a DML-based supervised pre-training method is presented to enhance the performance of the proposed model.

3.1. Recurrent Neural Networks as Feature Extractors

EHR data typically consist of *static features* and *sequential features*, both of which are important for the time-to-event prediction tasks. We regard the background information of each patient as static features $\boldsymbol{x}_i^{\text{sta}} \in \mathbb{R}^{\text{n}_{\text{sta}}}$. We use i, n_{sta} to denote the patient sample

index and the number of static features, respectively. In addition, features observed at all time-steps constitute the sequential feature matrix $\boldsymbol{X}_i^{\text{seq}} = [\boldsymbol{x}_i^0, \boldsymbol{x}_i^1, \ldots, \boldsymbol{x}_i^{t_i}]^\top \in \mathbb{R}^{t_i \times \text{n}_{\text{seq}}}$, where t_i is the number of observed time-steps for the i-th patient sample and n_{seq} denotes the number of sequential features. Due to the high sparsity or redundancy in raw features spaces, it has been shown to be beneficial to first apply a (non-linear) embedding layer on the raw features to learn the static hidden representation $\boldsymbol{h}_i^{\text{sta}}$ and sequential feature embeddings $\boldsymbol{X}_i^{\text{seq\_emb}}$ (Esteban et al., 2016). Formally, we have

$$\boldsymbol{h}_i^{\text{sta}} = g_1(\boldsymbol{A}\boldsymbol{x}_i^{\text{sta}}) \in \mathbb{R}^{\text{n}_{\text{sta\_repr}}}$$
$$\boldsymbol{X}_i^{\text{seq\_emb}} = g_2(\boldsymbol{X}_i^{\text{seq}}\boldsymbol{B}) \in \mathbb{R}^{t_i \times \text{n}_{\text{seq\_emb}}}$$

where $\boldsymbol{A} \in \mathbb{R}^{\text{n}_{\text{sta\_repr}} \times \text{n}_{\text{sta}}}, \boldsymbol{B} \in \mathbb{R}^{\text{n}_{\text{seq}} \times \text{n}_{\text{seq\_emb}}}$ are embedding matrices, $g_1(\cdot), g_2(\cdot)$ are activation functions like $\tanh(\cdot)$, $\text{n}_{\text{sta\_repr}}, \text{n}_{\text{seq\_emb}}$ denote the dimension of the static hidden representations and the sequential feature embeddings, respectively. Afterward, more advanced variants of RNNs, LSTM or GRU, are used to encode the sequential feature embeddings $\boldsymbol{X}_i^{\text{seq\_emb}}$ into sequential latent representations $\boldsymbol{h}_i^{\text{seq}}$. Since we are mainly interested in modeling the time-to-event, only the last hidden states from LSTM/GRU are involved as inputs in the down-streaming tasks. Formally, we have

$$\boldsymbol{h}_i^{\text{seq}} = \text{RNN}(\boldsymbol{X}_i^{\text{seq\_emb}}) \in \mathbb{R}^{\text{n}_{\text{seq\_repr}}},$$

where $\text{n}_{\text{seq\_repr}}$ is the dimension of sequential hidden representations and $\text{RNN}(\cdot)$ could be an LSTM or GRU.

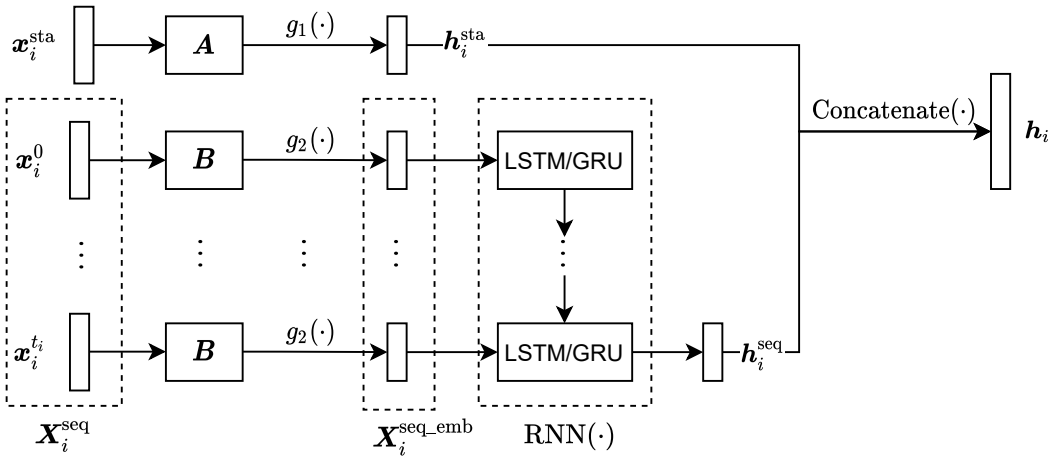

Figure 1: Illustration of the RNN-based feature extractor: An (non-linear) embedding layer is first involved in learning the sequential feature embeddings, which are then fed into RNN-based models to encode the sequential hidden representations. These are then concatenated with the static hidden representations. The complete hidden representation is expected to encode all relevant patient information and serves as abstract covariates for the down-streaming time-to-event prediction task.

The complete structure of the feature extractor is shown in Fig. 1. Similar to the encoder part in Cho et al. (2014), this procedure is especially appealing for the patients with different observed time-steps, as it is theoretically capable of storing all relevant information in the medical events with variable lengths but remains a consistent form of representations.

3.2. Scalable Variational Gaussian Processes for Time-to-Event Prediction

From the last section, we have learned the static hidden representation $\boldsymbol{h}_i^{\text{sta}}$ and the sequential hidden representation $\boldsymbol{h}_i^{\text{seq}}$ through RNN-based feature extractors. By concatenating them, we get the complete hidden representation as

$$\boldsymbol{h}_i = [\boldsymbol{h}_i^{\text{sta}}; \boldsymbol{h}_i^{\text{seq}}] \in \mathbb{R}^{\text{n}_{\text{sta\_repr}}+\text{n}_{\text{seq\_repr}}},$$

which can be viewed as abstract covariates of patients in a latent feature space.

The class of Accelerated Failure Time (AFT) models is a general class of models, where the covariates of the patients are assumed to act multiplicatively on the time-scale (Collett, 2015). Compared to the semi-parametric Cox proportional hazards (CPH) models, the AFT models take advantage of their parametric nature and include a wider range of survival time distributions. Formally, with the time-to-event target variable z_i, the AFT models predict its logarithm as $\log z_i := y_i = \boldsymbol{\beta}^\top \boldsymbol{h}_i + \epsilon_i$, where $\boldsymbol{\beta}^\top \boldsymbol{h}_i$ is the linear predictor with the (trainable) parameter vector $\boldsymbol{\beta}$, $\epsilon_i \overset{i.i.d.}{\sim} \mathcal{D}_\epsilon$ denotes the error term which is specified by a particular probability distribution $\mathcal{D}_\epsilon$. Common choices for $\mathcal{D}_\epsilon$ include Normal, Weibull and Logistic distributions, which correspondingly specifies the target variable z_i to be log-normal, log-weibull and log-logistic distributed, respectively. In this work, we assume our target variable of interest to follow a log-normal distribution. In other words, we have correspondingly $\epsilon_i \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma_{\text{obs}}^2)$. Furthermore, we propose to replace the linear predictor $\boldsymbol{\beta}^\top \boldsymbol{h}_i$ with Gaussian Process posterior prediction to enable uncertainty-aware predictions. In the following, we shall introduce this approach in detail.

