Learning Temporal Point Processes with Intermittent Observations

Vinayak Gupta IIT Delhi
Srikanta Bedathur IIT Delhi
Sourangshu Bhattacharya IIT Kharagpur
Abir De IIT Bombay
{vinayak.gupta, srikanta}@cse.iitd.ac.in sourangshu@cse.iitkgp.ac.in abir@cse.iitb.ac.in

Abstract

Marked temporal point processes (MTPP) have emerged as a powerful framework to model the underlying generative mechanism of asynchronous events localized in continuous time. Most existing models and inference methods in MTPP framework consider only the complete observation scenario *i.e.* the event sequence being modeled is completely observed with no missing events – an ideal setting barely encountered in practice. A recent line of work which considers missing events uses supervised learning techniques which require a *missing* or *observed* label for each event. In this work, we provide a novel unsupervised model and inference method for MTPPs in presence of missing events. We first model the generative processes of observed events and missing events using two MTPPs, where the missing events are represented as latent random variables. Then we devise an unsupervised training method that jointly learns both the MTPPs by means of variational inference. Experiments with real datasets show that our modeling and inference frameworks can effectively impute the missing data among the observed events, which in turn enhances its predictive prowess.

1 Introduction

In recent years, marked temporal point processes (MTPPs) (Valera et al., 2014; Rizoiu et al., 2017; Wang et al., 2017; Daley and Vere-Jones, 2007) have shown an outstanding potential to characterize asynchronous events localized in continuous time that appear in a wide range of applications in healthcare (Lorch et al., 2018; Rizoiu et al., 2018), traffic (Du et al., 2016; Guo et al., 2018), web and social networks (Valera et al., 2014; Du et al., 2015; Tabibian et al., 2019; Kumar et al., 2019; De et al., 2016; Du et al., 2016; Farajtabar et al., 2017; Jing and Smola, 2017; Likhyani et al., 2020), finance (Bacry et al., 2015) and many more. In MTPP, an event is represented using two quantities: (i) the time of its occurrence and (ii) the associated mark, where the latter indicates the category of the event and therefore bears different meaning for different applications. For example, in a social network setting, the marks may indicate users' likes, topics and opinions of the posts; in finance, they may correspond to the stock prices and the amount of sales; in healthcare, they may indicate the state of the disease of an individual.

In this context, most of the MTPP models (Valera et al., 2014; Wang et al., 2017; Huang et al., 2019; Du et al., 2016; Zuo et al., 2020; Zhang et al., 2020)— with a few recent exceptions (Shelton et al., 2018; Mei et al., 2019)— have considered only the settings where the training data is completely observed or, in other words, there is no missing observation at all. While working with fully observed data is ideal for understanding any dynamical system, this is not possible in many practical scenarios. We may miss observing events due to constraints such as crawling restrictions by social media platforms, privacy restrictions (certain users may disallow collection of certain types of data), budgetary factors such as data collection for exit polls, or other practical factors e.g. a patient may not be available at a certain time. This results in poor predictive performance of MTPP models (Du et al., 2016; Zuo et al., 2020; Zhang et al., 2020) that skirt this issue.

Statistical analysis in presence of missing data have been widely researched in literature in various contexts (Che et al., 2018; Yoon et al., 2019; Tian et al., 2018; Śmieja et al., 2018). Little and Rubin (2019) offers a comprehensive survey. It provides three models that capture data missing mechanisms in the increasing or-

Proceedings of the 24[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130.

der of complexity, *viz.*, MCAR (missing completely at random), MAR (missing at random) and MNAR (missing not at random). Recently, Shelton et al. (2018) and Mei et al. (2019) proposed novel methods to impute missing events in MTPPs from the viewpoint of MNAR mechanism. However, they focus on imputing missing data in between a-priori available observed events, rather than predicting observed events in the face of missing events. Moreover, they deploy expensive learning and sampling mechanisms, which make them often intractable in practice, especially in case of learning from a sequence of streaming events. For example, Shelton et al. (2018) apply expensive MCMC sampling procedure to draw missing events between the observation pairs, which requires several simulations of the sampling procedure upon arrival of a new sample. On the other hand Mei et al. (2019) uses bidirectional RNN which re-generates all missing events by making a completely new pass over the backward RNN, whenever one new observation arrives. As a consequence, it suffers from the quadratic complexity with respect to the number of observed events. On the other hand, the proposal of Shelton et al. (2018) depends on a pre-defined influence structure among the underlying events, which is available in linear multivariate parameterized point processes. In more complex point processes, such a structure is not explicit, which limits their applicability in practice.

1.1 Present work

In this work, we overcome the above limitations using a novel modeling framework for point processes called **IMTPP** (**I**ntermittently-observed **M**arked **T**emporal **P**oint **P**rocesses), which characterizes the dynamics of both observed and missing events as two coupled MTPPs, conditioned on the history of previous events. In our setup, the generation of missing events depends both on the previously occurred missing events as well as the previously occurred observed events. Therefore, they are MNAR (missing not at random), in the context of the literature of missing data (Little and Rubin, 2019). In contrast to the prior models (Mei et al., 2019; Shelton et al., 2018), IMTPP aims to learn the dynamics of both observed and missing events, rather than imputing missing events in between the known observed events, which is reflected in its superior predictive power over those existing models.

Precisely, IMTPP represents the missing events as latent random variables, which together with the previously observed events, seed the generative processes of the subsequent observed and missing events. Then it deploys three generative models— MTPP for observed events, prior MTPP for missing events and posterior MTPP for missing events, using recurrent neural networks (RNN) that capture the nonlinear influence of the past events. To do so, it enjoys several technical innovations, which significantly boosts its efficiency as well as predictive accuracy.
(1) In a marked departure from almost all existing MTPP models (Du et al., 2016; Tabibian et al., 2019; Mei et al., 2019; De et al., 2016) which rely strongly on conditional intensity functions, we use a log-normal distribution to sample arrival times of the events. As suggested by Shchur et al. (2020), such a distribution allows efficient sampling as well as more accurate prediction than the intensity-based models.
(2) The built-in RNNs in our model can only make forward computations. Therefore, they incrementally update the dynamics upon the arrival of a new observation. Consequently, unlike the prior models, it does not require to re-generate all the missing events responding to the arrival of an observation, which significantly boosts the efficiency of both learning and prediction as compared to both the previous work (Mei et al., 2019; Shelton et al., 2018).

Our modeling framework allows us to train IMTPP using efficient variational inference method, that maximizes the evidence lower bound (ELBO) of the likelihood of the observed events. Such a formulation connects our model with the variational autoencoders (VAEs) (Chung et al., 2015; Bowman et al., 2015). However, in a sharp contrast to traditional VAEs, where the random noises or seeds often do not have immediate interpretations, our random variables bear concrete physical explanations *i.e.* they are missing events, which renders our model more explainable than an off-the-shelf VAE.

Finally, our experiments with six diverse real datasets show that IMTPP can model missing observations, within a stream of observed events, and enhances the predictive power of the original generative process for a full observation scenario.

2 Problem setup

In this section, we first introduce the notations and then setup of our problem of learning marked temporal point process with missing events.

2.1 Preliminaries and notations

A marked temporal point process (MTPP) is a stochastic process whose realization consists of a sequence of discrete events $\mathcal{S}_k = \{e_i = (x_i, t_i) | i \in [k], t_i < t_{i+1}\}$, where $t_i \in \mathbb{R}^+$ is the time of occurrence and $x_i \in \mathcal{C}$ is a discrete mark of the i-th observed event that occurred at time t_i, with $\mathcal{C}$ to be the set of discrete marks. Here, $\mathcal{S}_k$ denotes the set of first k

observed events. We denote the inter-arrival times of the observed events as, $\Delta_{t,k} = t_k - t_{k-1}$.

