INITIATOR: Noise-contrastive Estimation for Marked Temporal Point Process

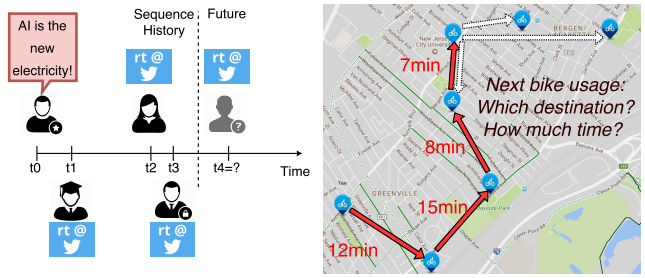
Ruocheng Guo, Jundong Li, Huan Liu
Computer Science and Engineering, Arizona State University, USA
{rguo12, jundongl, huan.liu}@asu.edu

Abstract

Copious sequential event data has consistently increased in various high-impact domains such as social media and sharing economy. When events start to take place in a sequential fashion, an important question arises: *"what type of event will happen at what time in the near future?"* To answer the question, a class of mathematical models called the marked temporal point process is often exploited as it can model the timing and properties of events seamlessly in a joint framework. Recently, various recurrent neural network (RNN) models are proposed to enhance the predictive power of mark temporal point process. However, existing marked temporal point models are fundamentally based on the Maximum Likelihood Estimation (MLE) framework for the training, and inevitably suffer from the problem resulted from the intractable likelihood function. Surprisingly, little attention has been paid to address this issue. In this work, we propose INITIATOR - a novel training framework based on noise-contrastive estimation to resolve this problem. Theoretically, we show the exists a strong connection between the proposed INITIATOR and the exact MLE. Experimentally, the efficacy of INITIATOR is demonstrated over the state-of-the-art approaches on several real-world datasets from various areas.

1 Introduction

Recent years have witnessed a booming of sequential event data in a variety of high-impact domains, ranging from the streams of reposts in microblogging platforms to the usage records of a bike in bike sharing programs. More often than not, such data often carries two sources of valuable information - the *type* (a.k.a. *feature* or *mark*) and the *timing* of the event. For example, as shown in Figure 1a, given a sequence of retweets for the tweet "AI is the new electricity!" on Twitter[1], the event type refers to the category of users who retweet and can either be a celebrity or an average Joe, and the timing of the event refers to the detailed retweet timestamp in the timeline (e.g., t_0, t_1...). With the wide availability of sequential event data, one natural question to ask is: *"given observed sequential events, can we predict the exact timestamp of a particular event in the near future?"* As in the case of bike usage sequence in Figure 1b, we may want to know *"when the given bike will arrive at which bike station soon?"*. The above-mentioned prediction task has significant implications in advancing a variety of real-world applications such as patient treatment suggestion [Lian *et al.*, 2015], predictive maintenance [Xiao *et al.*, 2017], trending tweet prediction [Zhao *et al.*, 2015] and mining human mobility [Gao and Liu, 2015]. To model such sequential data for downstream predictive tasks, a class of mathematical models called the marked temporal point process (MTPP) is often exploited. Given the time and feature of an event, these models jointly estimate how likely the event will happen in the near future by the conditional intensity function (CIF).



(a) Retweet behaviors of a tweet. (b) Usage patterns of a bike.

Figure 1: Two real-world examples of sequential event data.

However, existing efforts are overwhelmingly devoted to the parametrization of MTPP models. Conventional studies on MTPP models heavily focused on the design of CIF to effectively model both the event feature and the timing information [Shen *et al.*, 2014; Gao *et al.*, 2015; Guo and Shakarian, 2016; Tabibian *et al.*, 2017; Liu *et al.*, 2017]. Recently, there is a surge of research in developing RNN based MTPP models [Du *et al.*, 2016; Mei and Eisner, 2016; Xiao *et al.*, 2017]. The work mentioned above aims to enhance the predictive power of MTPP through learning representations for event sequences. Despite their empirical success, little attention has been paid to the training process of MTPP models. The vast majority of existing work leverages