As shown in Fig. 2, after we obtain the hidden representations $\boldsymbol{h}_i$ from the feature extractor $f_{\boldsymbol{\Phi}}(\cdot)$ (details see Sec. 3.1), these are used as abstract patient covariates for the

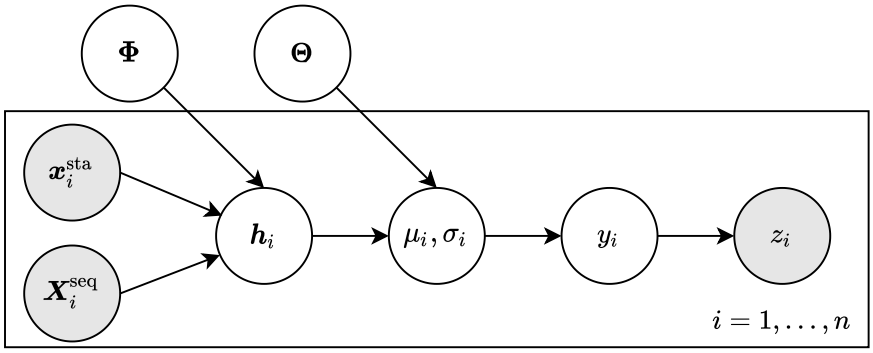

Figure 2: Graphical model of Deep Kernel Accelerated Failure Time models in plate notation. Nodes represent variables, where shaded ones are observable and non-shaded ones are latent. Plates indicate the repetition of the subgraph.

subsequent GP-based models $g_{\boldsymbol{\Theta}}(\cdot)$ to generate predictive distribution $y_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$, the logarithm of the target variable. We denote $\boldsymbol{\Phi}$ and $\boldsymbol{\Theta}$ as the trainable parameters in the RNN-based feature extractor and the GP-based predictive model, respectively. Since we take advantage of neural networks with GP-based models as an advanced version of AFT models, we name it Deep Kernel Accelerated Failure Time (DKAFT) models. In the following, we will discuss different GP-based models in our proposed method.

In regression, y_i is a noisy observation of the GP function value $f_i = f(\boldsymbol{h}_i)$, which is assumed to behave a priori according to

$$p(\boldsymbol{f}|\boldsymbol{h}_1, \ldots, \boldsymbol{h}_n) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{K}),$$

where $\boldsymbol{f} = [f_1, \ldots, f_n]^\top \in \mathbb{R}^n$ is a vector of GP function values and $\boldsymbol{K} \in \mathbb{R}^{n \times n}$ is a covariance matrix, whose entries are given by the covariance function $k_{ij} = k(\boldsymbol{h}_i, \boldsymbol{h}_j)$. The choice of the covariance function reflects the prior knowledge of the generative process of the model, where a Radial Basis Function (RBF) kernel is commonly used. There are some important hyper-parameters in the covariance function, e.g., the length-scale and signal variance in the RBF kernel, which can be learned through maximizing the log marginal likelihood defined as

$$\mathcal{L}_{\text{ExactGP}} = -\frac{1}{2}\boldsymbol{y}^\top \left(\boldsymbol{K} + \sigma_{\text{obs}}^2 I\right)^{-1} \boldsymbol{y} - \frac{1}{2}\log\left|\boldsymbol{K} + \sigma_{\text{obs}}^2 I\right| - \frac{n}{2}\log 2\pi. \tag{1}$$

With the optimized parameters, the prediction of a test sample f_* can be understood as computing the conditional probability of the test location given all values in the training dataset. Formally, the GP model outputs a predictive distribution as

$$f_* \sim \mathcal{N}(\boldsymbol{k}_*^\top \left(\boldsymbol{K} + \sigma_{\text{obs}}^2 \boldsymbol{I}\right)^{-1} \boldsymbol{y}, k_{**} - \boldsymbol{k}_*^\top \left(\boldsymbol{K} + \sigma_{\text{obs}}^2 \boldsymbol{I}\right)^{-1} \boldsymbol{k}_*), \tag{2}$$

where $\boldsymbol{k}_* = [k(\boldsymbol{h}_1, \boldsymbol{h}_*), \ldots, k(\boldsymbol{h}_n, \boldsymbol{h}_*)]^\top \in \mathbb{R}^n$ denotes the covariance function values between the training inputs and the test input $\boldsymbol{h}_*$. Please note that, in contrast to the original AFT formulation, the covariates $\boldsymbol{h}_i$ do not influence the logarithm of the target variable directly in GP. Instead, the accelerating effect is realized via the covariance function.

Equation 1 and Equation 2 reveal the training and inference step for our proposed DKAFT model with an *Exact GP output layer*. However, the Exact GP cannot scale well to a large-scale dataset due to the $\mathcal{O}(n^3)$ computational complexity from the inverse operations of the large covariance matrix $\boldsymbol{K}$.

A tremendous amount of work has been proposed to address the scalability issue in the Exact GP, where the techniques of inducing points with variational inference have found most interest (Quinonero-Candela and Rasmussen, 2005). In short, the inducing points constitute a "summary" dataset, which is learned to generalize the original dataset to reduce the $\mathcal{O}(n^3)$ computational complexity. They consist of inducing inputs $\{\boldsymbol{u}_i\}_{i=1}^m =: \boldsymbol{U}$ (corresponding to $\{\boldsymbol{h}_i\}_{i=1}^n$) and inducing variables $\{v_i\}_{i=1}^m =: \boldsymbol{v}$ (corresponding to $\{f_i\}_{i=1}^n$), where $m \ll n$. In the context of our DKAFT model, the inducing inputs refer to a summary of the abstract patient covariates in the latent space. The learning of the inducing points is facilitated by variational methods under different approximation assumptions, e.g., the prior approximation and posterior approximation (Liu et al., 2020).

Among various GP approximations, the Scalable Variational Gaussian Process (SVGP) proposed by Hensman et al. (2013) reduces the computational complexity to $\mathcal{O}(m^3)$ and

makes the training amenable to stochastic gradient descent (SGD)-based methods. More concretely, the variational distribution $q(\boldsymbol{v})$ of the inducing variables is assumed to follow a multivariate Normal distribution $\mathcal{N}(\boldsymbol{m}, \boldsymbol{S})$ in SVGP. Following the notation in Jankowiak et al. (2020), instead of the log marginal likelihood objective in an Exact GP, we optimize the parameters by maximizing the Evidence Lower BOund (ELBO)

$$\mathcal{L}_{\text{SVGP}} = \sum_{i=1}^{n} \left\{ \log \mathcal{N}\left(y_i \mid \mu_{\boldsymbol{f}}\left(\boldsymbol{h}_i\right), \sigma^2_{\text{obs}}\right) - \frac{\sigma^2_{\boldsymbol{f}}\left(\boldsymbol{h}_i\right)}{2\sigma^2_{\text{obs}}} \right\} - \text{KL}(q(\boldsymbol{v}) \| p(\boldsymbol{v})) \tag{3}$$

and the predictive distribution for each sample is

$$f_i \sim \mathcal{N}(\mu_{\boldsymbol{f}}(\boldsymbol{h}_i), \sigma^2_{\boldsymbol{f}}(\boldsymbol{h}_i)) = \mathcal{N}(\boldsymbol{k}_i^\top \boldsymbol{K}_{\boldsymbol{vv}}^{-1} \boldsymbol{m}, k_{ii} - \boldsymbol{k}_i^\top \boldsymbol{K}_{\boldsymbol{vv}}^{-1} \boldsymbol{k}_i + \boldsymbol{k}_i^\top \boldsymbol{K}_{\boldsymbol{vv}}^{-1} \boldsymbol{S} \boldsymbol{K}_{\boldsymbol{vv}}^{-1} \boldsymbol{k}_i). \tag{4}$$

$\boldsymbol{K}_{\boldsymbol{vv}} \in \mathbb{R}^{m \times m}$ is the covariance matrix of the inducing variables, whose entries are computed based on inducing inputs as $k_{ij}^{\boldsymbol{vv}} = k(\boldsymbol{u}_i, \boldsymbol{u}_j)$, $\boldsymbol{k}_i = [k(\boldsymbol{u}_1, \boldsymbol{h}_i), \ldots, k(\boldsymbol{u}_m, \boldsymbol{h}_i)]^\top \in \mathbb{R}^m$ is the covariance function values between all inducing inputs with the sample input $\boldsymbol{h}_i$, and $\text{KL}(\cdot\|\cdot)$ denotes the Kullback–Leibler divergence between two distributions.

Both training and inference of SVGP in Equation 3 and Equation 4 are more computationally tractable, since they only involve the inducing points instead of the whole dataset as in Exact GP. We can therefore take advantage of this formulation for large-scale datasets. With $\mathcal{L}_{\text{SVGP}}$ as an objective, we have our DKAFT model with an *SVGP output layer*.