However in reality, there may be instances where an event has actually taken place, but not recorded in $\mathcal{S}$. To this end, we introduce the *MTPP for missing events*— a latent MTPP— which is characterized by a sequence of hidden events $\mathcal{M}_r = \{\epsilon_j = (y_j, \tau_j) | j \in [r], \tau_j < \tau_{j+1}\}$ where $\tau_j \in \mathbb{R}^+$ and $y_j \in \mathcal{C}$ are the times and the marks of the j-th missing events. Therefore, $\mathcal{M}_r$ defines the set of first r missing events. Moreover, we denote the inter-arrival times of the missing events as, $\Delta_{\tau,r} = \tau_r - \tau_{r-1}$. Note that $\tau_\bullet$, $y_\bullet$, $\mathcal{M}_\bullet$ and $\Delta_{\tau,\bullet}$ for the MTPP of missing events share similar meanings with $t_\bullet$, $x_\bullet$, $\mathcal{S}_\bullet$ and $\Delta_{t,\bullet}$ respectively for the MTPP of observed events. We further define two quantities $\underline{k}$ and $\overline{k}$ as follows:

$$\underline{k} = \operatorname*{argmin}_r \{\tau_r \,|\, t_k < \tau_r < t_{k+1}\} \tag{1}$$

$$\overline{k} = \operatorname*{argmax}_r \{\tau_r \,|\, t_k < \tau_r < t_{k+1}\} \tag{2}$$

Here, $\underline{k}$ and $\overline{k}$ are the indices of the first and the last missing events respectively, among those which have arrived between k-th and $k+1$-th observed events. Figure 1 (a) illustrates our setup.

In practice, the arrival times (t and τ) of both observed and missing events are continuous random variables, whereas the marks (x and y) are discrete random variables. Therefore, following the state-of-the-art MTPP models, we model density function to draw timings and probability mass function to draw marks, which in turn induce a density function characterizing the generative process.

2.2 Our distinctive goal

Our goal in this paper is to design an MTPP model which can generate the subsequent observed (e_{k+1}) and missing events (ϵ_{r+1}), in a recursive manner, conditioned on the history of events $\mathcal{S}_k \cup \mathcal{M}_r$ that have occurred thus far. Given the input sequence of observations $\mathcal{S}_K$ consisting of first K observed events $\{e_1, e_2, ..., e_K\}$, we first train our generative model and then predict the next observed event e_{K+1} [1].

3 Components of IMTPP

At the very outset, IMTPP, our proposed generative model, connects two stochastic processes— one for the observed events, which samples the observed and the other for the missing events— based on the history of previously generated missing and observed events. Note, that the sequence of training events that is given as input to IMTPP consists of only the observed events. We model the missing event sequence through latent random variables, which, along with the previously observed events, drive a unified generative model for the complete (observed and missing) event sequence.

Given a stream of observed events $\mathcal{S}_K = \{e_1 = (x_1, t_1), e_2 = (x_2, t_2), \ldots, e_K = (x_K, t_K)\}$, if we use the maximum likelihood principle to train IMTPP, then we should maximize the marginal log-likelihood of the observed stream of events, *i.e.*, $\log p(\mathcal{S}_K)$. However, computation of $\log p(\mathcal{S}_K)$ demands marginalization with respect to the set of latent missing events $\mathcal{M}_{\overline{K-1}}$, which is typically intractable. Therefore, we resort to maximizing a variational lower bound or evidence lower bound (ELBO) of the log-likelihood of the observed stream of events $\mathcal{S}_K$. More specifically, we note that:

$$p(\mathcal{S}_K) = \prod_{k=0}^{K-1} \int_{\mathcal{M}_{\overline{k}}} p(e_{k+1} \,|\, \mathcal{S}_k, \mathcal{M}_{\overline{k}})\, p(\mathcal{M}_{\overline{k}})\, d\omega(\mathcal{M}_{\overline{k}})$$

$$= \mathbb{E}_{q(\mathcal{M}_{\overline{K-1}} \,|\, \mathcal{S}_K)} \prod_{k=0}^{K-1} \frac{p(e_{k+1} \,|\, \mathcal{S}_k, \mathcal{M}_{\overline{k}}) \prod_{r=\underline{k}}^{\overline{k}} p(\epsilon_r \,|\, \mathcal{S}_k, \mathcal{M}_{r-1})}{\prod_{r=\underline{k}}^{\overline{k}} q(\epsilon_r \,|\, e_{k+1}, \mathcal{S}_k, \mathcal{M}_{r-1})}$$

where, $\omega(\mathcal{M})$ is the measure of the set $\mathcal{M}$, q is an approximate posterior distribution which aims to interpolate missing events ϵ_r within the interval (t_k, t_{k+1}), based on the knowledge of the next observed event e_k, along with all previous events $\mathcal{S}_k \cup \mathcal{M}_{r-1}$. Recall that $\underline{k}$ ($\overline{k}$) is the index r of the first (last) missing event ϵ_r among those which have arrived between k-th and $k+1$-th observed events, *i.e.*, $\underline{k} = \operatorname{argmin}_r \{\tau_r \,|\, t_k < \tau_r < t_{k+1}\}$ and $\overline{k} = \operatorname{argmax}_r \{\tau_r \,|\, t_k < \tau_r < t_{k+1}\}$. Next, by Jensen inequality, $\log p(\mathcal{S}_K)$ is at-least

$$\mathbb{E}_{q(\mathcal{M}_{\overline{K-1}} \,|\, \mathcal{S}_K)} \sum_{k=0}^{K-1} \log p(e_{k+1} \,|\, \mathcal{S}_k, \mathcal{M}_{\overline{k}})$$
$$- \sum_{k=0}^{K-1} \sum_{r=\underline{k}}^{\overline{k}} \mathrm{KL}\Big[q(\epsilon_r \,|\, e_{k+1}, \mathcal{S}_k, \mathcal{M}_{r-1}) || p(\epsilon_r \,|\, \mathcal{S}_k, \mathcal{M}_{r-1})\Big], \tag{3}$$

While the above inequality holds for any distribution q, the quality of this lower bound depends on the expressivity of q, which we would model using a deep recurrent neural network. Moreover, the above lower bound suggests that our model consists of the following components.

MTPP for observed events. The distribution $p(e_{k+1} \,|\, \mathcal{S}_k, \mathcal{M}_{\overline{k}})$ models the MTPP for observed events, which generates the $(k+1)$-th event, e_{k+1}, based on the history of all k observed events $\mathcal{S}_k$ and all missing events $\mathcal{M}_{\overline{k}}$ generated so far.

[1] we can also predict the missing events but we evaluate the predictive performance only on observed events, since the missing events are not available in practice.

Prior MTPP for missing events. The distribution $p(\epsilon_r \,|\, \mathcal{S}_k, \mathcal{M}_{r-1})$ is the prior model of the MTPP for missing events. It generates the r-th missing event ϵ_r after the observed event e_k, based on the prior information— the history with all k observed events $\mathcal{S}_k$ and all missing events $\mathcal{M}_{r-1}$ generated so far.

Posterior MTPP for missing events. Given the set of observed events $\mathcal{S}_{k+1} = \{e_1, e_2, \ldots, e_{k+1}\}$, the distribution $q(\epsilon_r \,|\, e_{k+1}, \mathcal{S}_k, \mathcal{M}_{r-1})$ generates the r-th missing event ϵ_r, after the knowledge of the subsequent observed event e_{k+1} is taken into account, along with information about all previously observed events $\mathcal{S}_k$ and all missing events $\mathcal{M}_{r-1}$ generated so far.

