Hawkes Process with Flexible Triggering Kernels

Yamac Isik[1] YAMAC@AEROLOGY.AI

Paidamoyo Chapfuwa[2] PCHAPFUWA@MICROSOFT.COM

Connor Davis[3] CONNOR.DAVIS@DUKE.EDU

Ricardo Henao[1,4] RICARDO.HENAO@DUKE.EDU

[1] *Department of Biostatistcs and Bioinformatics, Duke University, Durham, USA*
[2] *Microsoft Health Futures, Microsoft Research, Redmond, USA*
[3] *Duke Institute for Health Innovation, Duke University, Durham, USA*
[4] *BESE Division, King Abdullah University of Science and Technology, Thuwal, KSA*

Editor: Editor's name

Abstract

Recently proposed encoder-decoder structures for modeling Hawkes processes use transformer-inspired architectures, which encode the history of events via embeddings and self-attention mechanisms. These models deliver better prediction and goodness-of-fit than their RNN-based counterparts. However, they often require high computational and memory complexity and fail to adequately capture the triggering function of the underlying process. So motivated, we introduce an efficient and general encoding of the historical event sequence by replacing the complex (multilayered) attention structures with triggering kernels of the observed data. Noting the similarity between the triggering kernels of a point process and the attention scores, we use a triggering kernel to replace the weights used to build history representations. Our estimator for the triggering function is equipped with a sigmoid gating mechanism that captures local-in-time triggering effects that are otherwise challenging with standard decaying-over-time kernels. Further, taking both event type representations and temporal embeddings as inputs, the model learns the underlying triggering type-time kernel parameters given pairs of event types. We present experiments on synthetic and real data sets widely used by competing models, and further include a COVID-19 dataset to illustrate the use of longitudinal covariates. Our results show the proposed model outperforms existing approaches, is more efficient in terms of computational complexity, and yields interpretable results via direct application of the newly introduced kernel.

1. Introduction

Temporal point processes are the preferred choice when modeling asynchronous event sequences (Cox and Isham, 1980). Of particular interest is the Hawkes Process (Hawkes, 1971), a self-exciting point process that is used in numerous applications. In finance, the Hawkes process is typically used to model market returns, volatility, and stability (Yang et al., 2018; Lee and Seo, 2017; Bacry et al., 2015). In social media, it is employed to analyze malicious activity, and actions of users on social media platforms (Alvari and Shakarian, 2019; Rizoiu et al., 2017). More recently, Hawkes processes have become critical in healthcare scenarios, such as modeling adverse drug reactions, disease progression, and the spread of COVID-19 outbreaks (Bao et al., 2017; Chiang et al., 2021; Sun et al., 2021).

Hawkes (1971) first introduced the Hawkes process to model the aftershocks of earthquakes. The self-excitation of the conditional intensity function allowed for modeling the clustered behavior of aftershock occurrences. Ozaki (1979) introduced the parametric estimation of the intensity function via the likelihood function. Later, non-parametric approaches based on the expectation-maximization (EM) algorithm have been proposed to estimate the conditional intensity function (Lewis and Mohler, 2011; Chen and Hall, 2016). Though flexible, these approaches require expensive computational budgets needed to solve additional ordinary differential equation terms or the choice of the kernel functions needed to estimate the intensity

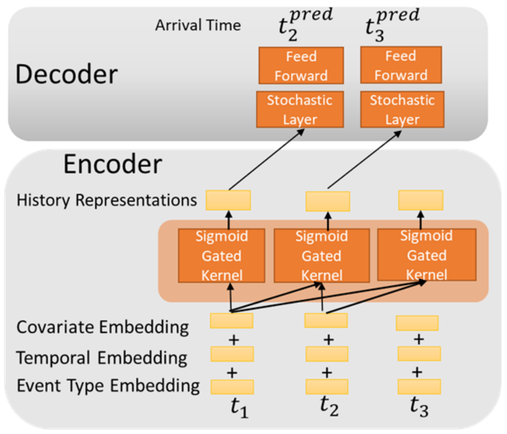

Figure 1: The SGHP (sigmoid gated Hawkes process) is computationally efficient, powerful in terms of predictive performance, and allows for interpretable results via the estimation of pairwise triggering functions. Note that in a transformer-based Hawkes process, the sigmoid gated kernel is replaced by a stack of self-attention modules with feed-forward transformations.

function. Pan et al. (2021) introduced an approach based on Gaussian Process (GP) to non-parametrically model the triggering functions of the point processes. Their work delivers on predictive performance and interpretability, but lacks the efficiency of non-GP approaches.

Most recent deep learning approaches for modeling event sequences can be grouped into two categories. These are RNN-based models (Du et al., 2016; Mei and Eisner, 2017; Shchur et al., 2020; Omi et al., 2019), and Attention-Based models Vaswani et al. (2017); Zhang et al. (2020); Zuo et al. (2020); Zhang et al. (2022); Gupta and Bedathur (2022); Gupta et al. (2022). Both of these approaches use an encoding-decoding structure. They model intensity as a non-linear function of the sequence history to summarize the past events and use feed-forward decoders to infer the conditional intensity, cumulative intensity, or the conditional probabilities. RNN-based models inherit the known issues associated with RNNs, such as vanishing or exploding gradients (Pascanu et al., 2013), or the inability to capture long-term or very short-term dependencies (Zuo et al., 2020). We note however that exploding gradient issues can be largely alleviated via gradient clipping. Alternatively, attention-based structures can predict arrival times relatively well, but suffer from high computational and memory requirements. Furthermore, their results can be more interpretable than RNN-based approaches but are limited to the relative contribution of past events.

Motivated to address the challenges faced by RNN- and attention-based approaches to the modeling of Hawkes processes, we propose a method that replaces the complex attention mechanisms with a more efficient, flexible, and interpretable encoding structure. Specifically, our approach replaces the attention mechanism with the *direct* estimation of the triggering kernels by taking advantage of the resemblance between the attention weights used to create the history embedding and the triggering function of the Hawkes process itself. The resulting encoder is substantially more efficient than both RNNs and multi-layered attention structures, while being able to learn the triggering kernels of the underlying distribution directly from data. The proposed kernel uniquely combines a sigmoid gate (Steinruecken et al., 2019) with the rational quadratic function, thus enabling the learning of a more general family of triggering kernels, such as *local-in-time* triggering, as well as the standard decaying-over-time functions. In addition to flexibility, the proposed *sigmoid gated Hawkes process* (SGHP) model can accommodate a different triggering kernel for each pair of event types. Utilizing the event type embeddings introduced by existing attention-based models, we estimate the triggering function of each event type combination, while preventing the rapid growth of the parameter set, which typically scales quadratically with the number of event types. Figure 1 compares the proposed approach with existing transformer-based models.