[1] https://twitter.com/

Maximum Likelihood Estimation (MLE) to train MTPP models. However, the likelihood function of a MTPP model is often difficult to estimate because it has to be normalized by a definite integral of CIF which could be intractable to compute, especially for neural MTPP models. To alleviate this issue, existing approaches either: (1) limit CIF to integrable functions; or (2) approximate the likelihood with Monte Carlo sampling. Nonetheless, these two ways either lead to suboptimal specification of CIF or have to take the marginal distribution of event time as a priori. Not to mention other problems of MLE such as mode dropping [Arjovsky and Bottou, 2017]), which refers to the fact that MLE attempts to minimize the asymmetric KL divergence between the data distribution and the generative model. These issues inherently limit the usage of MLE for MTPP models in practice.

In this work, we focus on the development of a more principled framework to facilitate the training process of complex MTPP models whose likelihood functions are intractable. In particular, to overcome the bottlenecks of existing MLE based approaches, we propose a novel framework of noIseCoNtrastIve estmATIon for mArked Temporal pOint pRocess (INITIATOR for short). As opposed to existing MLE based frameworks, INITIATOR calculates the likelihood by learning the definite integral through a set of parameters, which is otherwise intractable to compute. In this regard, no additional constraints or assumptions on CIF or likelihood function need to be imposed, and the proposed INITIATOR framework allows us to search for the optimal CIF model in a wider functional space, which often guarantees the optimal or the near-optimal solution. The main contributions of this paper are:

- We systematically examine the bottlenecks of existing MLE based training framework for MTPP models.
- We propose a novel framework INITIATOR to resolve the issue of intractable likelihood function in training MTPP models.
- We prove that the proposed INITIATOR framework has an inherent connection with its exact MLE counterpart for MTPP models.
- We conduct experiments on multiple real-world datasets to corroborate the efficacy of the proposed INITIATOR framework in the tasks of time and mark prediction.

2 Preliminaries

In this section, we briefly introduce the required background knowledge to facilitate the understanding of the MTPP models. First, we introduce the basic concepts of MTPP and also summarize the used notations throughout this paper. Afterwards, we review the existing MLE training frameworks for marked temporal point process and examine its potential issues or limitations.

2.1 Marked Temporal Point Process

We explain the concepts of marked temporal point process (MTPP) with the retweet sequence example (see Figure 1a) for a better understanding. A sequence of M retweets $\boldsymbol{\tau} = \{(t_1, \boldsymbol{x}_1), ...(t_i, \boldsymbol{x}_i), ..., (t_M, \boldsymbol{x}_M)\}$ is an instance of a marked temporal point process, where $(t_1, ..., t_M) \in \mathbb{R}^M_{>0}$ refers to a strictly ascending sequence of timestamps, and $(\boldsymbol{x}_1, ..., \boldsymbol{x}_i, ..., \boldsymbol{x}_M)$ is the corresponding sequence of d-dimensional marks (e.g., features of retweeters). The symbol $\mathcal{X}^j$ denotes the domain of definition for x_i^j, i.e., the j-th dimension of mark $\boldsymbol{x}_i$ ($\forall i = 1, ..., M; j = 1, ..., d$). Without loss of generality, in this work, the mark of event is treated as a continuous multidimensional variable[2], $\boldsymbol{x}$. For example, in a retweet sequence (Figure 1a), the mark of an event indicates whether the retweeter is a celebrity or not, which naturally makes it a discrete and unidimensional variable. The notation $\tau = (t, \boldsymbol{x})$ denotes the random variables of an event, and τ_i with subscript denotes the i-th event $(t_i, \boldsymbol{x}_i)$.