4 Architecture of IMTPP

We first present a high level overview of deep neural network parameterization of different components of IMTPP model, and then describe component-wise architecture in detail. Finally, we briefly present the salient features of our proposal.

4.1 High level overview

We approximate the *MTPP for observed events*, $p(e_{k+1} \,|\, \mathcal{S}_k, \mathcal{M}_{\overline{k}})$ using p_θ and the *posterior MTPP for missing events* $q(\epsilon_r \,|\, e_{k+1}, \mathcal{S}_k, \mathcal{M}_{r-1})$ using q_ϕ, both implemented as neural networks with parameters θ and ϕ respectively. We set the *prior MTPP for missing events* $p(\epsilon_r \,|\, \mathcal{S}_k, \mathcal{M}_{r-1})$ as a known distribution p_{prior} using the history of all the events it is conditioned on. In this context, we design two recurrent neural networks (RNNs) which embed the history of observed events $\mathcal{S}$ into the hidden vectors $\boldsymbol{s}$ and the missing events $\mathcal{M}$ into the hidden vector $\boldsymbol{m}$, similar to several state-of-the art MTPP models (Du et al., 2016; Mei and Eisner, 2017; Mei et al., 2019). In particular, the embeddings $\boldsymbol{s}_k$ and $\boldsymbol{m}_r$ encode the influence of the arrival time and the mark of the first k observed events from $\mathcal{S}_k$ and first r missing events from $\mathcal{M}_r$ respectively. Here, the RNN for the observed events updates $\boldsymbol{s}_k$ to $\boldsymbol{s}_{k+1}$ by incorporating the effect of e_{k+1}. Similarly, the RNN for the missing events updates $\boldsymbol{m}_r$ to $\boldsymbol{m}_{r+1}$ by taking into account of the event ϵ_{r+1}.

Each event has two components, its *mark* and the *arrival-time*, which are discrete and continuous random variables respectively. Hence, we characterize the event distribution as a density function which is the product of the density function $(p_{\theta,\Delta}, q_{\phi,\Delta}, p_{\text{prior},\Delta})$ of the inter-arrival time and the probability distribution $(\mathbb{P}_{\theta,x}, \mathbb{Q}_{\phi,y}, \mathbb{P}_{\text{prior},y})$ of the mark, *i.e.*,

$$\begin{aligned} &p_\theta(e_{k+1} = (x_{k+1}, t_{k+1}) \,|\, \mathcal{S}_k, \mathcal{M}_{\overline{k}}) \\ &\quad = \mathbb{P}_{\theta,x}(x_{k+1} \,|\, \Delta_{t,k+1}, \boldsymbol{s}_k, \boldsymbol{m}_{\overline{k}}) \\ &\qquad \cdot p_{\theta,\Delta}(\Delta_{t,k+1} \,|\, \boldsymbol{s}_k, \boldsymbol{m}_{\overline{k}}) \qquad (4) \\ &q_\phi(\epsilon_r = (y_r, \tau_r) \,|\, e_{k+1}, \mathcal{S}_k, \mathcal{M}_{r-1}) \\ &\quad = \mathbb{Q}_{\phi,y}(y_r \,|\, \Delta_{\tau,r}, e_{k+1}, \boldsymbol{s}_k, \boldsymbol{m}_{r-1}) \cdot \\ &\qquad q_{\phi,\Delta}(\Delta_{\tau,r} \,|\, e_{k+1}, \boldsymbol{s}_k, \boldsymbol{m}_{r-1}) \qquad (5) \\ &p_{\text{prior}}(\epsilon_r = (y_r, \tau_r) \,|\, \mathcal{S}_k, \mathcal{M}_{r-1}) \\ &\quad = \mathbb{P}_{\text{prior},y}(y_r \,|\, \Delta_{\tau,r}, \boldsymbol{s}_k, \boldsymbol{m}_{r-1}) \cdot \\ &\qquad p_{\text{prior},\tau}(\Delta_{\tau,r} \,|\, \boldsymbol{s}_k, \boldsymbol{m}_{r-1}) \qquad (6) \end{aligned}$$

where, as mentioned, the inter-arrival times $\Delta_{t,k}$ and $\Delta_{\tau,r}$ are given as $\Delta_{t,k} = t_k - t_{k-1}$ and $\Delta_r = \tau_r - \tau_{r-1}$. Moreover, $p_{\theta,\Delta}$, $q_{\phi,\Delta}$ and $p_{\text{prior},\Delta}$ denote the density of the inter-arrival times for the observed events, posterior density and the prior density of the inter-arrival times of the missing events; and, $\mathbb{P}_{\theta,x}$, $\mathbb{Q}_{\phi,y}$ and $\mathbb{P}_{\text{prior},y}$ denote the corresponding probability mass functions of the mark distributions. Panels (b) and (c) in Figure 1 illustrate the neural architecture of IMTPP.

4.2 Parameterization of p_θ

We realize p_θ in Eq. 4 using a three layer architecture.

Input layer. The first level is the input layer, which takes the last event as input and represents it through a suitable vector. In particular, upon arrival of e_k, it computes the corresponding vector $\boldsymbol{v}_k$ as:

$$\boldsymbol{v}_k = \boldsymbol{w}_{t,v} t_k + \boldsymbol{w}_{x,v} x_k + \boldsymbol{w}_{t,\Delta}(t_k - t_{k-1}) + \boldsymbol{a}_v, \quad (7)$$

where $\boldsymbol{w}_{\bullet,\bullet}$ and $\boldsymbol{a}_v$ are trainable parameters.

Hidden layer. The next level is the hidden layer that embeds the sequence of observations into finite dimensional vectors $\boldsymbol{s}_\bullet$, computed using RNN. Such a layer takes $\boldsymbol{v}_i$ as input and feed it into an RNN to update its hidden states in the following way.

$$\boldsymbol{s}_k = \tanh(\boldsymbol{W}_{s,s}\boldsymbol{s}_{k-1} + \boldsymbol{W}_{s,v}\boldsymbol{v}_k + (t_k - t_{k-1})\boldsymbol{w}_{s,k} + \boldsymbol{a}_s)$$

where $\boldsymbol{W}_{s,\bullet}$ and $\boldsymbol{a}_s$ are trainable parameters. This hidden state $\boldsymbol{s}_k$ can also be considered as a sufficient statistic of $\mathcal{S}_k$, the sequence of the first k observations.