Our contributions are as follows: *i*) We propose an encoder that is inherently more efficient and flexible than the currently existing transformer-based approaches. We show the flexibility of our model by introducing a stochastic decoding layer that captures the conditional distributional information and uncertainty around our arrival time predictions. *ii*) To the best of our knowledge, this is the first parametric approach that explores different structures to produce general triggering kernels (without deep architectures) geared toward point process modeling. Further, the proposed composite kernel is poised to encourage the development of new kernel mixtures for even more flexible triggering kernel learning. *iii*) We introduce an efficient way of estimating kernels for each event type pair via the use of type embeddings that allow interpreting the effect of observing an event type conditioned on other event types.

To showcase our contributions, we present experiments on both artificial and a wide range of real-world data; including comparisons with state-of-the-art models. Results indicate that our model outperforms all the baseline models in terms of arrival time prediction and reproducing underlying triggering kernels, while being very competitive for event type prediction. We present results on a COVID-19 dataset of the patients admitted to the emergency department at Duke University Hospital and highlight the interpretability of our model via the learned triggering kernels for several event type pairings.

Generalizable Insights about Machine Learning in the Context of Healthcare

This work was initially motivated by the need to understand the behavior (journey) of COVID-19 patients admitted with the emergency department at our institution. Specifically, to better understand and predict transitions to ICU step-down units, discharge and death. However, the methods are not only very relevant to healthcare scenarios, but also generalizable and applicable to other settings beyond healthcare (we present results on three additional real-world datasets, one of which being MIMIC-II), where the interest is the analysis and prediction with datasets that can be construed as point processes, *e.g.*, sparse time series with irregular sampling rate.

2. Background

Let S be a sequence of L observed events (timestamps) from K different possible event types. Each sequence S can be represented as $S_n = \{(u_i, t_i)\}_{i=1}^{L_n}$, where $u_i \in \{1, 2, ...K\}$, and u_i and t_i correspond to the type and timestamp of an observed event, respectively. We use the subscript n to represent an individual sequence, and i and j to denote events within a given sequence. Our objective is to model the underlying distribution of the collection of sequences for a given dataset $\{S_n\}_{n=1}^{N}$ of N sequences, as well as predicting the *arrival times* and *event types* of future events conditioned on the *history* of past events. Similar to other models in the literature, we assume there are no disruptions in the event sequences, *i.e.*, all occurring events are observed.

Temporal Point Process Temporal point processes (TPPs) are most often used to model the distribution of event sequences. Each temporal point process is characterized by its conditional intensity function $\lambda(\cdot)$, defined as

$$\lambda(t|h_j) = \frac{f(t|h_j)}{1 - F(t|h_j)}, \tag{1}$$

where h_j is the history of previous events for $t_i < j$, for $j = t$, *i.e.*, $h_j = \{(t_1, k_1), (t_2, k_2), \ldots, (t_i, k_i)\}$, $f(t|h_j)$ is the conditional probability density function for the (next) event at time t given the history of the sequence and $F(t|h_j)$ is the corresponding conditional cumulative distribution function.

Hawkes Process The Hawkes process (Hawkes, 1971) is a special case of a TPP with the following intensity function

$$\lambda(t) = \mu(t) + \sum_{i \in h_t} \phi(t - t_i), \tag{2}$$

where $\mu(t)$ is the exogenous background intensity and $\phi(\cdot)$ is the *triggering kernel* that allows the intensity function to depend on past events. Note that (2) is fully additive provided that both $\mu(t)$ and $\phi(\cdot)$ are non-negative functions. Moreover, in some practical scenarios, the background intensity is assumed to be constant (Rasmussen and Williams, 2005). Similarly, for the multivariate (multi-event) case

$$\lambda_k(t) = \mu_k(t) + \sum_{i \in h_t} \phi_{k_i k}(t - t_i), \tag{3}$$

where $\phi_{k_i k}(\cdot)$ represents the triggering kernel of an event of type k_i on the current event type k, and $\mu_k(t)$ is the background intensity for type k.

History Representations via Self Attention Models based on self-attention have been proposed to capture the pairwise influence of all previous events in a sequence as a means to create historical representations (Zhang et al., 2020; Zuo et al., 2020). Specifically, the historical representation for event j in the sequence is represented as follows

$$h_j = \left(\sum_{i<j} f(x_j, x_i) x_i \right) / \sum_{i<j} f(x_j, x_i), \tag{4}$$

where x_j is the latent vector (embedding) of the j-th event in the sequence and is defined below, and $f(\cdot, \cdot)$ is a similarity function usually specified as the exponentiated inner product between two embeddings, *i.e.*, $f(x_j, x_i) = \exp(x_j^T x_i)$. Note that other similarity functions are available, however, the exponentiated inner product is still one of the most popular choices.

Temporal Encoding Attention-based models do not intrinsically capture (positional) information about the order of the events in a sequence like, for instance, RNN-based models do through recurrence. Zuo et al. (2020) introduced positional embeddings to account for such information into their encoding architecture. Further, Zhang et al. (2020) modified these embeddings for event sequences by incorporating event timestamps into the positional embeddings.

These modified positional embeddings, also known as *temporal embeddings*, are represented as t_i^{enc} for the i-th event occurring at time t_i. Assuming a temporal embedding in D dimensions, each of its elements, d, are defined as follows

$$t_{i,d}^{\text{enc}} = \begin{cases} \sin(w_d i + \omega_d t_i), & \text{if } d \text{ is even} \\ \cos(w_d i + \omega_d t_i), & \text{if } d \text{ is odd} \end{cases}, \tag{5}$$

where w_d is defined as $w_d = 1/(10,000^{2d/D})$ and depicts the angular frequency of the d-th dimension. Moreover, ω_d is a scaling parameter that controls the weight of the time-shift and is learned from the data.

Event Type Embedding To represent each event type in the sequence, a linear embedding layer is specified as in Zhang et al. (2020) via $e_i^{\text{enc}} = e_i W$, where e_i^{enc} is a D-dimensional (dense) vector representing the embedding for event at time t_i, e_i its one-hot encoding, and $W \in \mathbb{R}^{KxD}$ is the embedding matrix, which is learned from data. Note that there are only K distinct e_i^{enc} event type embeddings, which (in a slight abuse of notation) are indexed in e_i^{enc} over time using the subscript i.

3. Replacing Attention with a Kernel

Below we describe the proposed *sigmoid gated kernel* and explain how we use it as a replacement for the attention mechanism in existing architectures (Zuo et al., 2020; Zhang et al., 2020).