More recently, Jankowiak et al. (2020) found that the predictive uncertainty from SVGP is dominated by the input-independent observational noise σ^2_{obs}, whereas it is indeed the input-dependent function variance $\sigma^2_{\boldsymbol{f}}(\boldsymbol{h}_i)$ that makes the GP posteriors attractive. Different from the SVGP objective in Equation 3, the Parametric Predictive Gaussian Process (PPGP) Regressor takes advantage of the predictive distribution in Equation 4 and embeds it directly in the objective using Maximum Likelihood Estimation (MLE) methods. Formally, the objective in PPGP is defined as

$$\mathcal{L}_{\text{PPGP}} = \sum_{i=1}^{n} \log \mathcal{N}\left(y_i \mid \mu_{\boldsymbol{f}}\left(\boldsymbol{h}_i\right), \sigma^2_{\text{obs}} + \sigma^2_{\boldsymbol{f}}\left(\boldsymbol{h}_i\right)\right) - \text{KL}(q(\boldsymbol{v}) \| p(\boldsymbol{v})). \tag{5}$$

With $\mathcal{L}_{\text{PPGP}}$ as a training objective, we have our DKAFT model with *an PPGP output layer*.

The objectives introduced above are defined for samples with observed time-to-event. For right-censored cases, we can take advantage of the parametric predictive distribution in Equation 4 to compute the survival function, whose logarithm contributes to the final objective together with the ELBO objective. Such an optimization objective is also used in AFT models, where the non-censored cases contribute to the objective through their respective probability distribution function and the censored ones through the survival function (see Collett, 2015, Equation 5.9). Formally, the survival function of the log-normal distribution in our DKAFT model is

$$S(z|\boldsymbol{h_i}) = 1 - \Phi\left(\frac{\log z - \mu_{\boldsymbol{f}}(\boldsymbol{h}_i)}{\sigma_{\boldsymbol{f}}(\boldsymbol{h}_i) + \sigma_{\text{obs}}}\right),$$

where $\Phi(\cdot)$ is the cumulative distribution function of a standard normal distribution.

3.3. Deep Metric Learning as Supervised Pre-training

The proposed architecture with RNNs as feature encoders and sparse GPs as predictive models is trainable in an end-to-end fashion with gradient descent. The free parameters include parameters in RNNs, the inducing points, and hyper-parameters in the covariance function. In our experiment, we realize that training such architecture from scratch could be challenging. Given an RNN and inducing points that are both randomly initialized, the covariance matrix in GP is also random. This often causes the length scale parameter in the RBF kernel to shrink to extremely small values, and the GP would then degrade to its prior, correspondingly. To alleviate such problems, we find the initialization of the inducing points to be an important step for obtaining good models. More specifically, we find that the training always fails if we initialize the inducing inputs with random vectors. On the other hand, initializing the inducing points with latent representations from the RNN-based feature extractor always shows good performance, even though the parameters in the feature extractor are initialized randomly. Formally, we get the initial inducing inputs as

$$\boldsymbol{u}_i^{\text{init}} = \boldsymbol{h}_i^{\text{init}} = f_{\boldsymbol{\Phi}_{\text{init}}}(\boldsymbol{x}_i^{\text{sta}}, \boldsymbol{X}_i^{\text{seq}}),$$

where a random subset of the training inputs $\{\boldsymbol{x}_i^{\text{sta}}, \boldsymbol{X}_i^{\text{seq}}\}_{i=1}^m$ is involved. To this end, we conjecture that a pre-training step on the feature extractor would boost the performance of our DKAFT model. Since many covariance functions, e.g., RBF kernels, take the distance between samples as input, it would be beneficial if the feature extractor generates abstract covariates in well-clustered latent spaces, where the samples with similar target variables are closer to each other. We propose that one could achieve such a beneficial configuration via Deep Metric Learning (DML), which is initially proposed for vision-related tasks like face verification (Schroff et al., 2015) and person re-identification (Hermans et al., 2017). What DML learns is to represent samples in a latent space that retains the similarity in the target variables.

In DML, pair loss or triplet loss provides the foundation for embedding samples using twin networks, which refers to the replications of the same feature extractor network. Various losses have been proposed to improve the embedding from different perspectives, including contrastive loss (Hadsell et al., 2006), triplet margin loss (Weinberger et al., 2006) or the more recent Signal-To-Noise Ratio loss (Yuan et al., 2019). More specifically, a triplet is defined with the class information to consist of an anchor, a positive, and a negative sample, $\{\boldsymbol{x}_i^{\text{sta}}, \boldsymbol{X}_i^{\text{seq}}\}^{\text{A/P/N}}$, where the anchor is of the same class as the positive and the negative is not. As there are no class labels in time-to-event prediction, we propose to categorize the target variables according to their binnings in the histogram to facilitate the triplet generation. In the context of our DKAFT model, we train the RNN-based feature extractor using, e.g., triplet margin loss (Schroff et al. (2015))

$$\mathcal{J}_{\text{triplet}} = \sum_{i=1}^{n} \left[\mathrm{d}(\boldsymbol{h}_i^{\text{A}}, \boldsymbol{h}_i^{\text{P}}) - \mathrm{d}(\boldsymbol{h}_i^{\text{A}}, \boldsymbol{h}_i^{\text{N}}) + \alpha\right]_+,$$

where $\mathrm{d}(\cdot, \cdot)$ is a distance metric, like Euclidean distance, $\boldsymbol{h}_i^{\text{A}}, \boldsymbol{h}_i^{\text{P}}, \boldsymbol{h}_i^{\text{N}}$ are the abstract covariates of anchor, positive, and negative samples, α is a predefined margin value, and $[\cdot]_+$ takes the positive part of the variable. From the GP perspective, it is the covariance function

that defines the "similarity" between samples, the choice of a specific loss in DML should therefore take it into consideration.

To find a suitable training epoch for the pre-training, we use an early stopping technique, which terminates the training automatically if the monitored metric does not improve over a given number of epochs. Mean Average Precision at R (MAP@R) proposed in Musgrave et al. (2020) is used as the monitored metric on the validation set, which combines the metrics of Mean Average Precision and R-precision.

To conclude, we propose to apply an RNN-based feature extractor to learn fix-sized latent representations from patient trajectories of variable lengths. The feature extractor can be randomly initialized or pre-trained with our proposed DML-based approach. The DKAFT model is trained end-to-end against (sparse) GP objectives using SGD-based methods and produces predictive distributions.

4. Cohort

4.1. Data Extraction

We have included two datasets to validate the effectiveness of our proposed method. In both datasets, we treat observed time-to-event as our target variables.

The first dataset is provided by the PRAEGNANT study network (Fasching et al., 2015), which focuses on patients suffering from metastatic and incurable breast cancer. Based on a patient's background information and medical history, we attempt to predict the Progression-Free Survival time (PFS-PRAEGNANT), i.e., the number of days till the next recorded progression. The raw data are hosted in a relational database system, secuTrial®, and can be accessed under restrictions. After querying and preprocessing, we retrieved a dataset of 1336 patient cases.

The second dataset comes from the Medical Information Mart for Intensive Care database (MIMIC-III), a freely accessible database, which contains data including $53,423$ distinct Intensive Care Unit (ICU) admissions of adult patients between 2001 and 2012 (Johnson et al., 2016). In this work, we consider a cohort of patients from MIMIC-III v1.4, who are older than 15 years at the time of ICU admission. Besides, only the first admission of these patients is included to prevent potential information leakage in the analysis. Based on the data collected during the first 48 hours, we attempt to predict the length-of-stay (LoS-MIMIC) for each admission, i.e., the number of days between hospital admission and discharge from the hospital. More specifically, we followed the scripts[1] provided by Purushotham et al. (2018) and extracted a dataset with $31,986$ patient admissions.

In both extracted datasets, all cases are with observed time-to-event. In case of the MIMIC dataset, the patients were always supposed to leave the ICU and the PRAEGNANT patients all have metastasis and were expecting multiple progressions.