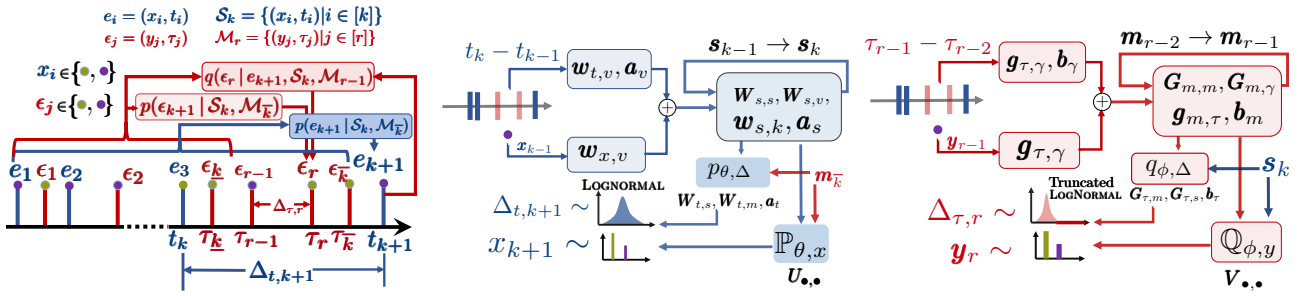
Output layer. The next level is the output layer which computes both $p_{\theta,\Delta}(\cdot)$ and $\mathbb{P}_{\theta,x}(\cdot)$ based on $\boldsymbol{s}_k$ and $\boldsymbol{m}_{\overline{k}}$. To this end, we have the density of inter-arrival times as

$$\begin{aligned} &p_{\theta,\Delta}(\Delta_{t,k+1} \,|\, \boldsymbol{s}_k, \boldsymbol{m}_{\overline{k}}) \\ &\quad = \textsc{Lognormal}\left(\mu_e(\boldsymbol{s}_k, \boldsymbol{m}_{\overline{k}}), \sigma_e^2(\boldsymbol{s}_k, \boldsymbol{m}_{\overline{k}})\right), \qquad (8) \end{aligned}$$

with $[\mu_e(\boldsymbol{s}_k, \boldsymbol{m}_{\overline{k}}), \sigma_e(\boldsymbol{s}_k, \boldsymbol{m}_{\overline{k}})] = \boldsymbol{W}_{t,s}^\top \boldsymbol{s}_k + \boldsymbol{W}_{t,m}^\top \boldsymbol{m}_{\overline{k}} + \boldsymbol{a}_t$; and, the mark distribution as,

$$\begin{aligned} &\mathbb{P}_{\theta,x}(x_{k+1} = x \,|\, \Delta_{t,k+1}, \boldsymbol{s}_k, \boldsymbol{m}_{\overline{k}}) \\ &\quad = \frac{\exp(\boldsymbol{U}_{x,s}^\top \boldsymbol{s}_k + \boldsymbol{U}_{x,m}^\top \boldsymbol{m}_{\overline{k}})}{\sum_{x' \in \mathcal{C}} \exp(\boldsymbol{U}_{x',s}^\top \boldsymbol{s}_k + \boldsymbol{U}_{x',m}^\top \boldsymbol{m}_{\overline{k}})}, \qquad (9) \end{aligned}$$

The distributions are finally used to draw the inter-arrival time $\Delta_{t,k+1}$ and the mark x_{k+1} for the event e_{k+1}. The sampled inter-arrival time $\Delta_{t,k+1}$ gives



(a) Setup and components of IMTPP (b) MTPP for observations p_θ (c) Posterior MTPP for missing events q_ϕ

Figure 1: Overview and neural architecture of IMTPP. Panel (a) illustrates an the problem setup, notations as well as an overview of the components IMTPP. Panel (b) shows the neural architecture of the MTPP of observations p_θ. Panel (c) shows the neural architecture of the posterior MTPP of missing events q_ϕ. Note that, the information of e_{k+1} here is used to truncate the log-normal distribution, whereas the log-normal distribution is non-truncated. Prior MTPP for missing events has a simpler architecture and therefore is omitted in this illustration.

$t_{k+1} = t_k + \Delta_{t,k}$. Here, the mark distribution is independent of $\Delta_{t,k+1}$. Finally, we note that $\theta = \{\boldsymbol{W}_{\bullet,\bullet}, \boldsymbol{w}_{\bullet,\bullet}, \boldsymbol{U}_{\bullet,\bullet}, \boldsymbol{a}_\bullet\}$ are trainable parameters.

We would like to highlight that, the proposed lognormal distribution of inter-arrival times $\Delta_{t,k}$ allows an easy re-parameterization trick—$\textsc{Lognormal}(\mu_e, \sigma_e) = \exp(\mu_e + \sigma_e \cdot \textsc{Normal}(0,1))$—which mitigates variance of estimated parameters and facilitates fast training and accurate prediction.

4.3 Parameterization of q_ϕ

At the very outset, $q_\phi(\bullet \mid e_k, \boldsymbol{s}_k, \boldsymbol{m}_{r-1})$ (Eq. 5) generates missing events that are likely to be omitted during the interval (t_k, t_{k+1}) after the knowledge of the subsequent observed event e_{k+1} is taken into account. To ensure that missing events are generated within desired interval, (t_k, t_{k+1}), whenever an event is drawn with $\tau_r > t_{k+1}$, then $q_\phi(\bullet \mid e_{k+1}, \boldsymbol{s}_k, \boldsymbol{m}_{r-1})$ is set to zero and $\bar{k}$ is set to $r-1$. Otherwise, $\bar{k}$ is flagged as $\underline{k}$. Similar to the generator for observed events p_θ, it has also a three level neural architecture.

Input layer. Given the subsequent observed event t_{k+1} along with $\mathcal{S}_k$ and $\epsilon_{r-1} = (y_{r-1}, \tau_{r-1})$ arrives with $\tau_{r-1} < t_{k+1}$ or equivalently if $r - 1 \neq \bar{k}$, then we first convert τ_{r-1} into a suitable representation.

$$\boldsymbol{\gamma}_{r-1} = \boldsymbol{g}_{\tau,\gamma}\tau_{r-1} + \boldsymbol{g}_{y,\gamma}y_{r-1} + \boldsymbol{g}_{\Delta,\gamma}(\tau_{r-1} - \tau_{r-2}) + \boldsymbol{b}_\gamma,$$

where $\boldsymbol{g}_{\bullet,\bullet}$ and $\boldsymbol{b}_\gamma$ are trainable parameters.

Hidden layer. Similar to the hidden layer used in p_θ model, the hidden layer here too embeds the sequence of missing events into finite dimensional vectors $\boldsymbol{m}_\bullet$, computed using RNN in a recurrent manner. Such a layer takes $\boldsymbol{\gamma}_{r-1}$ as input and feed it into an RNN to update its hidden states in the following way.

$$\boldsymbol{m}_{r-1} = \tanh\left(\boldsymbol{G}_{m,m}\boldsymbol{m}_{r-2} + \boldsymbol{G}_{m,\gamma}\boldsymbol{\gamma}_{r-1} + (\tau_{r-1} - \tau_{r-2})\,\boldsymbol{g}_{m,\tau} + \boldsymbol{b}_m\right), \quad (10)$$

where $\boldsymbol{G}_{\bullet,\bullet}, \boldsymbol{g}_{\bullet,\bullet}$ and $\boldsymbol{b}_m$ are trainable parameters.

Output layer. The next level is the output layer which computes both $q_{\phi,\Delta}(\cdot)$ and $\mathbb{Q}_{\phi,y}(\cdot)$ based on $\boldsymbol{m}_r$ and $\boldsymbol{s}_k$. To compute these quantities, it takes five signals as input: (i) the current update of the hidden state $\boldsymbol{m}_r$ for the RNN in the previous layer, (ii) the current update of the hidden state $\boldsymbol{s}_k$ that embeds the history of observed events, and (iii) the timing of the last observed event, t_k, (iv) the timing of the last missing event, τ_{r-1} and (v) the timing of the next observation, t_{k+1}. To this end, we have the density of inter-arrival times as

$$q_{\phi,\Delta}(\Delta_{\tau,r} \mid e_{k+1}, \boldsymbol{s}_k, \boldsymbol{m}_{r-1}) = \textsc{Lognormal}\left(\mu_\epsilon(\boldsymbol{m}_{r-1}, \boldsymbol{s}_k), \sigma^2_\epsilon(\boldsymbol{m}_{r-1}, \boldsymbol{s}_k)\right) \odot [\![\tau_{r-1} + \Delta_{\tau,r} < t_{k+1}]\!], \quad (11)$$