Self Attention via the Triggering Function We start by noting the similarity between the weights of the previous events for the attention mechanism in (2) obtained via $f(\cdot, \cdot)$, and the triggering kernel $\phi_{k_i k}(\cdot)$, for a conventional Hawkes process. Note that the triggering kernel $\phi_{k_i k}(\cdot)$ captures the extent of the influence of past events, for $t_i < t$, on the (future) event at time t. Similarly, $f(x_j, x_i)$ captures the influence of a past event x_i on the future event x_j, by proxy, leveraging representations x_i and x_j, for which we have

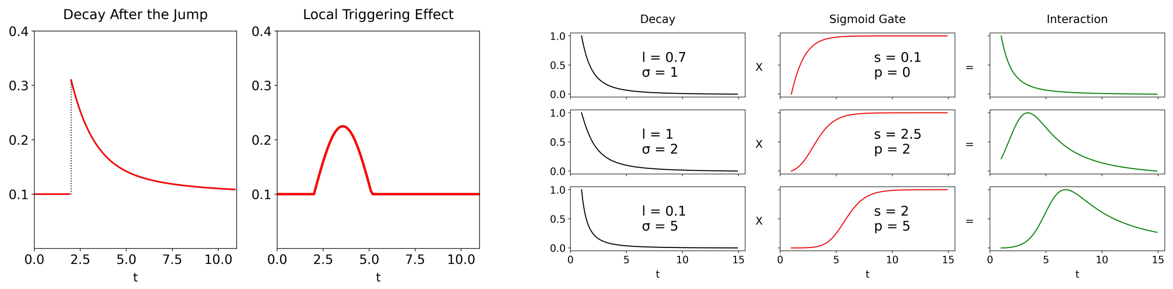

Figure 2: Standard RQ kernel *vs.* idealized local-effect kernel triggered at $t = 2$ (top), and examples of the proposed combination of the RQ kernel and the sigmoid gate in (8) with various parameters (bottom).

$x_i = t_i^{\text{enc}} + e_i^{\text{enc}}$, that encodes both temporal and event type information Zhang et al. (2020). A significant difference is that in the standard Hawkes process, a triggering kernel is specified for each event-type pairwise combination, whereas in models based on the self-attention mechanism, such information is encapsulated in the event type embedding defined above. Alternatively, we propose to directly learn a flexible triggering kernel combinations for event types while still considering latent representations for the events in the history of events. We now define the historical representation for the j-th event in the sequence as follows

$$h_j = \sum_{i \leq j} q_{k_i k_j}(|t_i - t_j|) x_i, \tag{6}$$

where $q_{k_i k_j}(\cdot)$ is a triggering kernel like in the standard Hawkes processes in (2), one for each pair of event type combinations, and x_i is the embedding of the i-th event obtained from the sum of the event type and temporal embeddings, $x_i = t_i^{\text{enc}} + e_i^{\text{enc}}$. Note that one can concatenate the embeddings to decouple the contribution of temporal and event type embeddings. However, based on our experiments we did not observe a significant increase in performance and chose to use the sum for efficiency purposes. In situations when longitudinal covariates are available, they can be readily incorporated as $x_i = [t_i^{\text{enc}} + e_i^{\text{enc}} |\ u_i^{\text{enc}}]$, where u_i^{enc} is the representation for the covariates at time t_i (see Section 6 for an example and Appendix D for details). In (6) we need not to normalize the weights as in (4) because in (6), h_j is meant to represent the aggregated, historical intensity rather than the average historical embedding.

Though for large values of K it may seem inefficient to specify a triggering kernel for each combination, (6) allows for the behavior of each event type combination to be modeled separately over time, unlike implicitly done so in attention-based approaches. Below we will show that kernels for all event type combinations can be obtained efficiently by specifying a flexible kernel whose parameters are set and learned separately for each event type combination.

Flexible Kernel with Decay and Gating Effects A natural choice for estimating a triggering function with a decaying effect over time is the squared exponential kernel, *i.e.*, the radial basis function (RBF), which is widely used in machine learning applications such as kernel machines (Hearst et al., 1998) and Gaussian processes (Rasmussen and Williams, 2005). Instead, we use the rational quadratic (RQ) kernel since it is a mixture of infinitely many squared exponential kernels, thus in principle, a more flexible yet easy to compute generalization of the squared exponential kernel (Rasmussen and Williams, 2005). For a one dimensional process (with a single event type), the rational quadratic kernel is defined as follows

$$k(d) = \sigma^2 \left(1 + \frac{d^2}{2\alpha\ell^2}\right)^{-\alpha}, \tag{7}$$

where $\alpha > 0$ and $\ell > 0$ control the decay behavior, σ is the scaling parameter, and d is a pairwise distance function, *e.g.*, $d = |t_i - t_j|$ in (6).

The RQ function is suitable for capturing monotonically decaying effects, which are fairly standard in Hawkes processes. However, underlying triggering kernels in real-world point processes can be often more complex. As a particular example illustrated in Figure 2, consider a situation where the effect of a past event on the current event ($t = 2$) does not decrease as the difference between event times grows, but rather exhibits a localized behavior (at $t = 4$) in which the effect is maximized at certain time difference (larger than zero). Such local effect is not possible to capture with regular decaying functions. To capture these local effects, we multiply the rational quadratic kernel with a sigmoid gate function, thus allowing the resulting kernel to capture both decay and local triggering effects. The proposed triggering kernel is then

$$q(d;\theta) = \sigma^2 \left(1 + \frac{d^2}{2\alpha\ell^2}\right)^{-\alpha} \left((1 + e^{p-d})^{-s}\right), \tag{8}$$

where the first term is the rational quadratic kernel defined in (7), the second term is the generalized logistic function (Gupta and Kundu, 2010), and $\theta = \{\sigma, \alpha, \ell, p, s\}$. We modify the sigmoid function by adding the parameters $p > 0$ and $s > 0$, which denote the location and rate of the change, respectively. As p changes, the change-point moves across time, while s controls the spread of the change. Moreover, l and α still control the decay while σ^2 is now the scaling parameter of the whole function. Figure 2 illustrates the behavior of (8) for different parameters choices and how the kernel flexibly models different triggering functions. We note that there are other ways of flexibly compose kernels, for instance via mixtures of kernels. However, these approaches require higher computational resources and thus are less convenient than the proposed Sigmoid gated approach in terms of efficiency. We leave the introduction of more complex mixtures of kernels for future work.

Efficient Learning of Event Type Kernel Pairs Provided the triggering kernel function being able to capture decay and local effects as defined above in (8), we can proceed to generalize it for the multivariate (multi-event) process. Inconveniently, the number of parameters for the triggering kernels scales quadratically with the number of unique event types, K, which may render the modeling prohibitive even for moderately large K. Specifically kernel parameters scale with $\mathcal{O}(PK^2)$, where P is the number of free parameters for the triggering kernel, *i.e.*, $|\theta| = 5$ in the proposed approach according to (8). For efficiency purposes, we estimate these parameters with feed-forward networks whose input are the concatenation of type embeddings, $\theta_{v|u} = g(e^{\text{enc}}_{v|u})$, where $e^{\text{enc}}_{v|u} = [e^{\text{enc}}_{v} \mid e^{\text{enc}}_{u}]$ is the concatenation of embeddings for event types v and u, and $g(\cdot)$ is specified as a collection of separate fully connected networks via

$$r_{v|u} = \text{softplus}(W_r^T e^{\text{enc}}_{v|u} + b_r), \tag{9}$$

where $r_{v|u}$ is a component of θ and $r = \{\sigma, \alpha, \ell, p, s\}$. The mapping in (9) indicates that each parameter in θ has a corresponding weight vector $W_r \in \mathbb{R}^D$ and bias $b_r \in \mathbb{R}$. As a result, kernel parameters scale linearly with P and D, *i.e.*, $\mathcal{O}(PD)$, thus effectively removing their dependency on K. Further, the softplus activation function ensures the non-negativity of the parameters. Note that one could also specify a single multilayer network for all parameters, however, in practice we found it more computationally expensive, but without substantial performance benefits.