4.2. Feature Processing

In the PRAEGNANT dataset, the static information includes 1) basic patient information, e.g., age and height 2) information on the primary tumor, and 3) history of metastasis before entering the study. In total, there are 26 features of binary, categorical, or numerical

[1] https://github.com/USC-Melady/Benchmarking_DL_MIMICIII

type. After performing one-hot encoding on the binary and categorical features, we obtain a feature vector $\boldsymbol{x}_i^{\text{sta}} \in \mathbb{R}^{114}$ for the static information with an average sparsity of 0.871. The sequential information includes 4) local recurrence 5) metastasis 6) clinical visits 7) radio-therapies 8) systemic therapies, and 9) surgeries. These events are observed with a timestamp, and multiple events could happen at the same timestamp. After performing binary-encoding on 26 sequential features, we extract a feature matrix $\boldsymbol{X}_i^{\text{seq}} \in \mathbb{R}^{t_i \times 188}$ for the sequential information of each patient case, where t_i denotes the length of the sequence before the progression. The length of the sequences t_i varies from 1 to 22 and is on average 6.42. The average sparsity of the sequential feature matrix is 0.973.

In the MIMIC-III dataset, the static information refers to the basic information during the admission, e.g., age and admission type. After performing binary encoding on five static features, we obtain a feature vector for each admission $\boldsymbol{x}_i^{\text{sta}} \in \mathbb{R}^{10}$. Moreover, the sequential information refers to the continuously monitored measurements or prescriptions in the ICU environment. They are sampled or aggregated every one hour to represent the patient status at different time steps. 136 sequential features have been selected. Those features are available for most patients. As a result, we extract a feature matrix $\boldsymbol{X}_i^{\text{seq}} \in \mathbb{R}^{t_i \times 136}$ for the sequential information, where t_i is 48 for all patient admissions. A complete list of the chosen features can be found in Appendix A.

5. Experiments

5.1. Experimental Details and Evaluation Approaches

We conducted cross-validations (CV) for the PFS-PRAEGNANT and LoS-MIMIC prediction tasks with 90% samples in the dataset. The validation set is used for tuning hyper-parameters like, e.g., the dimension of the latent representations, weight decay, and training epochs. As a result, we have $4, 32, 128$ and $4, 64, 64$ for the size of the static latent representations $\text{n}_{\text{sta\_repr}}$, sequential feature embeddings $\text{n}_{\text{seq\_emb}}$, and sequential latent representations $\text{n}_{\text{seq\_repr}}$ in the PFS-PRAEGNANT and LoS-MIMIC prediction tasks, respectively. The evaluation metrics reported in the following are all computed based on the remaining unseen 10% samples of the dataset.

All our NN-based models are built with the PyTorch package (Paszke et al., 2019). The GP-related methods are implemented with the help of the GPyTorch package (Gardner et al., 2018). Related scripts[2] are published to ensure the reproducibility of the work.

Since the target variable z_i is assumed to follow a log-normal distribution, it is not appropriate to measure the results using Root Mean Square Error (RMSE) in its original scale. Therefore, we report the more robust metric of Median Absolute Deviation (MAD) defined as $\text{MAD} = \text{median}_i(|z_i - \hat{z}_i|)$. In addition, we report the RMSE in the logarithmic scale of the target variable, which is defined as $\text{RMSE} = \sqrt{\frac{1}{n}\sum_i^n (y_i - \hat{y}_i)^2}$.

In addition to the point estimate performance, we also evaluate the meaningfulness of the predictive variance σ_i of our proposed model as well as other uncertainty-aware baselines using a *Quantile Performance (QP) plot* (Wu et al., 2021; Yang and Buettner, 2021). Intuitively, the predictive confidence generated by a model is only systematically meaningful if the model assigns higher confidence to the more accurate predictions and lower

[2] https://github.com/ZhiliangWu/DKAFT

confidence to the less confident ones. In the scope of our work, we interpret the predictive variance as a form of confidence estimation, where smaller values correspond to higher confidence. Therefore, the predictive variance from our model enables a formal evaluation of such expected behavior. Concretely speaking, we extract the evaluation pairs $\{y_i, \hat{y}_i\}_{i=1}^n$ (or $\{z_i, \hat{z}_i\}_{i=1}^n$) for each quantile of the predictive variance of the model $q \in \{\frac{1}{Q}, \frac{2}{Q}, \ldots, 1\}$, where the corresponding predictive variance σ_i^2 is smaller than or equal to the q-th quantile. Formally, we have the performance in each quantile as

$$\text{Performance}_\text{q} := \text{Metric}(\{(y_i, \hat{y}_i) \text{ or } (z_i, \hat{z}_i) \mid \forall \sigma_i^2 \leq q\text{-th quantile}\}),$$

where Metric$(\cdot)$ could be MAD for z_i or RMSE for y_i. We plot the performance of each quantile on the y-axis against the corresponding q values on the x-axis. For a model with meaningful uncertain-awareness, a monotonically increasing line is expected in the QP plot. Furthermore, a stronger correlation between the metric and confidence across the quantiles suggest a better quantification of the predictive uncertainty.

Finally, we plot the (empirical) Cumulative Distribution Function (CDF) of the normalized residuals to show how well our model is calibrated as a further evaluation metrics. According to our normal distribution assumption on the logarithmic scale of the target variable, it is expected to be close to the CDF of a standard normal distribution $\mathcal{N}(0, 1)$. In addition, the Continuous Ranking Probability Score (CRPS), a popular calibration metric for regression (Gneiting and Raftery, 2007; Jankowiak et al., 2020), is reported to have a quantitative comparison.

5.2. Evaluation of the PFS-PRAEGNANT and LoS-MIMIC prediction

As weak baselines, we report the performance of standard Cox and AFT regression using the R package *survival* (Terry M. Therneau and Patricia M. Grambsch, 2000; Therneau, 2020) with raw features aggregated w.r.t. the time axis. Such aggregation has been used in Esteban et al. (2015) and Yang et al. (2017a), which turns out to be a reasonable solution to deal with features with time-stamps by ignoring the order of the events.

To investigate the performance of different output layers as we discussed in Sec. 3, we train all models with the same RNN-based feature extractor. Using a linear output layer as our strong baseline (*RNN+AFT*) (Yang et al., 2017a), we include the SVGP and PPGP output layers for both prediction tasks, which are denoted as *DKAFT (SVGP)* and *DKAFT (PPGP)*, respectively. Thanks to the moderate size of the PRAEGNANT dataset, we also have an ExactGP output layer for the PFS-PRAEGNANT prediction task, which we denote as *DKAFT (ExactGP)*. To validate our proposed initialization method, we pre-trained the feature extractor using DML and then fine-tune the proposed model. In such a case, the performance of models without pre-training naturally serves as the baselines. Note that, for our DKAFT models, only the mean predictions are involved in the evaluation of point estimates. Results are summarized in Tab. 1. For NN-based CPH baselines (Kvamme et al., 2019), we report the results in Tab. 3 in Appendix C.

From Tab. 1 we can see that our DKAFT models demonstrate much stronger performance compared to the Cox Regression and AFT Regression with aggregated features. For the evaluation w.r.t. RMSE, our DKAFT models all outperform the corresponding strong baselines, the RNN+AFT models. Tab. 2 shows the p-values of paired t-tests quantifying

Table 1: Experimental results: MAD in the original scale (days) and RMSE in the logarithmic scale for PFS-PRAEGNANT and LoS-MIMIC prediction tasks. Our DKAFT models outperform the baselines, including Cox Regression, AFT regression, and RNN+AFT models. More baselines in Appendix. C.