with $[\mu_\epsilon(\boldsymbol{m}_{r-1}, \boldsymbol{s}_k), \sigma_\epsilon(\boldsymbol{m}_{r-1}, \boldsymbol{s}_k)] = \boldsymbol{G}^\top_{\tau,m}\boldsymbol{m}_{r-1} + \boldsymbol{G}^\top_{\tau,s}\boldsymbol{s}_k + \boldsymbol{b}_\tau$; and, the mark distribution as,

$$\mathbb{P}_{\theta,x}(y_r = y \mid \Delta_{\tau,r}, e_{k+1}, \boldsymbol{s}_k, \boldsymbol{m}_{r-1}) = \frac{[\![\tau_{r-1} + \Delta_{\tau,r} < t_{k+1}]\!] \odot \exp(\boldsymbol{V}^\top_{y,s}\boldsymbol{s}_k + \boldsymbol{V}^\top_{y,m}\boldsymbol{m}_{r-1})}{\sum_{y' \in \mathcal{C}} \exp(\boldsymbol{V}^\top_{y',s}\boldsymbol{s}_k + \boldsymbol{V}^\top_{y',m}\boldsymbol{m}_{r-1})}, \quad (12)$$

Hence, we have:

$$\Delta_{\tau,r} \sim q_{\phi,\Delta}(\bullet \mid e_{k+1}, \boldsymbol{s}_k, \boldsymbol{m}_{r-1})$$

If $\Delta_{\tau,r} < t_{k+1} - \tau_{r-1}$:

$$\tau_r = \tau_j + \Delta\tau,$$
$$y_r \sim \mathbb{P}_{\theta,x}(y_r = y \,|\, \Delta_{\tau,r}, e_{k+1}, \boldsymbol{s}_k, \boldsymbol{m}_{r-1})$$
$$\overline{k} = \infty \texttt{ (Allow more missing events)}$$

Otherwise:

$$\overline{k} = r - 1.$$

Here, note that the mark distribution depends on $\Delta_{\tau,r}$. $\phi = \{\boldsymbol{G}_{\bullet,\bullet}, \boldsymbol{g}_{\bullet,\bullet}, \boldsymbol{V}_{\bullet,\bullet}, \boldsymbol{b}_\bullet\}$ are trainable parameters.

The distributions in Eqs. 11– 12 ensure that given the first $k+1$ observations, q_ϕ generates the missing events only for (t_k, t_{k+1}) and not for further subsequent intervals.

4.4 Prior MTPP model p_{prior}

We model the prior density (Eq. 6) of the arrival times of the missing events as,

$$p_{\text{prior},\Delta}(\Delta_{\tau,r} \,|\, \boldsymbol{s}_k, \boldsymbol{m}_{r-1}) = \textsc{Lognormal}\left(\mu(\boldsymbol{s}_k, \boldsymbol{m}_{r-1}), \sigma^2(\boldsymbol{s}_k, \boldsymbol{m}_{r-1})\right)$$

with $[\mu(\boldsymbol{s}_k, \boldsymbol{m}_{r-1}), \sigma^2(\boldsymbol{s}_k, \boldsymbol{m}_{r-1}] = \boldsymbol{q}_{\mu,m}^\top \boldsymbol{m}_{r-1} + \boldsymbol{q}_{\mu,s}^\top \boldsymbol{s}_k + \boldsymbol{c}$; and, the mark distribution of the missing events as,

$$\mathbb{P}_{\text{prior},y}(y_r = y \,|\, \Delta_{\tau,r}, \boldsymbol{s}_k, \boldsymbol{m}_{r-1}) = \frac{\exp(\boldsymbol{Q}_{y,s}^\top \boldsymbol{s}_k + \boldsymbol{Q}_{y,m}^\top \boldsymbol{m}_{r-1})}{\sum_{y' \in \mathcal{C}} \exp(\boldsymbol{Q}_{y',s}^\top \boldsymbol{s}_k + \boldsymbol{Q}_{y',m}^\top \boldsymbol{m}_{r-1})}, \tag{13}$$

All parameters $\boldsymbol{Q}_{\bullet,\bullet}$, $\boldsymbol{q}_{\bullet,\bullet}$ and $\boldsymbol{c}$ are set a-priori.

4.5 Training θ and ϕ

Note that the trainable parameters for observed and posterior MTPPs are $\theta = \{\boldsymbol{w}_{\bullet,\bullet}, \boldsymbol{W}_{\bullet,\bullet}, \boldsymbol{a}_\bullet, \boldsymbol{U}_{\bullet,\bullet}\}$ and $\phi = \{\boldsymbol{g}_{\bullet,\bullet}, \boldsymbol{G}_{\bullet,\bullet}, \boldsymbol{b}_\bullet, \boldsymbol{V}_{\bullet,\bullet}\}$ respectively. Given a history $\mathcal{S}_K$ of observed events, we aim to learn θ and ϕ by maximizing ELBO, as defined in Eq. 3, *i.e.*

$$\max_{\theta,\phi} \text{ELBO}(\theta, \phi) \tag{14}$$

We compute optimal parameters θ^* and ϕ^* that maximizes $\text{ELBO}(\theta, \phi)$ by means of stochastic gradient descent (SGD) method (Rumelhart et al., 1986).

4.6 Salient features of our proposal

It is worth noting the similarity of our modeling and inference framework to variational autoencoders (Chung et al., 2015; Doersch, 2016; Bowman et al., 2015), with q_ϕ and p_θ playing the roles of encoder and decoder respectively, while p_{prior} plays the role of the prior distribution of latent events. However, the random seeds in our model are not simply noise as they are interpreted in autoencoders. They can be concretely interpreted in IMTPP as missing events, making our model physically interpretable.

Secondly, note that the proposal of (Mei et al., 2019) aims to impute the missing events based on the entire observation sequence $\mathcal{S}_K$, rather than to predict observed events in the face of missing events. For this purpose, it uses a bi-directional RNN and, whenever a new observation arrives, it re-generates all missing events by making a completely new pass over the backward RNN. As a consequence, such an imputation method suffers from the quadratic complexity with respect to the number of observed events. In contrast, our proposal is designed to generate subsequent observed and missing events rather than imputing missing events in between observed events[2]. To that aim, we only make forward computations and therefore, it does not require to re-generate all missing events whenever a new observation arrives, which makes it much more efficient than (Mei et al., 2019) in terms of both learning and prediction. Through our experiments we also show the exceptionally time-effective operation of IMTPP over other missing-data models.

Finally, unlike most of the prior work (Du et al., 2016; Zhang et al., 2020; Mei et al., 2019; Mei and Eisner, 2017; Shelton et al., 2018; Zuo et al., 2020) we model our distribution for inter-arrival times using log-normal. While Shchur et al. (2020) also use model inter-arrival times using log-normal, they do not focus to predict observations in the face of missing events. However, it is important to reiterate (see Shchur et al. (2020) for details) that this modeling choice offers significant advantages over intensity based models in terms of providing ease of re-parameterization trick for efficient training, allowing a closed form expression for expected arrival times, and enabling usability for supervised training as well.

5 Experiments

In this section, we report a comprehensive empirical evaluation of IMTPP. Specifically, we address the following research questions. **RQ1:** Can IMTPP accurately predict the dynamics of the missing events? **RQ2:** What is the mark and time prediction performance of IMTPP in comparison to the state-of-the-art baselines? Where are the gains and losses? **RQ3:** How does IMTPP perform in the long term forecasting? **RQ4:** How does the efficiency of IMTPP compare with the proposal of Mei et al. (2019)?