4. Predicting the Next Event

Our model characterizes the underlying distribution of the point process by learning (temporal and event type) embeddings and the triggering kernels during the encoding stage. This allows for added flexibility in the decoder architecture. Specifically, by directly emphasizing the prediction of the next event times and types during the decoding stage, we can boost the predictive ability of the model. Further, we introduce stochasticity into the decoder to increase the variation (uncertainty) in the predictions and approximate the predictive conditional distribution function for the arrival times.

Arrival Time Prediction Given the historical embeddings for each time step, we can focus on predicting the next arrival time at each step. A straightforward way of obtaining arrival time predictions is to use a neural network whose input is the historical representation. Such a feed-forward neural network is defined as

$$t_{j+1}^{\text{pred}} = \text{softplus}(W_t^T h_j + b_t), \tag{10}$$

where h_j is the encoded history vector defined in (6), $W_t \in \mathbb{R}^D$ and $b_t \in \mathbb{R}$ are the weight and bias of the feed-forward network, respectively. Note that t_{j+1}^{pred} stands for the arrival time prediction corresponding to event $j+1$, and that we use the information (history representation) we have up to event j to predict the arrival time for the next time step $j+1$. The softplus activation is added to the output layer to ensure the arrival time predictions stay non-negative.

Adding Stochasticity to Predictions As previously leveraged for generative models with adversarial learning (Goodfellow et al., 2014; Mirza and Osindero, 2014), we introduce stochasticity into our arrival time prediction by sampling noise vectors from an easy to sample distribution and adding them to the history representations. Specifically, the stochastic layer samples noise vectors of size $|h_j|$ from a uniform distribution. Each noise vector is passed through a linear layer before being added to h_j, *i.e.*,

$$t_{j+1,m}^{\text{pred}} = \text{softplus}(W_t(W_h h_j + W_n n_m) + b_t), \tag{11}$$

where $n_m \sim U_{[0,1]}$ is a sample from a uniform distribution, and W_h and W_n are specified accordingly to the feedforward network mentioned above. The collection of M samples $\{t_{j+1,m}^{\text{pred}}\}_{m=1}^{M}$ can be thought as implicitly sampled from the desired predicted conditional distribution, *i.e.*, $t_{j+1,m}^{\text{pred}} \sim p(t_{j+1}^{pred}|t_{j,m}, h_j)$. This approach effectively allows one to infer the arrival time (conditional) probability density for $t_{j+1,m}^{\text{pred}}$ relative to the observed $t_{j,m}$, which is similar to the approach in Shchur et al. (2020). Note that it is also possible to approximate the conditional intensity $\lambda(t|h_j)$ in (1) from the conditional density $p(t_{j+1}^{pred}|t_{j,m}, h_j)$ (see Shchur et al. (2020) for details).

Since the loss function (defined below) takes the predictions from all of these samples and updates the parameters of the model based on all of them, we are in principle able to approximate the conditional distribution of the arrival times. In practice, if distributional predictions are not a priority, we can simply summarize the M-sample empirical distributions with sample characteristics such as mean, median or standard deviation. In the experiments, we use the sample mean as our final prediction for the arrival times. Figure 7 in the Appendix further illustrates how such stochasticity can be used capture the uncertainty around the arrival time predictions.

Event Type Prediction For multivariate event sequences, predicting the next event type is as important as predicting the arrival times. Unlike the models that use conditional intensity values of each event type, we can specify another feed-forward network that uses the history vectors as input to predict the event types directly. The event prediction is probabilistically defined as follows

$$p(e_{j+1}^{\text{pred}}|h_j) = \text{softmax}(W_e h_j + b_e), \tag{12}$$

where $W_e \in \mathbb{R}^{K \times D}$ and $b_e \in \mathbb{R}^K$ are the weights and the bias of the feed-forward network, h_j is the historical representation of event j defined in (6), and $e_{j+1}^{\text{pred}} = \text{argmax } p(e_{j+1}^{\text{pred}}|h_j) \in (0,1)^K$, *i.e.*, the output of the softmax function that produces event type probabilities that can be converted into predicted event types for time $j+1$.

Loss Function We estimate the underlying distribution of the point process by learning the type embeddings and triggering kernels, and use the next arrival times and event type as the target for the objective function. This allows us to avoid the integral approximations traditionally used in intensity-based decoders Zhang et al. (2020); Zuo et al. (2020); Shchur et al. (2020) and emphasize the predictive ability.

Since the decoders defined above predict both the arrival times and event types, we ought to specify two different loss functions to train the model. Specifically, we use the ℓ_1 loss for the arrival times and the

Table 1: Summary statistics of the datasets used in the experiments. Types and length refer to the events and sequences, respectively.

Dataset	Types	Mean Length	Number of Sequences		
			Training	Validation	Test
2D-Hawkes	2	150	3200	400	400
MIMIC-II	75	4	527	58	65
StackOverflow	22	250	4777	530	1326
Retweet	3	109	20000	2000	2000
COVID-19	5	4	3212	450	500

cross-entropy loss for the event types. The complete loss is defined for the prediction of the next event at each observed timestamp as

$$L_\Phi(S_n) = \sum_j \left(\left| \tilde{t}^{\text{pred}}_{j+1} - t_{j+1} \right| + \sum_{k=1}^K y_{k,j+1} \log(p_{k,j+1}) \right),$$

where $\tilde{t}^{\text{pred}}_{j+1}$ is a sample summary from (11), $p_{k,j+1}$ is the k-th element of $p(e^{\text{pred}}_{j+1} = k|h_j)$ and $y_{k,j+1}$ is the ground truth indicator of whether time $j+1$ is of event type k. Moreover, S_n indicates that the loss is for a single sequence (preventing notation overload), and Φ represents the set of all the parameters needed to be learned for the model. Precisely, the parameters of *i*) the triggering kernel parameter estimator defined in Section 3; *ii*) the arrival time and event time predictors defined in Section 4; *iii*) the scaling parameters of the temporal embeddings in Section 2; and *iv*) the event type embeddings defined in Section 2. Finally, for the minimization problem we use the ADAM (Kingma and Ba, 2015), provided it has proven to be computationally efficient and requiring little tuning compared to alternative algorithms.