Method	Pre-training	Progression-Free Survival		Length-of-Stay	
		MAD	RMSE	MAD	RMSE
Cox Regression	−*	200.800 ± 16.984	1.609 ± 0.054	2.727 ± 0.007	0.638 ± 0.0002
AFT Regression	−*	206.065 ± 8.988	1.685 ± 0.080	2.742 ± 0.014	0.630 ± 0.0001
RNN+AFT	None	150.918 ± 3.009	1.273 ± 0.019	2.476 ± 0.040	0.575 ± 0.003
DKAFT (ExactGP)	None	144.622 ± 8.689	1.225 ± 0.022	−†	−†
DKAFT (SVGP)	None	154.237 ± 13.490	1.211 ± 0.020	2.428 ± 0.056	0.572 ± 0.003
DKAFT (PPGP)	None	147.108 ± 6.284	1.220 ± 0.019	2.351 ± 0.021	0.563 ± 0.001
RNN+AFT	DML	138.155 ± 7.496	1.267 ± 0.007	2.452 ± 0.057	0.568 ± 0.001
DKAFT (ExactGP)	DML	$\mathbf{134.422 \pm 7.255}$	1.202 ± 0.012	−†	−†
DKAFT (SVGP)	DML	151.852 ± 11.305	1.221 ± 0.007	2.438 ± 0.079	0.567 ± 0.005
DKAFT (PPGP)	DML	146.616 ± 17.109	$\mathbf{1.195 \pm 0.008}$	$\mathbf{2.346 \pm 0.042}$	$\mathbf{0.557 \pm 0.002}$

*Not applicable to the method.
†Not possible due to the $\mathcal{O}(n^3)$ computational complexity.

the performance improvement of our DKAFT models. In the task of PFS-PRAEGNANT prediction, we can see our DKAFT models outperform RNN+AFT models significantly (with a significant level $\alpha = 0.05$). In contrast, only DKAFT (PPGP) models show significantly better RMSE performance in the LoS-MIMIC prediction task. For the evaluation regarding MAD, our DKAFT (ExactGP) model performs best in the PFS-PRAEGNANT prediction task, while it is DKAFT (PPGP) for the LoS-MIMIC prediction task. Meanwhile, we can observe a moderate improvement for most models pre-trained with DML compared to those trained from scratch.

Table 2: p-values from paired t-tests to assess the significance of improvement achieved by our DKAFT models, based on the RMSEs collected from multiple cross validations. A two factor ANOVA on the effect of different model setups can be found in the Appendix.B

Comparison candidates	No pre-training		DML	
	PFS	LoS	PFS	LoS
DKAFT (ExactGP) vs. RNN+AFT	0.021	−*	2e-4	−*
DKAFT (SVGP) vs. RNN+AFT	0.010	0.282	0.001	0.682
DKAFT (PPGP) vs. RNN+AFT	0.002	0.001	3e-4	3e-4

*Not possible due to the $\mathcal{O}(n^3)$ computational complexity.

As discussed in Sec. 3, DKAFT models are trained by either optimizing the log marginal likelihood objective or the variational approximation of it, which is essentially a generalization to mean square error in linear regression. Therefore, improved performance is expected as the GP-based output layers include inducing points during the training and inference time, which implicitly facilitates integrating all possible linear models. Meanwhile, since MAD is a more robust metric than RMSE regarding outliers, there are some performance differences of the same model between these two metrics.

The superior performance of the DKAFT (SVGP) and DKAFT (PPGP) compared to DKAFT (ExactGP) w.r.t. RMSE is a bit surprising since these models are essentially approximations of the latter. We speculate that such phenomenon comes from the sparse and high-dimensional features in the PRAEGNANT dataset, which results in redundant or even repetitive patient covariates from the feature extractor. This makes the GP model struggle to capture the correlation correctly. On the contrary, the inducing points in SVGP or PPGP are not constrained by the raw input features and could avoid this coupling during the optimization.

For the initialization with pre-training methods, DML helps attain a feature extractor, which clusters samples with similar targets in nearby regions. Models with all output layers, including linear layers, achieve moderate improvement. This can be attributed to 1) the non-convexity nature of the log marginal likelihood objective or the ELBO objective with deep neural networks, where a good starting point could offer advantages for the following optimization procedure; 2) the good initialization of the inducing inputs generated by the pre-trained feature extractor.

5.3. Evaluation of the predictive variances

Apart from improved performance for point estimates, the main advantage of the proposed method lies in the uncertainty-aware nature of the model. As a baseline, we include MC Dropout (Gal and Ghahramani, 2016), a well-known method for enabling uncertainty estimates in neural networks. By adding a dropout layer (Srivastava et al., 2014) before each weight layer, the resulting model is proved to be mathematically equivalent to a probabilistic deep Gaussian Process. In our experiments, we followed the proposed method with the suggested dropout rate of 0.2. The mean prediction and function variance are computed from performing 50 stochastic forward passes through the network. Since QP plots demonstrate unstable performance if the number of evaluation pairs in each quantile is too small (like the test set for the PFS-PRAEGNANT prediction task with only 134 samples in total), we only report the evaluation on the LoS-MIMIC prediction task in Fig. 3.

In Fig. 3 we visualize the quantile performance for MAD and RMSE, where the solid line represents an average performance and the error bar for the standard deviation across CV splits. We observe a strong monotonically increasing line in both metrics from our DKAFT (PPGP) model. For the evaluation pairs that the model is more confident with, e.g., ones corresponding to the predictive variances at quantile 10%, the MAD and RMSE are 1.163 ± 0.030 and 0.316 ± 0.005, respectively. Such results correspond to only half of the values reported in Tab. 1, indicating a significant improvement. Meanwhile, the DKAFT (SVGP) model shows an increasing dependency only in MAD. For the models with MC Dropout, there seems to be no performance difference between quantiles.

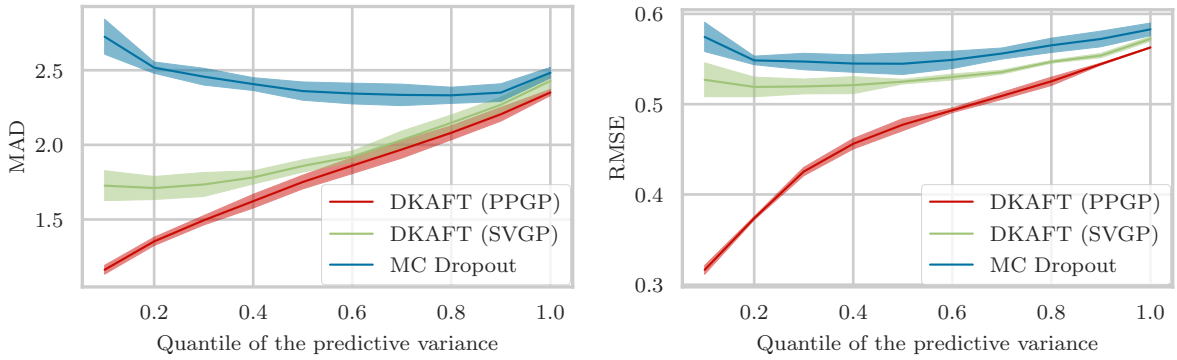

Figure 3: Quantile MAD (left) and Quantile RMSE (right) in the y-axis against quantile predictive uncertainty in the x-axis for the LoS-MIMIC prediction task. Our DKAFT model with a PPGP output layer (red) shows the strongest increasing trend in both MAD and RMSE. This indicates that the model is monotonically more confident in predictions that are indeed closer to the ground-truths. We emphasize that this is a desirable feature to expect from an uncertainty-aware prediction model.

Compared with MC Dropout, our DKAFT models deliver more meaningful uncertainty estimates since the inducing point technique realizes explicit modeling of the predictive variances. As expected, the most meaningful uncertainty estimates are visible from the models with a PPGP output layer, since the function variance is restored explicitly in the training objective compared to those with an SVGP output layer. These observations indeed motivated the application of our DKAFT (PPGP) model when meaningful uncertainty estimates become a higher priority.

5.4. Calibration of the Model

Calibration of a predictive model refers to the statistical consistency between the predictive distribution from the model and the observations (Gneiting et al., 2007), which is arguably also an important aspect for healthcare applications. In GPs, the predictive variance incorporates both the modeling uncertainty (function variance) and data uncertainty (observational noise). Even for data-points lying far from the training data, the resulting predictive distribution tends to be well-calibrated (Rasmussen and Williams, 2005). In this section, we demonstrate that our DKAFT models inherit this nice property from GPs. Meanwhile, as a popular method for calibrating neural networks, MC Dropout is included in our experiment as a baseline. We visualize the empirical CDF of the normalized residuals with the predictive variances in Fig. 4. Besides, the CRPS score is computed to have a quantitative comparison. The CRPS score generalizes the Mean Absolute Error (MAE) to probabilistic predictions.