Given a history of sequence until t_i, i.e., $\mathcal{H}_i = \{(t_1, \boldsymbol{x}_1), ..., (t_i, \boldsymbol{x}_i)\}$, we can characterize it by the conditional intensity function (CIF) as follows:

$$\lambda(\tau|\mathcal{H}_i) = \mathbb{E}[N(t + dt, \boldsymbol{x}|\mathcal{H}_i) - N(t, \boldsymbol{x}|\mathcal{H}_i)], \quad (1)$$

where dt is an infinitesimal interval around t and $N(t, \boldsymbol{x})$ indicates the number of events (e.g., retweets) with mark $\boldsymbol{x}$ (e.g., user feature) in the sequence till t. For example, in Figure 1a, CIF in Eq. (1) evaluates how likely the next retweet would be posted at timestamp t by a user with the feature $\boldsymbol{x}$ by using the conditional intensity, which is a continuous unnormalized scalar. Using the chain rule, we can decompose the CIF into two parts such that $\lambda(\tau) = p(\boldsymbol{x}|t)\lambda(t)$ [Snyder and Miller, 2012], where $p(\boldsymbol{x}|t)$ is the conditional probability of the mark $\boldsymbol{x}$ conditioned on the timestamp t. By setting $p(\boldsymbol{x}|t) = 1$ ($\boldsymbol{x}$ can only take one value), a typical unmarked temporal point process (TPP) can be used to model $\lambda(t)$, and two of the most popular models as follows:

Homogeneous Poisson Process [Kingman, 1993] comes with the assumption that inter-event time intervals are *i.i.d.* samples from an exponential distribution. Thus, the CIF is a constant $\lambda(t) = \frac{\mathbb{E}[N(t)]}{t}$, where $N(t)$ counts events.

Hawkes Process [Hawkes, 1971] has the following formulation of CIF $\lambda(t|\mathcal{H}_i) = \mu_0 + \alpha \sum_{j=1}^{i} \phi(t, t_j)$, where $\phi(t, t_j) \geq 0$ denotes the self-exciting kernel and $\mu_0 \in \mathbb{R}$ is a parameter.

2.2 Maximum Likelihood Estimation for Marked Temporal Point Process

With the likelihood function defined in [Rasmussen, 2011], MLE is the most widely used estimator for TPP models [Shen *et al.*, 2014; Zhao *et al.*, 2015; Gao *et al.*, 2015; Valera and Gomez-Rodriguez, 2015; Du *et al.*, 2016; Mei and Eisner, 2016]. In particular, given the history sequence $\mathcal{H}_{i-1}$, the likelihood of observing the i-th event $\tau_i = (t_i, \boldsymbol{x}_i), t_i > t_{i-1}$ with the CIF $\lambda_{\boldsymbol{\theta}}$ can be formulated as:

$$p_{\boldsymbol{\theta}}(\tau_i|\mathcal{H}_{i-1}) = \lambda_{\boldsymbol{\theta}}(\tau_i) \exp(-\int_{\boldsymbol{x} \in \mathcal{X}} \int_{t_{i-1}}^{t_i} \lambda_{\boldsymbol{\theta}}(\tau) dt d\boldsymbol{x}). \quad (2)$$

Thus, the log likelihood function of observing a sequence of N events $\boldsymbol{\tau} = (\tau_1, \tau_2, ..., \tau_N)$ at time t_N can be written as:

$$\log p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) = \sum_{i=1}^{N} [\log \lambda_{\boldsymbol{\theta}}(\tau_i) - \int_{\boldsymbol{x} \in \mathcal{X}} \int_{t_{i-1}}^{t_i} \lambda_{\boldsymbol{\theta}}(\tau) dt d\boldsymbol{x}], \quad (3)$$

[2] with discrete and unidimensional variable as a special case.

where $t_0 = 0$. By maximizing the above log likelihood, we can obtain the estimated model parameters $\boldsymbol{\theta}$. However, the normalizer $-\int_{\boldsymbol{x}\in\mathcal{X}}\int_{t_{i-1}}^{t_i}\lambda_{\boldsymbol{\theta}}(\tau)dtd\boldsymbol{x}$ is a definite integral of CIF and can often be infeasible to compute, especially when neural networks are used to parameterize CIF.

Although approximation methods such as Monte Carlo sampling can be applied to compute the normalizer and its gradients, strong assumptions have to be made. For example, [Mei and Eisner, 2016] assumed that the events of each sequence are uniformly distributed along the continuous time space. However, such assumptions may not always hold on real-world sequential event data. Hence, it motivates us to develop a novel training framework for complex MTPP models.