5. Related Work

RNN-based approaches have been proposed to model the event sequences. Du et al. (2016) and Mei and Eisner (2017) model the intensity as a non-linear function of the sequence history by using RNNs to summarize the past events and feed-forward decoders. These approaches provide a flexible estimation of the conditional intensity function by avoiding strict assumptions on its functional form. However, the lack of a closed-form solution for the intensity function necessitates computationally expensive approximation techniques such as Monte Carlo to calculate the log-likelihood function values. Shchur et al. (2020) propose to replace the decoder used in the previous models with a log-normal mixture structure. By focusing on the conditional density function for the arrival times, they produce a closed-form solution for the expectation of the process. Their approach avoids expensive approximations and yields efficient sampling of new sequences at the expense of assuming that all arrival times follow a log-normal distribution.

Models with architectures similar to the transformer (Vaswani et al., 2017) encode the historical influence of past events with multiple attention layers and attention heads. Zhang et al. (2020) and Zuo et al. (2020) introduce transformer-based approaches for Hawkes process modeling. These models create history embeddings based on a scoring function that compares the similarity of input vectors for each event in the representation space. Though these attention-based structures can predict arrival times relatively well they still struggle to infer the underlying intensity distribution of the process. Pan et al. (2021) introduces a GP-based approach to non-parametrically model the triggering functions of the point processes. Their work provides similar prediction performance to attention-based models and also captures certain event-type pairing effects, however, it lacks the efficiency of the proposed sigmoid gated approach.

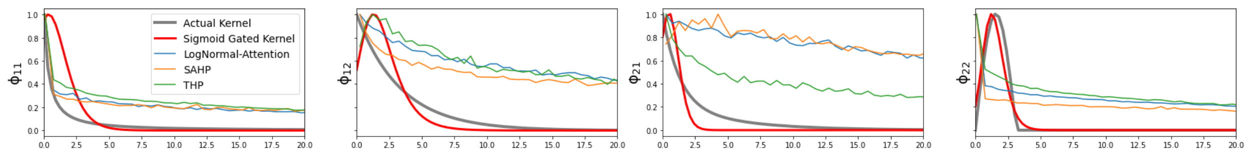

Figure 3: Comparing triggering kernel predictions across models. The x-axis denotes time and the y-axis represents the kernel intensities. The sigmoid-gated kernel is able to learn both the exponential kernel and sinusoidal kernel more accurately.

6. Experiments

We present results on various synthetic and real-world datasets to compare the proposed *sigmoid gated Hawkes process* (SGHP) model against state-of-the-art Hawkes process models. Training details can be found in Appendix C.

Datasets We use a synthetic dataset sampled from a 2D-Hawkes process and three real-word datasets previously used by the baseline models, namely, MIMIC-II, StackOverflow, and Retweet. Further, we consider a COVID-19 inpatient dataset collected at Duke University Health System which further includes patient vitals as covariates. Table 1 provides summary statistics and data splits for all datasets. A detailed explanation of each dataset can be found in Appendix A. Processed datasets and source code are publicly available at `https://github.com/rhenaog/sghp`. The original Stack Overflow (Leskovec and Krevl, 2014), Retwitt (Zhao et al., 2015) and MIMIC-II (Lee et al., 2011) datasets can be found online. The COVID-19 dataset cannot be made openly available due to institutional constraints, but requests for access can be made and evaluated on a case by case basis.

6.1. Baseline Models

We consider two attention-based Hawkes process models (Zhang et al., 2020; Zuo et al., 2020) and an RNN-based log-normal mixture decoder model (Shchur et al., 2020), and a recently introduced GP-based model (Pan et al., 2021). Further, by replacing the RNN encoder in the original log-normal model with attention we introduce a new baseline model.

Self Attentive Hawkes Process (SAHP): An attention-based model that uses multiple layers of multi-headed attention to model the intensity based on historical embeddings (Zhang et al., 2020).

Transformer Hawkes Process (THP): The model by Zuo et al. (2020) is a variant of an attention-based approach similar to SAHP.

Log-Normal Mixture Model (LMM): The log-normal mixture model (Shchur et al., 2020) is an intensity-free approach that models the conditional probability distribution of arrival times directly via an RNN encoder.

LMM with Attention (LMMA): We replace the RNN decoding of the log-normal mixture model with the attention mechanism proposed in the SAHP model. This simple extension of LMM is introduced to understand the performance of attention models with an intensity-free formulation. Note that LMMA is similar to the proposed SGHP in that both use intensity-free formulations, however, SGHP replaces attention with the proposed triggering kernel.

Self-Adaptable Point Processes with Nonparametric Time Decays (SPRITE): This is a GP-based approach for modelling point processes (Pan et al., 2021). It does not utilize an encoder-decoder structure like the other baseline models, but we still include their results on MIMIC, StackOverflow, and Retweet datasets. Since the authors provide last event type prediction results for prediction accuracy instead of F1 scores, we only incorporate their arrival time performance metrics (in RMSE).

Table 2: RMSE and F1 Scores for last event type and arrival time predictions.

	2-D Hawkes		MIMIC-II		StackOverflow		Retweet		COVID	
Model	RMSE	F1	RMSE	F1	RMSE	F1	RMSE	F1	RMSE	F1
LMM	2.25	0.573	0.88	0.892	1.46	0.384	0.61	0.544	341.5	0.90
LMMA	2.23	0.550	0.80	**0.900**	1.49	0.375	0.64	0.528	356.1	0.89
THP	2.42	0.575	0.859	0.877	1.66	**0.411**	0.18	0.539	374.1	**0.93**
SAHP	2.29	0.585	0.821	0.646	1.55	0.242	0.12	0.531	-	0.76
SPRITE	-	-	0.96	-	**1.15**	-	0.09	-	-	-
SGHP	**2.11**	**0.610**	**0.79**	0.810	1.30	0.389	**0.08**	**0.605**	**332.2**	0.89

Table 3: APS for mortality on COVID data with (COVID+v) and without patient vitals.

Dataset	LMM	LMMA	THP	SAHP	SGHP
COVID	**0.18**	0.16	0.07	0.004	0.12
COVID+v	0.12	0.21	0.1	0.005	**0.22**

7. Results

Capturing Triggering Kernels We present the estimated kernels for all the models against the ground-truth triggering functions for the synthetic data and compare the goodness-of-fit to the underlying distribution. Since our approach learns the parameters for each triggering kernel, we can plot the estimation directly as a function of time. For the attention-based models, we use the attention scores for each time difference observed in the data as proxy. We further split these scores across event type pairings. Moreover, to reduce the fluctuations in the attention-based scores, we pass them through a moving average filter where we take mean scores over a small time window ($t = 0.5$). Figure 3 shows the estimates for all models. As expected, the proposed gated kernel (SGHP) is the best match for all of the underlying triggering kernel pairings. Importantly, unlike the other models, it can accommodate to different event type pairings and capture both decaying and local effects.