In Fig. 4, we visualize the CDF plots for both time-to-event prediction tasks. Graphically speaking, all methods demonstrate well-calibrated behavior based on their closeness to the best possible calibrated CDF. The ECDF of our DKAFT models is closer to the ideal CDF

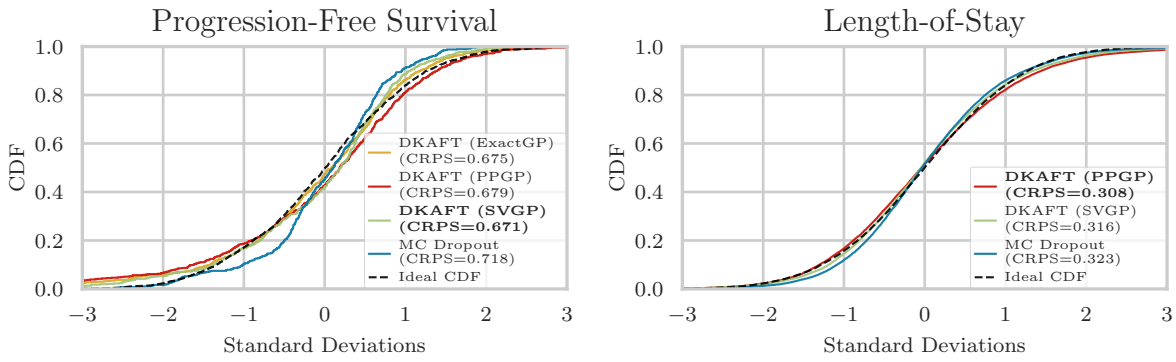

Figure 4: Empirical CDF of the normalized residual, $(y_i - \mu_i)/\sigma_i$, from different models against an "ideal CDF" from a standard normal distribution. Continuous Ranking Probability Score is reported to provide a quantitative comparison. All models show well-calibrated behavior, where our DKAFT models are better calibrated than the baseline method, MC Dropout.

than MC Dropout. Such observation is also verified by the lower values of the respective CRPS score. Within DKAFT models, the ones with an SVGP output layer perform slightly better than those with PPGP and ExactGP output layers in the PFS-PRAEGNANT prediction task. In contrast, the DKAFT models with a PPGP output layer outperform the ones with an SVGP output layer in the LoS-MIMIC prediction task.

It is worth highlighting that the inference in the MC Dropout requires multiple stochastic forward passes through the sampled network, which results in significantly slower processing than in our DKAFT models. This would further motivate the application of the analytical solutions from the proposed DKAFT models when computational efficiency plays an essential role in real-world applications, like in real-time response systems.

6. Conclusion and Future Works

In this work, we propose the Deep Kernel Accelerated Failure Time (DKAFT) model to address the lack of uncertainty estimates in recurrent neural network (RNN) based solutions for time-to-event prediction tasks. Our DKAFT model consists of an RNN encoder and a sparse GP as the prediction model. The former serves as a trainable feature extractor to embed the patient features into a latent space of abstract covariates. The GP-based output layer consumes the abstract covariates of the patients, and outputs a predictive distribution for the time-to-event prediction.

We show that the proposed model can be trained in an end-to-end fashion, like typical neural networks, using stochastic gradient descent-based methods. In addition, a deep metric learning-based pre-training method is proposed to further improve the performance of the proposed model. Through experiments on two real-world datasets, the DKAFT models show better performance in terms of the point estimates than the RNN-based models with linear output layers. More importantly, the predictive variances from our DKAFT model

reflect the confidence of the predictions. It produces better metrics evaluation in terms of RMSE and MAD with monotonically higher confidence about the predictions. Such uncertainty estimates would improve the trustworthiness of the provided model when it interacts with the physicians. Furthermore, the predictive variance also serves to improve the calibration of the model. Compared to MC Dropout, a popular method to augment the uncertainty in the neural networks, our DKAFT model shows better performance and enjoys lower computational cost, which motivates its usage in real-world applications.

As future work, we would like to further study the interpretation of the uncertainty in the proposed model. From a machine learning's perspective, it refers to checking whether a test sample lies far from the manifold constituted by the training samples. From a decision support's perspective—since our proposed model offers predictive variances based on the "neighboring" training samples in the feature spaces—it would offer practical help if it also fits physicians' understanding.

Limitations As shown in Sec. 3.2, in our DKAFT models, (right) censored observations will contribute to the training objective through its survival function instead of the probability density function. However, due to the nature of the time-to-event prediction tasks we focus on in this work, administrative censoring is not included in the experiments. Besides, the evaluation of censoring cases is also beyond the scope of this manuscript. We leave these perspectives as part of our future work.

Acknowledgement The authors acknowledge the support by the German Federal Ministry for Education and Research (BMBF), funding project "MLWin" (grant 01IS18050).

GEFÖRDERT VOM

References

Ahmed M Alaa and Mihaela van der Schaar. Deep multi-task gaussian processes for survival analysis with competing risks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 2326–2334, 2017.

Edmon Begoli, Tanmoy Bhattacharya, and Dimitri Kusnezov. The need for uncertainty quantification in machine-assisted medical decision making. *Nature Machine Intelligence*, 1(1):20–23, 2019.

Alexis Bellot and Mihaela Van Der Schaar. Flexible modelling of longitudinal medical data: A bayesian nonparametric approach. *ACM Transactions on Computing for Healthcare*, 1 (1):1–15, 2020.

Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

John Bradshaw, Alexander G de G Matthews, and Zoubin Ghahramani. Adversarial examples, uncertainty, and transfer testing robustness in gaussian process hybrid deep networks. *arXiv preprint arXiv:1707.02476*, 2017.

George H Chen. Deep kernel survival analysis and subject-specific survival time prediction intervals. In *Machine Learning for Healthcare Conference*, pages 537–565. PMLR, 2020.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, 2014.

Edward Choi, Andy Schuetz, Walter F Stewart, and Jimeng Sun. Using recurrent neural network models for early detection of heart failure onset. *Journal of the American Medical Informatics Association*, 24(2):361–370, 2017.

Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning, December 2014*, 2014.

David Collett. *Modelling survival data in medical research*. CRC press, 2015.

Cristóbal Esteban, Danilo Schmidt, Denis Krompaß, and Volker Tresp. Predicting sequences of clinical events by using a personalized temporal latent embedding model. In *2015 International Conference on Healthcare Informatics*, pages 130–139. IEEE, 2015.

Cristóbal Esteban, Oliver Staeck, Stephan Baier, Yinchong Yang, and Volker Tresp. Predicting clinical events by combining static and dynamic information using recurrent neural networks. In *2016 IEEE International Conference on Healthcare Informatics (ICHI)*, pages 93–101. IEEE, 2016.

P.A. Fasching, S.Y. Brucker, T.N. Fehm, F. Overkamp, W. Janni, M. Wallwiener, P. Hadji, E. Belleville, L. Häberle, F.A. Taran, D. Luftner, M.P. Lux, J. Ettl, V. Muller, H. Tesch, D. Wallwiener, and A. Schneeweiss. Biomarkers in patients with metastatic breast cancer and the praegnant study network. *Geburtshilfe Frauenheilkunde*, 75(01):41–50, 2015. URL http://www.praegnant.org/.

Tamara Fernández, Nicolás Rivera, and Yee Whye Teh. Gaussian processes for survival analysis. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 5021–5029, 2016.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016.

Jacob Gardner, Geoff Pleiss, Kilian Q Weinberger, David Bindel, and Andrew G Wilson. Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration. *Advances in Neural Information Processing Systems*, 31:7576–7586, 2018.

Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

Tilmann Gneiting, Fadoua Balabdaoui, and Adrian E Raftery. Probabilistic forecasts, calibration and sharpness. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(2):243–268, 2007.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330. PMLR, 2017.

Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.

Yoni Halpern, Steven Horng, Youngduck Choi, and David Sontag. Electronic medical record phenotyping using the anchor and learn framework. *Journal of the American Medical Informatics Association*, 23(4):731–740, 2016.

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.

James Hensman, Nicolo Fusi, and Neil D Lawrence. Gaussian processes for big data. In *Uncertainty in Artificial Intelligence*, page 282. Citeseer, 2013.

Alexander Hermans, Lucas Beyer, and Bastian Leibe. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*, 2017.