[2] However, note that the posterior distribution q_ϕ can easily be used to impute missing events between already occurred events.

	Mean Absolute Error (MAE)						Mean Prediction Accuracy (MPA)					
	Movies	Toys	Taxi	Retweet	SO	Foursquare	Movies	Toys	Taxi	Retweet	SO	Foursquare
HP (Hawkes, 1971)	0.060	0.062	0.220	0.049	0.010	0.098	0.482	0.685	0.894	0.531	0.418	0.523
SMHP (Liniger, 2009)	0.062	0.061	0.213	0.051	0.008	0.091	0.501	0.683	0.893	0.554	0.423	0.520
RMTPP (Du et al., 2016)	0.053	0.048	0.128	0.040	**0.005**	0.047	0.548	0.734	0.929	0.572	0.446	0.605
SAHP (Zhang et al., 2020)	0.072	0.073	0.174	0.081	0.017	0.108	0.458	0.602	0.863	0.461	0.343	0.459
THP (Zuo et al., 2020)	0.068	0.057	0.193	0.047	0.006	0.052	0.537	0.724	0.931	0.526	**0.458**	**0.624**
PFPP (Mei et al., 2019)	0.058	0.055	0.181	0.042	0.007	0.076	0.559	0.738	0.925	0.569	0.437	0.582
HPMD (Shelton et al., 2018)	0.060	0.061	0.208	0.048	0.008	0.087	0.513	0.688	0.907	0.558	0.439	0.531
IMTPP (our proposal)	**0.049**	**0.045**	**0.108**	**0.038**	**0.005**	**0.041**	**0.574**	**0.746**	**0.938**	**0.577**	0.451	0.612

Table 1: Performance of all the methods in terms of time prediction error (MAE) and mark prediction accuracy (MPA) across all datasets on the 20% test set. Numbers with bold font (boxes) indicate best (second best) performer. It shows that IMTPP is the best performer for a majority of all the datasets.

5.1 Experimental setup

Datasets. We utilize a publicly available synthetic dataset [3] used in Zhang et al. (2020), to address **RQ1**. Here, we randomly sample 10% events and tag them to be missing. For all other experiments, we use six diverse datasets: Amazon movies (Movies) (Ni et al., 2019), Amazon toys (Toys) (Ni et al., 2019), NYC-Taxi (Taxi), Retweet (Zhao et al., 2015), Stackoverflow (SO) (Du et al., 2016) and Foursquare (Yang et al., 2019), as described below.

(1) **Amazon Movies** (Ni et al., 2019). For this dataset we consider the reviews given to items under the category "Movies" on Amazon. For each item we consider the time of the written review as the time of event in the sequence and the rating as the corresponding mark.

(2) **Amazon Toys** (Ni et al., 2019). Similar to Amazon Movies, but here we consider the items under the category "Toys".

(3) **NYC Taxi**[4]. Each sequence corresponds to a series of time-stamped pick-up and drop-off events of a taxi in New York City with locations as marks.

(4) **Retweet** (Zhao et al., 2015). Similar to (Mei and Eisner, 2017), we group retweeting users into three classes based on their connectivity: ordinary user (degree lower than the median), popular user (degree lower than 95-percentile) and *influencers* (degree higher than 95-percentile). Each stream of retweets is treated as a sequence of events with retweet time as the event time, and user class as the mark.

(5) **Stack Overflow**. Similar to (Du et al., 2016), we treat the badge awarded to a user on the *stack overflow* forum as a mark. Thus we have each user corresponding a sequence of events with *times* corresponding to the time of mark affiliation.

(6) **Foursquare**. As a novel evaluation dataset, we use Foursquare (a location search and discovery app) crawls (Yang et al., 2019) to construct a collection of check-ins from *Japan*. Each user has a sequence with the mark corresponding to the *type* of the check-in location (e.g. "Jazz Club") and the time as the timestamp of the check-in.

[3] https://github.com/QiangAIResearcher/sahp_repo

[4] https://chriswhong.com/open-data/foil_nyc_taxi/

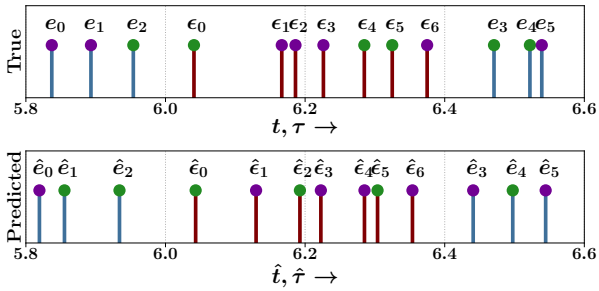


Figure 2: Qualitative Analysis of mark and time prediction performance of IMTPP over the synthetic dataset. In the top figure, we show the *observed* (bars in blue) as well as events *hidden* (bars in brown) during learning between timestamps 6.0 and 6.4. In the lower figure, we show the events predicted by IMTPP for the same sequence. Marks are represented using different colored (cyan and purple) circles.

Baselines. We compare IMTPP with eight baselines: (i) Hawkes process (HP) (Hawkes, 1971; Du et al., 2016), (ii) Self modulating Hawkes process (SMHP) (Liniger, 2009), (iii) Recurrent marked temporal point process (RMTPP) (Du et al., 2016), (iv) Self attentive Hawkes process (SAHP) (Zhang et al., 2020), (v) Transformer Hawkes process (THP) (Zuo et al., 2020) (vi) Particle filtering of point process with missing events (PFPP) (Mei et al., 2019) and (vii) Hawkes process with missing data (HPMD) (Shelton et al., 2018).

Evaluation protocol. Given a stream of N *observed* events $\mathcal{S}_N$, we split them into training $\mathcal{S}_K$ and test set $\mathcal{S}_N \backslash \mathcal{S}_K$, where the training set (test set) consists of first 80% (last 20%) events, *i.e.*, $K = \lceil 0.8N \rceil$. We train IMTPP and the baselines on $\mathcal{S}_K$ and then evaluate the trained models on the test set $\mathcal{S}_N \backslash \mathcal{S}_K$ in terms of (i) mean absolute error (MAE) of predicted times, *i.e.*, $\frac{1}{|\mathcal{S}_N \backslash \mathcal{S}_K|} \sum_{e_i \in \mathcal{S}_N \backslash \mathcal{S}_K} \mathbb{E}[|t_i - \hat{t}_i|]$ and (ii) mark prediction accuracy (MPA), *i.e.*, $\frac{1}{|\mathcal{S}_N \backslash \mathcal{S}_K|} \sum_{e_i \in \mathcal{S}_N \backslash \mathcal{S}_K} \mathbb{P}(x_i =$

$\hat{x}_i$). Here $\hat{t}_i$ and $\hat{x}_i$ are the predicted time and mark the i-th event in test set. Note that such predictions are made only on observed events in real datasets.

For our experiments in this paper, we make our code and data public at https://github.com/data-iitd/imtpp. Our code uses Tensorflow[5] v.1.13.1 and Tensorflow-Probability v0.6.0[6].

5.2 Implementation details

Parameter Settings. For our experiments, we set $\dim(\boldsymbol{v}_\bullet) = 16$, and $\dim(\gamma_\bullet) = 32$, where $\boldsymbol{v}_\bullet$ and $\gamma_\bullet$ are the output of the first layers in p^*_θ and $q^*\phi$ respectively; the sizes of hidden states as $\dim(\boldsymbol{h}_\bullet) = 64$ and $\dim(\boldsymbol{z}_\bullet) = 128$; batch-size $B = 64$. In addition we set an l_2 regularizer over the parameters where the l_2 regularizing coefficient has value 0.001.