3 The Proposed INITIATOR Framework

In this section, we first show how to build the proposed marked temporal point process framework with the principle of noise-contrastive estimation in details, and then we show that the proposed framework has a strong connection with the exact MLE which is often desired by existing MTPP models. Then, we elaborate the training process of INITIATOR by proposing an adaptive noise generation algorithm. At last, we introduce an instantiation of the proposed INITIATOR framework with the state-of-the-art deep learning techniques in modeling sequential data.

3.1 MTPP with Noise-Contrastive Estimation

In noise-contrastive estimation (NCE) [Gutmann and Hyvärinen, 2010], parameters are learned by solving a binary classification problem where samples are classified into two classes, namely *true sample* or *noise sample*. Here, true and noise samples refer to the events observed in the data distribution p_d and a specified noise distribution p_n, respectively. Thus, we use $p(y=1|\tau)$ to denote the probability that the event τ is a sample observed in p_d. Similarly, $p(y=0|\tau)$ denotes the probability that the event τ is not observed in the data but generated from the noise distribution p_n. Intuitively, our target is to maximize $p(y=1|\tau)$ and $p(y=0|\tau)$ for those observed events and generated noise, respectively. Hence, we obtain the following objective function:

$$\arg\max_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{\tau}\sim p_d}\log p(y=1|\boldsymbol{\tau}) + K\mathbb{E}_{\boldsymbol{\tau}\sim p_n}\log p(y=0|\boldsymbol{\tau}), \tag{4}$$

where K is the number of noise samples generated for each sample in the data. In MTPP, given the history sequence $\mathcal{H}_i$ and a random variable $\tau = (t, \boldsymbol{x}), t > t_i$, its posterior probability can be written as:

$$p(y=1|\tau;\mathcal{H}_i) = \frac{p_d(\tau)}{p_d(\tau) + Kp_n(\tau)}, \tag{5}$$

where $p_d(\tau)$ and $p_n(\tau)$ are short for $p_d(\tau|\mathcal{H}_i)$ and $p_n(\tau|\mathcal{H}_i)$, respectively. In detail, $p_d(\tau)$ denotes the probability of observing τ in the data. Similar to MLE, a family of parametric models $p_{\boldsymbol{\theta}}(\tau)$ is used to approximate $p_d(\tau)$. Following the setting of NCE, instead of computing the normalizer as in Eq. (3), we learn a function $z_{\boldsymbol{\theta}_z}(\cdot)$ to replace the normalizer. The re-parametrization and implementation of $z_{\boldsymbol{\theta}_z}$ will be introduced later. The likelihood function of INITIATOR is formulated as follows:

$$p_{\boldsymbol{\theta}}(\tau) = \lambda_{\boldsymbol{\theta}_\lambda}(\tau)\exp(z_{\boldsymbol{\theta}_z}(\tau|\mathcal{H}_i)), \tag{6}$$

where $\boldsymbol{\theta} = \{\boldsymbol{\theta}_\lambda, \boldsymbol{\theta}_z\}$ is the model parameter of INITIATOR. It should be mentioned that directly maximizing the likelihood function in Eq. (6) over the data distribution leads to trivial solutions when the normalizer $z_{\boldsymbol{\theta}_z} \to -\infty$. With $p_{\boldsymbol{\theta}}$ defined, we can reformulate Eq. (4) as:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}|\mathcal{H}_i) = -\mathbb{E}_{\tau\sim p_d(\tau)}[\log\frac{p_{\boldsymbol{\theta}}(\tau)}{p_{\boldsymbol{\theta}}(\tau)+Kp_n(\tau)}] - \\ K\mathbb{E}_{\tau\sim p_n(\tau)}[\log\frac{p_n(\tau)}{p_{\boldsymbol{\theta}}(\tau)+Kp_n(\tau)}].\end{aligned} \tag{7}$$