Sequence Prediction Predicting the next event arrival time is a key objective for event sequences. To capture the ability all models have to learn from the sequential event information, we report the event type and event time results for the last observed event in each sequence. Specifically, we report (micro) F1 scores for event types, except for the COVID data for which a high class imbalance is observed, *i.e.*, 4% mortality. Instead, results for mortality prediction are provided below using average precision score as performance metric. For event arrival times, we report the mean squared error (RMSE). Table 2 show RMSE and F1 scores, respectively, for the last event predictions in the test sets of all datasets and models being evaluated. In terms of RMSE, the proposed SGHP consistently outperforms the other approaches expect the StackOverflow dataset, while in terms of event type prediction, SGHP outperforms the others on the 2-D Hawkes and Retweet datasets and delivers competitive results on MIMIC-II, Stackoverflow and COVID-19. Interestingly, the best performing event type prediction model in MIMIC-II is LMMA, which is also introduced in this paper. We hypothesize that our model's fluctuating event type performance is related to the number of observed events. More specifically, to event incidence imbalances. For instance, the mortality rate in the COVID dataset is 4% and some events in the MIMIC-II and StackOverflow dataset have occurrences in the single digit range. We believe this can be addressed by modifying the event type predictors and leave it for future work. Overall, results in Table 2 show that the proposed SGHP is highly competitive in relation to more complex (multilayer attention and recurrent) alternatives once both event type and arrival time predictions are considered.

Mortality Predictions for Covid-19 Patients Complementing the real-world datasets in Table 2, we also seek to test the proposed model in a more realistic healthcare setting. Specifically, we compare models in terms of their ability to predict patient mortality. We treat the mortality prediction as a binary problem,

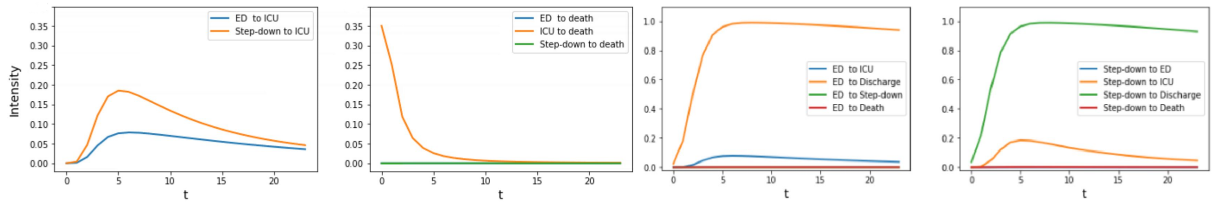

Figure 4: Estimated triggering kernels for various combinations of event type pairs for the COVID-19 data.

e.g., death *vs.* survival. Since there is a significant class imbalance, we report the average precision score (APS) as opposed to F1 or the area under the receiving operating characteristic (see Appendix B for these). Moreover, we incorporate patient vitals as longitudinal covariates as a means to showcase that the model can readily process covariate information. Existing approaches do not consider covariates, but we have incorporated them as described in Appendix D for a fair comparison. Table 3 shows mortality prediction results on both variations of the COVID data (with and without vitals). Without covariates, SGHP delivers average performance and outperforms the other approaches when longitudinal covariates are included.

Model Complexity RNN and attention-based encoders tend to suffer from high complexity and memory requirements due to the large number of parameters that need to be estimated. To highlight the efficiency of our model we calculate the number of estimated parameters for all the prediction tasks. It is important to note that the training speed can also be used as an efficiency metric. However, we argue that the total runtime is more susceptible to the variation across different settings and frameworks such as TensorFlow, PyTorch, *etc.* In contrast, the total number of parameters only depends on the model's structure and is fixed across different frameworks. The average estimated parameters (in thousands) across all datasets are observed as follows, SGHP: **3.1**, SAHP: 4.8, LMM: 16.9, LMMA: 17.8, THP: 417.8, and show that our model requires a significantly smaller number of parameters to deliver comparable or superior performance on all prediction tasks. Table 6 in Appendix B further details the total number of estimated parameters across all the data sets.

Kernels Interpretation for COVID Data The estimated triggering kernels for each event type pairing allows us to interpret the results in terms of intensity functions. Specifically, we report the likelihood of being admitted to ICU and mortality after observing that a patient is admitted to the ED, ICU, or a step-down unit. Figure 4 shows the learned triggering kernels for some of these possible transitions. As expected, the likelihood of mortality is higher for patients already admitted to the ICU, relative to patients in ED beds or step-down units. Moreover, the likelihood of being admitted to the ICU is much higher for patients originating from a step-down unit.

8. Conclusion

We introduced a flexible, efficient, and interpretable approach for modeling Hawkes processes with an encoder-decoder structure. Our approach replaced the complex and more difficult to interpret attention mechanism with directly learning the underlying triggering kernels of the point processes. The proposed encoder uniquely combines a regular decaying kernel with a sigmoid gate, thus allowing it to capture a wide range of triggering functions otherwise not possible with other attention-based approaches. We further generalized our method for the multivariate Hawkes process and incorporated longitudinal covariates into the temporal process characterization.

Limitations Though we showed that the proposed parametric formulation of the triggering kernel is competitive with nonparametric formulations, that may not always be the case, thus identifying practical scenarios where, in principle, more general nonparametric specifications will be of interest. Further, additional work is needed to better understand and leverage the availability of longitudinal covariates to improve sequence and event type predictions. Finally, though we provided proof-of-principle interpretation of the estimated triggering kernels on the COVID-19 data, comprehensive quantitative evaluation will be needed if such interpretations were to be used to impact clinical care.

Acknowledgments

The authors would like to thank the anonymous reviewers for their insightful comments and the Duke Institute for Health Innovation (DIHI) for providing access to the COVID-19 data. This research was supported by NIH/NINDS 1R61NS120246, NIH/NIDDK R01- DK123062, and ONR N00014-18-1-2871-P00002-3.

References

Hamidreza Alvari and Paulo Shakarian. Hawkes process for understanding the influence of pathogenic social media accounts. In *International Conference on Data Intelligence and Security*, 2019.

Emmanuel Bacry, Iacopo Mastromatteo, and Jean-François Muzy. Hawkes processes in finance. *Market Microstructure and Liquidity*, 2015.

Yujia Bao, Zhaobin Kuang, Peggy Peissig, David Page, and Rebecca Willett. Hawkes process modeling of adverse drug reactions with longitudinal observational data. In *Machine Learning for Healthcare Conference*, 2017.

Feng Chen and Peter Hall. Nonparametric estimation for self-exciting point processes—a parsimonious approach. *Journal of Computational and Graphical Statistics*, 2016.

Wen-Hao Chiang, Xueying Liu, and George Mohler. Hawkes process modeling of covid-19 with mobility leading indicators and spatial covariates. *International Journal of Forecasting*, 2021.

D.R. Cox and V. Isham. *Point Processes*, pages 5–20. Chapman & Hall/CRC Monographs on Statistics & Applied Probability. Taylor & Francis, 1980.

Nan Du, Hanjun Dai, Rakshit Trivedi, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and Le Song. Recurrent marked temporal point processes: Embedding event history to vector. In *International Conference on Knowledge Discovery and Data Mining*, 2016.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, 2014.