System Configuration. All our experiments were done on a server running Ubuntu 16.04. CPU: Intel(R) Xeon(R) Gold 5118 CPU @ 2.30GHz , RAM: 125GB and GPU: NVIDIA Tesla T4 16GB DDR6.

Baseline Implementation Details. For implementations regarding Markov chain, we use the code[7] by (Du et al., 2016) and for RMTPP we use the Python-based implementation[8]. For Hawkes and self-modulating Hawkes, we use the codes[9] made available by Mei and Eisner (2017). Since HP and SMHP (Liniger, 2009) generate a sequence of events of a specified length from the weights learned over the training set, we generate $|N|$ sequences as per the data as $\mathcal{S} = \{s_1, s_2, \cdots s_N\}$ each with maximum sequence length. For evaluation, we consider the first l_i set of events for each sequence i. Recent research (Li et al., 2018) states that Neural Hawkes (Mei and Eisner, 2017) and (Du et al., 2016) show similar performance. These deviations are subjected to extensive to parameter tuning, which however is beyond scope of this paper. For rest of the baselines, we used the implementations provided by the respective authors — PFPP[10], HPMD[11], THP[12] and SAHP[13].

5.3 Results

Prediction of missing events. To address the research question **RQ1**, we first qualitatively demonstrate the ability of IMTPP in predicting *missing* events in a sequence. To do so, we train IMTPP over the observed events and use our trained model predict both observed and missing events. Figure 2 provides an illustrative setting of our results. It shows that IMTPP provides many accurate mark predictions. In addition, it qualitatively shows that the predicted inter-arrival times closely matches with the true inter-arrival times.

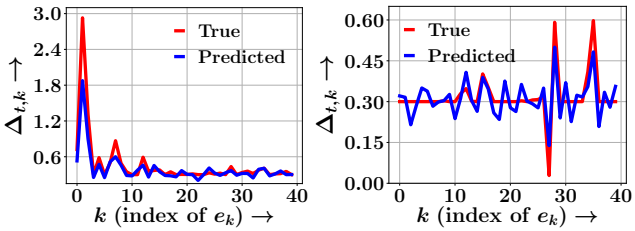
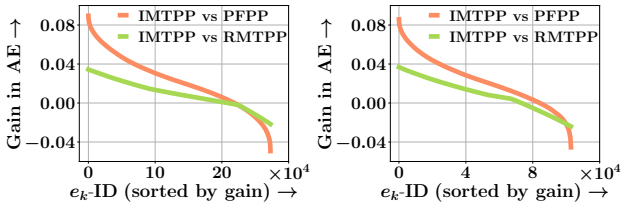


Figure 3: Real life examples of true and predicted inter-arrival times $\Delta_{t,k}$ of different events e_k, against k for $k \in \{k+1, \ldots, N\}$.

Comparative analysis on real data. Next, we address the research question **RQ2**. More specifically, we compare the performance of IMTPP with all the baselines introduced above over all six datasets.

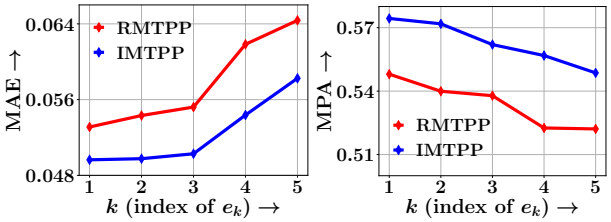
— *Analysis of MAE and MPA:* Table 1 summarizes the results, which sketches the comparative analysis in terms of mean absolute error (MAE) on time and mark prediction accuracy (MPA), respectively. They reveal the following observations. (1) IMTPP exhibits steady improvement over all the baselines in most of the datasets, in case of both time and mark prediction. However, for Stackoverflow and Foursquare datasets, THP outperforms all other models including IMTPP in terms of MPA. (2) RMTPP is the second best performer in terms of MAE of time prediction almost in all datasets. In fact, in Stackoverflow (SO) dataset, it shares the lowest MAE together with IMTPP. However, there is no consistent second best performer in terms of MPA. Notably, PFPP and IMTPP, which take into account of missing events, are the second best performers for four datasets. (3) Both PFPP (Mei et al., 2019) and HPMD (Shelton et al., 2018) fare poorly with respect to RMTPP in terms of MAE. This is because PFPP focus on imputing missing events based on the complete observations and, is not well suited to predict observed events in the face of missing observations. In fact, PFPP does not offer a joint training mechanism for the MTPP for observed events and the imputation model. Rather it trains an imputation model based on the observation model learned a-priori. On the other hand, HPMD only assumes linear Hawkes process with known influence structure. Therefore it shows poor performance with respect to

[5] https://www.tensorflow.org/
[6] https://www.tensorflow.org/probability
[7] https://github.com/dunan/NeuralPointProcess
[8] https://github.com/musically-ut/tf_rmtpp
[9] https://github.com/HMEIatJHU/neurawkes
[10] https://github.com/HMEIatJHU/neural-hawkes-particle-smoothing
[11] https://github.com/cshelton/hawkesinf
[12] https://github.com/SimiaoZuo/Transformer-Hawkes-Process
[13] https://github.com/QiangAIResearcher/sahp_repo



(a) Movies (b) Toys

Figure 4: Performance gain in terms of AE(baseline) − AE(IMTPP)— the gain (above x-axis) or loss (below x-axis) of the average error per event $\mathbb{E}[|t_k - \hat{t_k}|]$ of IMTPP— with respect to two competitive baselines: RMTPP and PFPP. Events in the test set are sorted by decreasing gain of IMTPP along x-axis.



(a) Movies, MAE (b) Movies, MPA

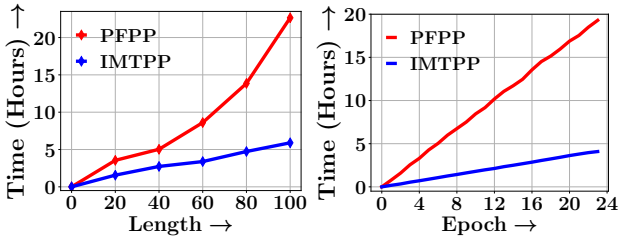
Figure 5: Variation of forecasting performance of IMTPP and RMTPP in terms of MAE and MPA at predicting next i-th event, against i for Movies dataset. Panels (a–b) show the variation of MAE while panels (c–d) show the variation of MPA.

RMTPP.

— *Qualitative analysis:* In addition, Figure 3 provides some real-life examples taken from Movies and Toys datasets, which qualitatively show that the predicted inter-arrival times closely matches with the true inter-arrival times.

— *Drill down analysis:* Next, we provide a comparative analysis of the time prediction performance at every event in the test set. To this end, for each observed event e_i in the test, we compute the gain (or loss) IMTPP achieves in terms of the average error per event $\mathbb{E}[|t_k - \hat{t_k}|]$, *i.e.*, AE(baseline) − AE(IMTPP) for two competitive baselines, *e.g.*, RMTPP and PFPP for Movies and Toys datasets. Figure 4 summarizes the results, which shows that IMTPP outperforms the most competitive baseline (RMTPP) for more than 70% events across both Movies and Toys datasets.

Forecasting future events. Here, we aim to address the research question **RQ3**. To make a more challenging evaluation of IMTPP against its competitors we design a difficult event prediction task, where we predict the next n events given only the current event as input. To do so, we keep sampling events using the trained model $p_{\hat{\theta}}$ and $q_{\hat{\phi}}$ till n-th prediction. Such



(a) Time vs $|\mathcal{S}_K|$ (b) Time vs # of epochs

Figure 6: Runtime performance of PFPP and IMTPP for across Movies Dataset. Panel (a) shows time vs. length of training sequence and panel (b) shows time vs. number of epochs.

an evaluation protocol effectively requires accurate inference of the missing data distribution, since, unlike during the training phase, the future observations are not fed into the missing event model. To this end, we compare the forecasting performance of IMTPP against RMTPP, the most competitive baseline.