Graeme L Hickey, Pete Philipson, Andrea Jorgensen, and Ruwanthi Kolamunnage-Dona. Joint modelling of time-to-event and multivariate longitudinal outcomes: recent developments and issues. *BMC medical research methodology*, 16(1):1–15, 2016.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Martin Jankowiak, Geoff Pleiss, and Jacob Gardner. Parametric gaussian process regressors. In *International Conference on Machine Learning*, pages 4702–4712. PMLR, 2020.

Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3:160035, 2016.

John D Kalbfleisch and Ross L Prentice. *The Statistical Analysis of Failure Time Data*, volume 360. John Wiley & Sons, 2002.

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.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *NIPS*, 2017.

Haitao Liu, Yew-Soon Ong, Xiaobo Shen, and Jianfei Cai. When gaussian process meets big data: A review of scalable gps. *IEEE transactions on neural networks and learning systems*, 31(11):4405–4423, 2020.

Tomáš Mikolov et al. Statistical language models based on neural networks. *Presentation at Google, Mountain View, 2nd April*, 80:26, 2012.

Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. A metric learning reality check. In *European Conference on Computer Vision*, pages 681–699. Springer, 2020.

Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 427–436, 2015.

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. 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., 2019.

Ross L Prentice. Linear rank tests with right censored data. *Biometrika*, 65(1):167–179, 1978.

Sanjay Purushotham, Chuizheng Meng, Zhengping Che, and Yan Liu. Benchmarking deep learning models on large healthcare datasets. *Journal of biomedical informatics*, 83:112–134, 2018.

Joaquin Quinonero-Candela and Carl Edward Rasmussen. A unifying view of sparse approximate gaussian process regression. *The Journal of Machine Learning Research*, 6: 1939–1959, 2005.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005. ISBN 026218253X.

Dimitris Rizopoulos. Dynamic predictions and prospective accuracy in joint models for longitudinal and time-to-event data. *Biometrics*, 67(3):819–829, 2011.

Alan D Saul. *Gaussian Process Based Approaches for Survival Analysis*. PhD thesis, University of Sheffield, 2016.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

Edward Snelson and Zoubin Ghahramani. Sparse gaussian processes using pseudo-inputs. *Advances in Neural Information Processing Systems*, 18:1259–1266, 2006.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

Terry M. Therneau and Patricia M. Grambsch. *Modeling Survival Data: Extending the Cox Model*. Springer, New York, 2000. ISBN 0-387-98784-3.

Terry M Therneau. *A Package for Survival Analysis in R*, 2020. URL https://CRAN.R-project.org/package=survival. R package version 3.2-7.

Michalis Titsias. Variational learning of inducing variables in sparse gaussian processes. In *Artificial intelligence and statistics*, pages 567–574. PMLR, 2009.

Volker Tresp. A bayesian committee machine. *Neural computation*, 12(11):2719–2741, 2000.

Volker Tresp, J Marc Overhage, Markus Bundschus, Shahrooz Rabizadeh, Peter A Fasching, and Shipeng Yu. Going digital: a survey on digitalization and large-scale data analytics in healthcare. *Proceedings of the IEEE*, 104(11):2180–2206, 2016.

Hans C Van Houwelingen. Dynamic prediction by landmarking in event history analysis. *Scandinavian Journal of Statistics*, 34(1):70–85, 2007.

Ke Alexander Wang, Geoff Pleiss, Jacob R Gardner, Stephen Tyree, Kilian Q Weinberger, and Andrew Gordon Wilson. Exact gaussian processes on a million data points. *Advances in Neural Information Processing Systems*, 32, 2019.

Kilian Q Weinberger, John Blitzer, and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. In *Advances in neural information processing systems*, pages 1473–1480, 2006.

Chris Williams and Matthias Seeger. Using the nyström method to speed up kernel machines. *Advances in Neural Information Processing Systems 13*, 2001.

Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P Xing. Deep kernel learning. In *Artificial intelligence and statistics*, pages 370–378. PMLR, 2016.

Zhiliang Wu, Yinchong Yang, Yunpu Ma, Yushan Liu, Rui Zhao, Michael Moor, and Volker Tresp. Learning individualized treatment rules with estimated translated inverse propensity score. In *2020 IEEE International Conference on Healthcare Informatics (ICHI)*, pages 1–11, 2020. doi: 10.1109/ICHI48887.2020.9374397.

Zhiliang Wu, Yinchong Yang, Jindong Gu, and Volker Tresp. Quantifying predictive uncertainty in medical image analysis with deep kernel learning. *arXiv preprint arXiv:2106.00638*, 2021.

Cao Xiao, Edward Choi, and Jimeng Sun. Opportunities and challenges in developing deep learning models using electronic health records data: a systematic review. *Journal of the American Medical Informatics Association*, 25(10):1419–1428, 2018.

Yinchong Yang and Florian Buettner. Multi-output gaussian processes for uncertainty-aware recommender systems. *arXiv preprint arXiv:2106.04221*, 2021.

Yinchong Yang, Peter A Fasching, and Volker Tresp. Modeling progression free survival in breast cancer with tensorized recurrent neural networks and accelerated failure time models. In *Machine Learning for Healthcare Conference*, pages 164–176. PMLR, 2017a.

Yinchong Yang, Peter A Fasching, and Volker Tresp. Predictive modeling of therapy decisions in metastatic breast cancer with recurrent neural network encoder and multinomial hierarchical regression decoder. In *2017 IEEE International Conference on Healthcare Informatics (ICHI)*, pages 46–55. IEEE, 2017b.

Tongtong Yuan, Weihong Deng, Jian Tang, Yinan Tang, and Binghui Chen. Signal-to-noise ratio: A robust distance metric for deep metric learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4815–4824, 2019.

Appendix A. Feature Set in MIMIC-III

To best represent the clinical status, we extracted both static and sequential information from the MIMIC-III. The included features are the same as the *Feature Set C* defined in Purushotham et al. (2018). In the chosen features, most have continuous values except for acquired immunodeficiency syndrome, hematologic malignancy, metastatic cancer, and admission type. The missing rates of each feature can be found in Table A.26 in Purushotham et al. (2018).

Static Information: age, acquired immunodeficiency syndrome, hematologic malignancy, metastatic cancer, admission type

Sequential Information: Gastric Tube, Stool Out Stool, Urine Out Incontinent, Ultrafiltrate, Fecal Bag, Chest Tube 1, Chest Tube 2, Jackson Pratt 1, OR EBL, Pre-Admission, TF Residual, Albumin 5%, Fresh Frozen Plasma, Lorazepam (Ativan), Calcium Gluconate, Midazolam (Versed), Phenylephrine, Furosemide (Lasix), Hydralazine, Norepinephrine, Magnesium Sulfate, Nitroglycerin, Insulin Regular, Morphine Sulfate, Potassium Chloride, Packed Red Blood Cells, Gastric Meds, D5 1/2NS, LR, Solution, Sterile Water, Piggyback, OR Crystalloid Intake, PO Intake, GT Flush, KCL (Bolus), Magnesium Sulfate (Bolus), Hematocrit, Platelet count, Hemoglobin, MCHC, MCH, MCV, Red blood cells, RDW, Chloride, Anion gap, Creatinine, Glucose, Magnesium, Calcium total, Phosphate, INR(PT), PT, PTT, Lymphocytes, Monocytes, Neutrophils, Basophils, Eosinophils, PH, Base excess, Calculated total CO2, PCO2, Specific gravity, Lactate, Alanine aminotransferase (ALT), Asparate aminotransferase (AST), Alkaline phosphatase, ALBUMIN, Aspirin, Bisacodyl, Docusate Sodium, Humulin-R Insulin, Metoprolol Tartrate, Pantoprazolel, Arterial Blood Pressure diastolic, Arterial Blood Pressure mean, Respiratory Rate, Alarms On, Minute Volume Alarm-Low, Peakinsp.Pressure, PEEP set, Minute Volume, Tidal Volume (observed), Minute Volume Alarm High, Mean Airway Pressure, Central Venous Pressure, Respiratory Rate (Set), Pulmonary Artery Pressure mean, O2Flow, Glucose fingerstick, Heart Rate Alarm Low, Pulmonary Artery Pressure systolic, Tidal Volume (set), Pulmonary Artery Pressure diastolic, SpO2 Desat Limit, Resp Alarm High, Skin Care, gcsverbal, gcsmotor, gcseyes, systolic blood pressure abp mean, heart rate, body temperature, pao2, fiO2, urinary output sum, serum urea nitrogen level, white blood cells count mean, serum bicarbonate level mean, sodium level mean, potassium level mean, bilirubin level, ie ratio mean, diastolic blood pressure mean, arterial pressure mean, respiratory rate, SpO2 peripheral, glucose, weight, height, hgb, platelet, chloride, creatinine, norepinephrine, epinephrine, phenylephrine, vasopressin, dopamine, midazolam, fentanyl, propofol, peep, ph.