Rameshwar D Gupta and Debasis Kundu. Generalized logistic distributions. *Journal of Applied Statistical Science*, 2010.

Vinayak Gupta and Srikanta Bedathur. Proactive: Self-attentive temporal point process flows for activity sequences. *arXiv preprint arXiv:2206.05291*, 2022.

Vinayak Gupta, Srikanta Bedathur, Sourangshu Bhattacharya, and Abir De. Modeling continuous time sequences with intermittent observations using marked temporal point processes. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2022.

Alan G. Hawkes. Spectra of some self-exciting and mutually exciting. *Biometrika*, 1971.

M.A. Hearst, S.T. Dumais, E. Osuna, J. Platt, and B. Scholkopf. Support vector machines. *IEEE Intelligent Systems and their Applications*, 1998.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Joon Lee, Daniel J Scott, Mauricio Villarroel, Gari D Clifford, Mohammed Saeed, and Roger G Mark. Open-access mimic-ii database for intensive care research. In *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 8315–8318. IEEE, 2011.

Kyungsub Lee and Byoung Ki Seo. Marked hawkes process modeling of price dynamics and volatility estimation. *Journal of Empirical Finance*, 2017.

Jure Leskovec and Andrej Krevl. Snap datasets: Stanford large network dataset collection, 2014.

Erik Lewis and George Mohler. A nonparametric em algorithm for multiscale hawkes processes. *Journal of Nonparametric Statistics*, 2011.

Hongyuan Mei and Jason M Eisner. The neural hawkes process: A neurally self-modulating multivariate point process. In *Advances in Neural Information Processing Systems*, 2017.

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

Takahiro Omi, naonori ueda, and Kazuyuki Aihara. Fully neural network based model for general temporal point processes. In *Advances in Neural Information Processing Systems*, 2019.

Tohru Ozaki. Maximum likelihood estimation of hawkes' self-exciting point processes. *Annals of the Institute of Statistical Mathematics*, 1979.

Zhimeng Pan, Zheng Wang, Jeff Phillips, and Shandian Zhe. Self-adaptable point processes with nonparametric time decays. In *Advances in Neural Information Processing Systems*, 2021.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, 2013.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*, pages 86–87. The MIT Press, 2005.

Marian-Andrei Rizoiu, Young Lee, Swapnil Mishra, and Lexing Xie. A tutorial on hawkes processes for events in social media. *arXiv preprint arXiv:1708.06401*, 2017.

Oleksandr Shchur, Marin Biloš, and Stephan Günnemann. Intensity-free learning of temporal point processes. In *International Conference on Learning Representations*, 2020.

Christian Steinruecken, Emma Smith, David Janz, James Lloyd, and Zoubin Ghahramani. *The Automatic Statistician*, pages 161–173. 2019.

Zhaohong Sun, Zhoujian Sun, Wei Dong, Jinlong Shi, and Zhengxing Huang. Towards predictive analysis on disease progression: A variational hawkes process model. *IEEE Journal of Biomedical and Health Informatics*, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

Steve Y. Yang, Anqi Liu, Jing Chen, and Alan Hawkes. Applications of a multivariate hawkes process to joint modeling of sentiment and market return events. *Quantitative Finance*, 2018.

Lu-ning Zhang, Jian-wei Liu, Zhi-yan Song, and Xin Zuo. Temporal attention augmented transformer hawkes process. *Neural Computing and Applications*, 2022.

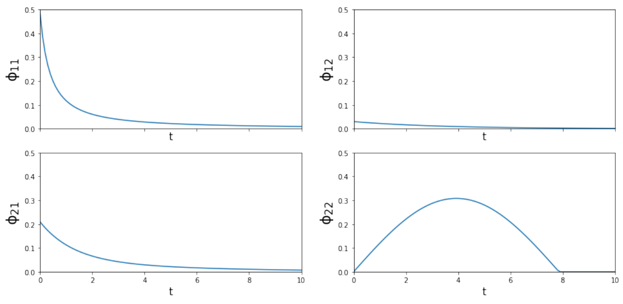

Figure 5: Triggering functions for all event type combinations in the synthetic dataset.

Table 4: AUROC results for mortality prediction on COVID data with and without patient vitals (COVID+v).

Dataset	LMM	LMMA	THP	SAHP	SGHP
COVID	0.752	**0.801**	0.450	0.448	0.800
COVID+v	0.790	**0.880**	0.745	0.482	0.840

Qiang Zhang, Aldo Lipani, Omer Kirnap, and Emine Yilmaz. Self-attentive Hawkes process. In *International Conference on Machine Learning*, 2020.

Qingyuan Zhao, Murat A Erdogdu, Hera Y He, Anand Rajaraman, and Jure Leskovec. Seismic: A self-exciting point process model for predicting tweet popularity. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1513–1522, 2015.

Simiao Zuo, Haoming Jiang, Zichong Li, Tuo Zhao, and Hongyuan Zha. Transformer Hawkes process. In *International Conference on Machine Learning*, 2020.

Appendix A. Dataset Details

A.1. Synthetic Dataset

We use the same 2D Hawkes dataset used by in SAHP (Zhang et al., 2020). For reproducibility, we resampled the dataset from the tick library with a fixed random seed. Below we provide the formulation for each triggering function in the synthetic dataset. The baseline intensities are $\mu_1 = 0.1$ and $\mu_2 = 0.2$. There are three regular decaying kernels and a sinusoidal kernel with a local effect. Figure 7 illustrates each triggering kernel.

$$\begin{aligned}
\phi_{11}(t) &= 0.2 \times t(0.5+t)^{-1.3} \\
\phi_{12}(t) &= 0.03 \times e^{-0.3t} \\
\phi_{12}(t) &= 0.05 \times e^{-0.2t} + 0.16 \times e^{-0.8t} \\
\phi_{22}(t) &= \max\big(0, \sin(t)/8\big) \text{ for } 0 \leq t \leq 4
\end{aligned} \tag{13}$$

A.2. Real World Data Sets

Stackoverflow (SOF): The data set consists of users of the famous question-and-answer website Stack Overflow. Users get medals based on their questions or answers, *e.g.*, good question, good answer, *etc.* Each user is modeled as a sequence with event types representing medals obtained over time.

Table 5: F1 results for last event type prediction for the COVID data with and without patient vitals (COVID+v).