Figure 5 summarizes the results, which shows that (i) the performances of all the algorithms deteriorate as n increases and; (ii) IMTPP achieves 5.5% improvements in MPA and significantly better 10.12% improvements in MAE than RMTPP on Movies dataset.

Scalability analysis. Here, we address the research question **RQ4** by comparing the runtime of IMTPP with PFPP across the no. of training epochs as well as the length of training sequence $|\mathcal{S}_K|$. Figure 6 summarizes the results, which shows that IMTPP enjoys a better latency than PFPP. In particular, we observe that the runtime of PFPP increases quadratically with respect to $|\mathcal{S}_K|$, whereas, the runtime of IMTPP increases linearly. The quadratic complexity of PFPP is due to the presence of an backward RNN which requires a complete pass whenever a new event arrives.

6 Conclusion

In this paper, we provide a method for incorporating missing events for training a marked temporal point processes, that simultaneously samples missing as well as observed events across continuous time. Most earlier methods relied on complete data - an ideal and rare setting. Experiments on several real datasets from diverse application domains show that our proposal outperforms several alternatives. Since including missing data, *improves* over standard learning procedures, this observation opens avenues for further research that includes missing data. We plan to further extend our model to incorporate partial missing aspects i.e. either time *or* mark is missing from the event data, which may give even more flexibility to our model.

Acknowledgements

Abir De acknowledges DST Inspire research grant.

References

Bacry, E., Mastromatteo, I., and Muzy, J.-F. (2015). Hawkes processes in finance. *Market Microstructure and Liquidity*, 1(01):1550005.

Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A. M., Jozefowicz, R., and Bengio, S. (2015). Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*.

Che, Z., Purushotham, S., Cho, K., Sontag, D., and Liu, Y. (2018). Recurrent neural networks for multivariate time series with missing values. *Nature Scientific Reports*.

Chung, J., Kastner, K., Dinh, L., Goel, K., Courville, A. C., and Bengio, Y. (2015). A recurrent latent variable model for sequential data. In *NIPS*.

Daley, D. J. and Vere-Jones, D. (2007). *An introduction to the theory of point processes: volume II: general theory and structure*. Springer Science & Business Media.

De, A., Valera, I., Ganguly, N., Bhattacharya, S., and Gomez-Rodriguez, M. (2016). Learning and forecasting opinion dynamics in social networks. In *NIPS*.

Doersch, C. (2016). Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*.

Du, N., Dai, H., Trivedi, R., Upadhyay, U., Gomez-Rodriguez, M., and Song, L. (2016). Recurrent marked temporal point processes: Embedding event history to vector. In *KDD*.

Du, N., Farajtabar, M., Ahmed, A., Smola, A. J., and Song, L. (2015). Dirichlet-hawkes processes with applications to clustering continuous-time document streams. In *KDD*.

Farajtabar, M., Yang, J., Ye, X., Xu, H., Trivedi, R., Khalil, E., Li, S., Song, L., and Zha, H. (2017). Fake news mitigation via point process based intervention. In *ICML*.

Guo, R., Li, J., and Liu, H. (2018). Initiator: Noise-contrastive estimation for marked temporal point process. In *IJCAI*.

Hawkes, A. G. (1971). Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1).

Huang, H., Wang, H., and Mak, B. (2019). Recurrent poisson process unit for speech recognition. In *AAAI*.

Jing, H. and Smola, A. J. (2017). Neural survival recommender. In *WSDM*.

Kumar, S., Zhang, X., and Leskovec, J. (2019). Predicting dynamic embedding trajectory in temporal interaction networks. In *KDD*.

Li, S., Xiao, S., Zhu, S., Du, N., Xie, Y., and Song, L. (2018). Learning temporal point processes via reinforcement learning. In *NIPS*.

Likhyani, A., Gupta, V., Srijith, P., Deepak, P., and Bedathur, S. (2020). Modeling implicit communities from geo-tagged event traces using spatio-temporal point processes. In *WISE*.

Liniger, T. J. (2009). *Multivariate hawkes processes*. PhD thesis, ETH Zurich.

Little, R. J. and Rubin, D. B. (2019). *Statistical analysis with missing data*, volume 793. John Wiley & Sons.

Lorch, L., De, A., Bhatt, S., Trouleau, W., Upadhyay, U., and Gomez-Rodriguez, M. (2018). Stochastic optimal control of epidemic processes in networks. *arXiv preprint arXiv:1810.13043*.

Mei, H. and Eisner, J. M. (2017). The neural hawkes process: A neurally self-modulating multivariate point process. In *NIPS*.

Mei, H., Qin, G., and Eisner, J. (2019). Imputing missing events in continuous-time event streams. In *ICML*.

Ni, J., Li, J., and McAuley, J. (2019). Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *EMNLP-IJCNLP*, pages 188–197.

Rizoiu, M.-A., Mishra, S., Kong, Q., Carman, M., and Xie, L. (2018). Sir-hawkes: on the relationship between epidemic models and hawkes point processes. In *The Web Conference*.

Rizoiu, M.-A., Xie, L., Sanner, S., Cebrian, M., Yu, H., and Van Hentenryck, P. (2017). Expecting to be hip: Hawkes intensity processes for social media popularity. In *WWW*.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088):533–536.

Shchur, O., Biloš, M., and Günnemann, S. (2020). Intensity-free learning of temporal point processes. In *ICLR*.

Shelton, C. R., Qin, Z., and Shetty, C. (2018). Hawkes process inference with missing data. In *AAAI*.

Śmieja, M., Struski, L., Tabor, J., Zieliński, B., and Spurek, P. (2018). Processing of missing data by neural networks. In *NIPS*.

Tabibian, B., Upadhyay, U., De, A., Zarezade, A., Schölkopf, B., and Gomez-Rodriguez, M. (2019). Enhancing human learning via spaced repetition optimization. *PNAS*.

Tian, Y., Zhang, K., Li, J., Lin, X., and Yang, B. (2018). Lstm-based traffic flow prediction with missing data. *Neurocomputing*.

Valera, I., Gomez-Rodriguez, M., and Gummadi, K. (2014). Modeling diffusion of competing products and conventions in social media. *arXiv preprint arXiv:1406.0516*.

Wang, P., Fu, Y., Liu, G., Hu, W., and Aggarwal, C. (2017). Human mobility synchronization and trip purpose detection with mixture of hawkes processes. In *KDD*.

Yang, D., Qu, B., Yang, J., and Cudre-Mauroux, P. (2019). Revisiting user mobility and social relationships in lbsns: A hypergraph embedding approach. In *WWW*.

Yoon, J., Zame, W. R., and van der Schaar, M. (2019). Estimating missing data in temporal data streams using multi-directional recurrent neural networks. *IEEE Trans. on Biomedical Engineering*.

Zhang, Q., Lipani, A., Kirnap, O., and Yilmaz, E. (2020). Self-attentive hawkes processes. *ICML*.

Zhao, Q., Erdogdu, M. A., He, H. Y., Rajaraman, A., and Leskovec, J. (2015). Seismic: A self-exciting point process model for predicting tweet popularity. In *KDD*.

Zuo, S., Jiang, H., Li, Z., Zhao, T., and Zha, H. (2020). Transformer hawkes process. In *ICML*.