Appendix B. ANOVA as an ablation study

We perform ANOVA to further verify the improvements reported in Tab. 1 in terms of RMSE. In case of the progression-free survival (PFS-PRAEGNANT) prediction task (with Fasching et al. (2015) dataset), we report a p-value of 0.064 w.r.t. the four choices of the prediction model, and p-value of 5.9e-6 w.r.t. applying deep metric learning as pre-training or not. In case of the length-of-stay (LoS-MIMIC) prediction task (with Johnson et al. (2016) dataset), we report a p-value of 1.5e-6 w.r.t. the three choices of the prediction model, and p-value of 2.0e-9 w.r.t. applying deep metric learning as pre-training or not.

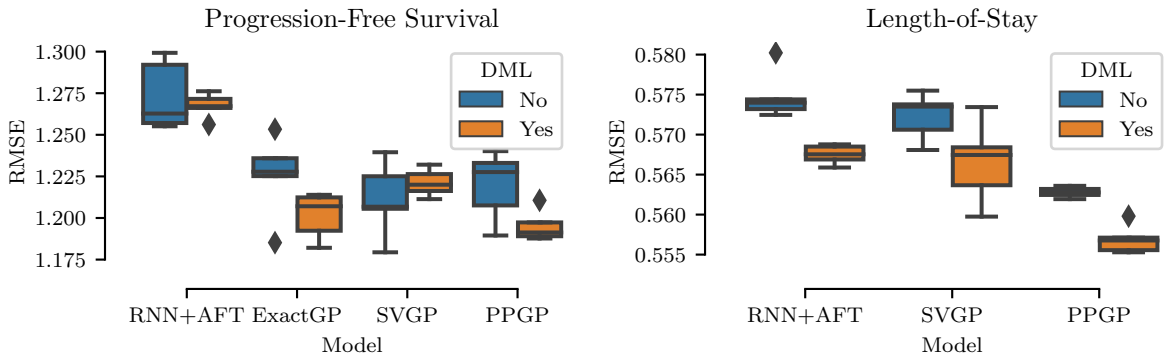

Figure 5: Grouped Box plots visualizing a comparison between different models, RNN+AFT and ExactGP, SVGP, PPGP of our DKAFT models. A similar conclusion to the reported p-values from ANOVA could be drawn from these plots.

It appears that only in the case of PFS-PRAEGNANT, the choice of the prediction model does not have a significant impact on the performance, presumably due to the relatively small number of patient samples. In Fig. 5 we visualize the effects of these two factors as grouped box plots.

Appendix C. Experimental Results with More Baselines and Metrics

We conducted experiments for NN-based CPH models using the PyCox package[3]. Both continuous-time models (*DeepSurv, CoxTime, CoxCC, and PCHazard*) and discrete-time models (*LogisticHazard, PMF, DeepHit, MTLR, and BCESurv*) are included, where the latter perform much worse w.r.t. the metrics we are interested in this manuscript. Therefore, we only report the results for continuous-time models. Besides, we include two neural network architectures for these models, where a Multiple-Layer Perceptron (MLP) receives aggregated features like Cox Regression and a Recurrent Neural Network (RNN) refers to the same base network we used for our DKAFT models. In addition, we report the concordance index (C-Index) (Harrell et al., 1982) of all methods to show their respective discriminative performance. From Tab. 3 we can see, that these continuous-time models only show comparable performance to our strong baseline (RNN+AFT). Meanwhile, the RNN variants of the same model always improve the performance w.r.t. RMSE and C-Index as they take the time-dependency of the patient information into consideration.

[3] https://github.com/havakv/pycox

Table 3: Experimental results: MAD in the original scale (days) and RMSE in the logarithmic scale for PFS-PRAEGNANT and LoS-MIMIC prediction tasks. Our DKAFT models outperform the baselines, including Cox Regression, AFT regression, NN-based CPH models, and RNN+AFT models.

Method	Pretraining	Progression-Free Survival			Length-of-Stay		
		MAD	RMSE	C-Index	MAD	RMSE	C-Index
Cox Regression	–*	200.800 ± 16.984	1.609 ± 0.054	0.676 ± 0.004	2.727 ± 0.007	0.638 ± 0.0002	0.683 ± 0.001
AFT Regression	–*	206.065 ± 8.988	1.685 ± 0.080	0.656 ± 0.008	2.742 ± 0.014	0.630 ± 0.0001	0.684 ± 0.001
MLP+DeepSurv	–*	153.900 ± 7.151	1.293 ± 0.024	0.731 ± 0.006	2.486 ± 0.028	0.593 ± 0.005	0.715 ± 0.003
RNN+DeepSurv	–*	166.700 ± 6.565	1.252 ± 0.016	0.733 ± 0.005	2.456 ± 0.039	0.579 ± 0.003	0.721 ± 0.003
MLP+CoxTime	–*	153.900 ± 6.924	1.387 ± 0.018	0.697 ± 0.004	2.599 ± 0.034	0.606 ± 0.008	0.703 ± 0.005
RNN+CoxTime	–*	146.000 ± 16.634	1.278 ± 0.010	0.723 ± 0.002	2.515 ± 0.051	0.585 ± 0.004	0.718 ± 0.003
MLP+CoxCC	–*	153.100 ± 9.557	1.305 ± 0.010	0.729 ± 0.003	2.463 ± 0.015	0.585 ± 0.005	0.719 ± 0.003
RNN+CoxCC	–*	153.800 ± 6.022	1.262 ± 0.009	0.728 ± 0.001	2.479 ± 0.018	0.578 ± 0.002	0.724 ± 0.002
MLP+PCHazard	–*	167.580 ± 17.490	1.577 ± 0.189	0.694 ± 0.018	3.036 ± 0.019	1.117 ± 0.020	0.665 ± 0.006
RNN+PCHazard	–*	141.041 ± 8.821	1.271 ± 0.017	0.710 ± 0.007	3.055 ± 0.021	0.593 ± 0.009	0.723 ± 0.002
RNN+AFT	None	150.918 ± 3.009	1.273 ± 0.019	0.729 ± 0.008	2.476 ± 0.040	0.575 ± 0.003	0.725 ± 0.002
DKAFT (ExactGP)	None	144.622 ± 8.689	1.225 ± 0.022	0.738 ± 0.008	–†	–†	–†
DKAFT (SVGP)	None	154.237 ± 13.490	1.211 ± 0.020	0.747 ± 0.007	2.428 ± 0.056	0.572 ± 0.003	0.727 ± 0.002
DKAFT (PPGP)	None	147.108 ± 6.284	1.220 ± 0.019	0.745 ± 0.006	2.351 ± 0.021	0.563 ± 0.001	0.734 ± 0.001
RNN+AFT	DML	138.155 ± 7.496	1.267 ± 0.007	0.732 ± 0.003	2.452 ± 0.057	0.568 ± 0.001	0.732 ± 0.001
DKAFT (ExactGP)	DML	$\mathbf{134.422 \pm 7.255}$	1.202 ± 0.012	$\mathbf{0.752 \pm 0.003}$	–†	–†	–†
DKAFT (SVGP)	DML	151.852 ± 11.305	1.221 ± 0.007	0.747 ± 0.002	2.438 ± 0.079	0.567 ± 0.005	0.733 ± 0.003
DKAFT (PPGP)	DML	146.616 ± 17.109	$\mathbf{1.195 \pm 0.008}$	$\mathbf{0.752 \pm 0.006}$	$\mathbf{2.346 \pm 0.042}$	$\mathbf{0.557 \pm 0.002}$	$\mathbf{0.738 \pm 0.001}$

*Not applicable to the method.
†Not possible due to the $\mathcal{O}(n^3)$ computational complexity.