Dataset	LMM	LMMA	THP	SAHP	SGHP
COVID	0.901	0.888	**0.932**	0.762	0.888
COVID+v	0.886	0.901	**0.921**	0.793	0.893

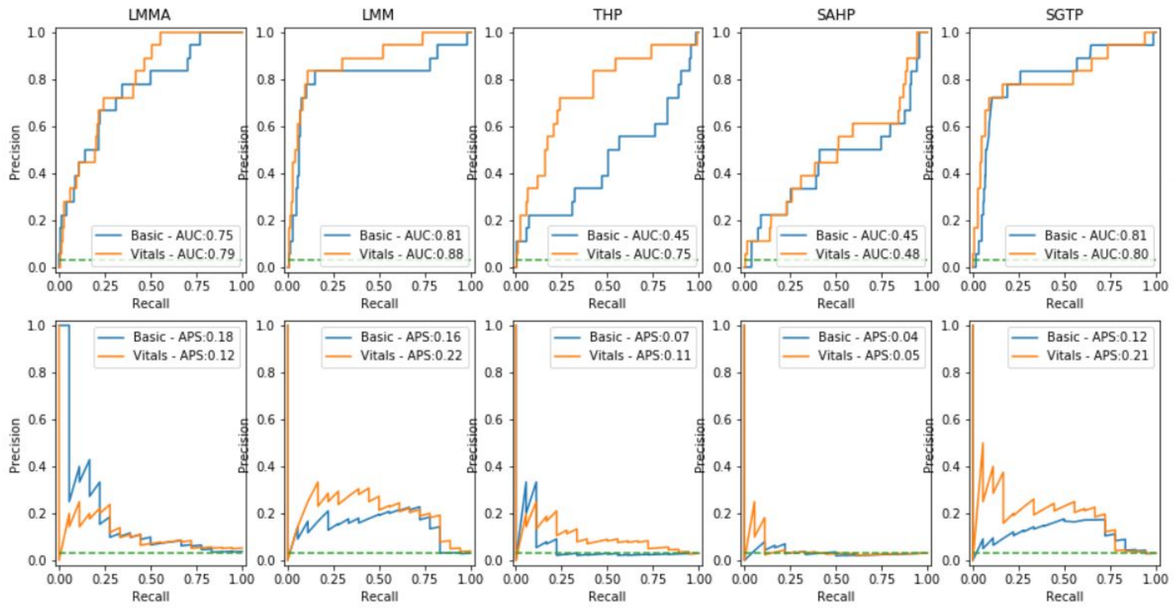

Figure 6: ROC and Precision-Recall curves for mortality prediction using the COVID-19 dataset.

Mimic-II (MMC): This electronic healthcare records dataset tracks the journey of patients through their hospital intensive care unit (ICU) stay over a period of seven years. Each patient sequence consists of timestamps and diagnoses for each visit.

Retweets (RT): The dataset tracks re-tweets made on particular tweets on the social media website Twitter. Every time a tweet is re-tweeted, the popularity of the re-tweeter and the timestamp of the retweet are recorded. There are three different re-tweeter categories based on their number of followers. Due to the wide range of the inter-arrival times, arrival time results are made relative by scaling over the maximum inter-arrival time.

COVID-19 Emergency Department (COVID): This dataset tracks the journey of COVID-19 patients admitted for patient care at Duke University Hospital Emergency Department (ED) between January of 2020 and December of 2021. We consider five different event types, namely, admission to ED for patient care, admission to ICU, admission to a step-down unit, discharge, and death. Further, the dataset also includes vitals measured between events. Specifically: pulse oximetry (SpO2), mean arterial pressure (MAP), blood pressure (BP), temperature, and respiratory rate (RR).

Table 6: Total number of learned parameters (in thousands).

Dataset	LMM	LMMA	THP	SAHP	SGHP
2-D Hawkes	17.2	17.2	25.3	4.1	**2.7**
MIMIC-II	17.8	19.6	189	6.5	**4.2**
StackOverflow	15.7	17.8	1595	4.8	**3.7**
Retweet	14.8	17.2	41.7	4.2	**2.4**
COVID-19	14.9	17.3	23.7	4.2	**3.8**
Average	16.9	17.8	417.8	4.8	**3.1**

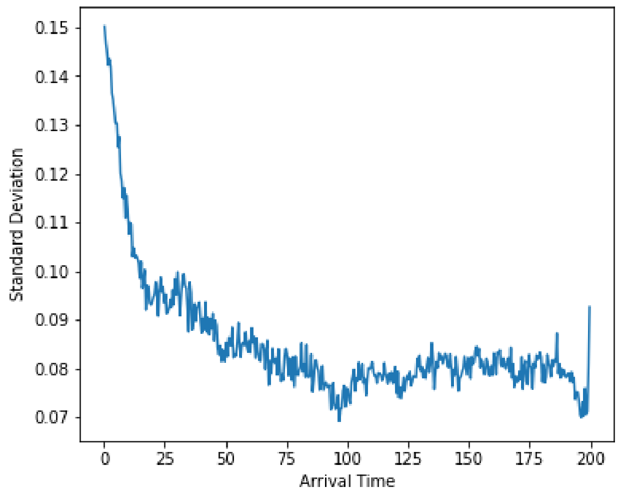

Figure 7: The standard deviation of the predicted arrival times as a function of the observed history length.

Appendix B. Additional Results

B.1. Covid-19 Emergency Department Predictions

Tables 4 and 5 present additional results for mortality and event type predictions on the COVID-19 dataset, namely area under the receiving operating characteristic (AUROC) and F1 scores, respectively. Further, Figure 6 shows the ROC and Precision-Recall Curves for the same task.

B.2. Total Number of Estimated Parameters

The efficiency of the proposed model is highlighted in Section 6. Table 6 further provides the the number of total estimated parameters across all datasets. As expected, SGHP requires significantly lower number of parameters across all prediction tasks.

B.3. Uncertainty Around the Predictions

Equipped with a stochastic prediction layer, our model can also provide uncertainty around the arrival time predictions via sample standard deviations. Figure 7 shows how the uncertainty, quantified as the average standard deviation of the predicted arrival times, changes as a function of the history length. As expected, observing longer history allows our model to encode richer history embeddings which results in lower uncertainty around our predictions.

Table 7: Hyperparameters used for training across all datasets.

Dataset	Epochs	Batch Size	H_d	Learning Rate	Sample Size
2-D Hawkes	100	16	16	0.0001	50
MIMIC-II	1000	32	32	0.0001	50
StackOverflow	250	32	16	0.0001	50
Retweet	250	25	16	0.0001	50
COVID-19	100	32	32	0.0001	50

Appendix C. Training Details

We train all the models on an NVIDIA Tesla P100 (16GB). Table 7 also provides details on hyper parameters such as number of epochs, batch size, size of the input vector, learning rate and number of samples used in the stochastic layer.

Appendix D. Longitudinal Covariates

We generalize the input vector to include the addition of longitudinal covariates. In order to inject the information from the covariates, we create a linear transformation that outputs a covariate embedding u_i^{enc} of dimensionality D and which is consistent with the dimension of the temporal and event type embeddings. We modify the concatenated embedding vector as follows

$$x_i = [t_i^{\text{enc}} + e_i^{\text{enc}} | \; u_i^{\text{enc}}], \tag{14}$$

where u_i^{enc} follows from

$$u_i^{\text{enc}} = W_u z_i + b_u, \tag{15}$$

and z_i is the vector of covariates for and event at time t_i, W_u and b_u are weights and the bias of the linear layer, respectively. This structure can be generalized to all encoder based models, and is used for the COVID-19 prediction results with patient vitals.