Given the j-th element of $\boldsymbol{\theta}$ as θ_j, the partial gradient of Eq. (7) against θ_j is:

$$\begin{aligned}\frac{\partial\mathcal{L}(\boldsymbol{\theta})}{\partial\theta_j} = -\mathbb{E}_{\tau\sim p_d(\tau)}[\frac{Kp_n(\tau)}{p_{\boldsymbol{\theta}}(\tau)+Kp_n(\tau)}\times\frac{\partial\log p_{\boldsymbol{\theta}}(\tau)}{\partial\theta_j}] + \\ K\mathbb{E}_{\tau\sim p_n(\tau)}[\frac{p_{\boldsymbol{\theta}}(\tau)}{p_{\boldsymbol{\theta}}(\tau)+Kp_n(\tau)}\times\frac{\partial\log p_{\boldsymbol{\theta}}(\tau)}{\partial\theta_j}].\end{aligned} \tag{8}$$

Then it is natural to ask if there are connections between the proposed INITIATOR and the existing training framework based on MLE. In the following theorem, we show they are inherently connected by the partial gradients.

Theorem 1. *The partial gradients of the loss function of INITIATOR (Eq. (8)) converge to those under the MLE framework as the number of noise samples per true sample $K \to +\infty$, with the following two mild assumptions: (1) the gradient $\bigtriangledown_{\boldsymbol{\theta}} p_{\boldsymbol{\theta}}$ exists for all $\boldsymbol{\theta}$; (2) there exists an integrable function $R(\tau)$ which is the upper bound of $\max_j|\frac{\partial p_{\boldsymbol{\theta}}(\tau)}{\partial\theta_j}|$.*

Proof. Given $\mathcal{H}_i$ and $\tau = (t, \boldsymbol{x}), t > t_i$, we use the definition of expectation $\mathbb{E}_{\tau\sim p}[f(\tau)] = \int_{\boldsymbol{x}\in\mathcal{X}}\int_{t_i}^{+\infty}p(\tau)f(\tau)dtd\boldsymbol{x}$ to expand Eq. (8) as:

$$\frac{\partial\mathcal{L}(\boldsymbol{\theta})}{\partial\theta_j} = \int_{\mathcal{X}}\int_{t_i}^{+\infty}\frac{Kp_n(\tau)(p_d(\tau)-p_{\boldsymbol{\theta}}(\tau))}{p_{\boldsymbol{\theta}}(\tau)+Kp_n(\tau)}\frac{\partial\log p_{\boldsymbol{\theta}}(\tau)}{\partial\theta_j}dtd\boldsymbol{x}. \tag{9}$$

When $K \to +\infty$, we have $\frac{Kp_n(\tau)}{p_{\boldsymbol{\theta}}(\tau)+Kp_n(\tau)} = 1$, thus:

$$\lim_{K\to+\infty}\frac{\partial\mathcal{L}(\boldsymbol{\theta})}{\partial\theta_j} = \int_{\mathcal{X}}\int_{t_i}^{+\infty}(p_d(\tau)-p_{\boldsymbol{\theta}}(\tau))\frac{\partial\log p_{\boldsymbol{\theta}}(\tau)}{\partial\theta_j}dtd\boldsymbol{x}. \tag{10}$$

Then we show that the second term of in Eq. (10) vanishes for all j as $p_{\boldsymbol{\theta}}(\tau)\frac{\partial\log p_{\boldsymbol{\theta}}(\tau)}{\partial\theta_j} = \frac{\partial p_{\boldsymbol{\theta}}(\tau)}{\partial\theta_j}$:

$$\int_{\mathcal{X}}\int_{t_i}^{+\infty}\frac{\partial p_{\boldsymbol{\theta}}(\tau)}{\partial\theta_j}dtd\boldsymbol{x} = \frac{\partial\int_{\mathcal{X}}\int_{t_i}^{+\infty}p_{\boldsymbol{\theta}}(\tau)dtd\boldsymbol{x}}{\partial\theta_j} = \frac{\partial 1}{\partial\theta_j} = 0, \tag{11}$$

where Leibniz Rule [Flanders, 1973] is used to swap the order of partial derivation and integral. Moreover, we know

that $\int_{\boldsymbol{x}\in\mathcal{X}}\int_{t_i}^{+\infty} p_{\boldsymbol{\theta}}(\tau)dtd\boldsymbol{x} = 1$ because the likelihood $p_{\boldsymbol{\theta}}(\tau)$ is a well-defined probability density function. Therefore, in Eq. (10) we are left with:

$$\begin{aligned}\lim_{K\to\infty}\frac{\partial\mathcal{L}(\boldsymbol{\theta})}{\partial\theta_j} &= \int_{\mathcal{X}}\int_{t_i}^{+\infty} p_d(\tau)\frac{\partial\log p_{\boldsymbol{\theta}}(\tau)}{\partial\theta_j}dtd\boldsymbol{x} \\ &= \mathbb{E}_{\tau\sim p_d}[\frac{\partial\log p_{\boldsymbol{\theta}}(\tau)}{\partial\theta_j}],\end{aligned} \tag{12}$$

which is equivalent to the expectation of the gradient of MLE over the data distribution. This completes the proof. □

Therefore, with a reasonable K and a proper p_n, reducing the objective of exact MLE (Eq. (5)) to that of INITIATOR, namely Eq. (7), does not significantly affect the gradients for model parameters $\boldsymbol{\theta}$ in the learning process.

Next, we introduce a re-parametrization trick for INITIATOR which can also be adapted to other NCE frameworks. With this trick, we can avoid the strong assumptions of negative sampling [Mikolov *et al.*, 2013]: (1) p_n is independent of the history $\mathcal{H}_i$ (2) p_n is a uniform distribution s.t. $Kp_n = 1$. Specifically, Eq. (4) can be rewritten as follows with $z' = z'(\tau|\mathcal{H}_i) = \frac{\exp(z_{\boldsymbol{\theta}_z})}{Kp_n(\tau)}$:

$$\begin{aligned}\arg\min_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta}) = -\sum_{\boldsymbol{\tau}}\sum_{j=1}^{i-1}[\log\frac{\lambda_{\boldsymbol{\theta}_\lambda}(\tau_j)z'}{\lambda_{\boldsymbol{\theta}_\lambda}(\tau_j)z'+1}+ \\ \sum_{k=1}^{K}\log\frac{1}{\lambda_{\boldsymbol{\theta}_\lambda}(\tau'_{j,k})z'+1}].\end{aligned} \tag{13}$$

With the aforementioned re-parametrization trick, we can directly learn z' instead of $z_{\boldsymbol{\theta}_z}$. Thus, we do not need to explicitly compute $p_n(\tau)$, which enables us to sidestep the constraint that p_n requires an analytical expression [Gutmann and Hyvärinen, 2010], which further expands the functional space to enable the search for the optimal p_n.

3.2 Adaptive Noise Sample Generation

Aforementioned INITIATOR framework enables us to train complex MTPP models with the principle of NCE. Nonetheless, the development of a sophisticated noise sample generation algorithm is still in its infancy. Here, we propose a novel algorithm for adaptive noise sample generation. This algorithm facilitates the training process of the proposed INITIATOR framework where at least one noise event $\tau'_{i,k}$ has to be generated for an observed event τ_i. As p_n is a continuous joint distribution of time t and mark $\boldsymbol{x}$, it is much more challenging to work out an intuitive p_n than the case of neural language models [Mikolov *et al.*, 2013] where p_n can be a simple univariate deterministic function of word frequency. Gutmann et al. [Gutmann and Hyvärinen, 2010] argued that p_n should be close to p_d because the more difficult the classification problem in Eq. (1) is, the more information model $p_{\boldsymbol{\theta}}$ would capture from the data distribution p_d. Without arbitrary assumptions on p_n, we propose a principled way for adaptive noise generation. The algorithm adaptively pushes the implicit noise distribution p_n towards p_d as $p_{\boldsymbol{\theta}}$ catches more information from p_d.

Algorithm 1 Adaptive noise generation for INITIATOR.

Input: $\mathcal{H}_i, p_{\boldsymbol{\theta}}$
1: Compute prediction $\hat{\tau}_{i+1} = (\hat{t}_{i+1}, \hat{\boldsymbol{x}}_{i+1})$
2: **for** $k = 1$ to K **do**
3: Sample $\tau'_{i+1,k}$ by Eq. (14)
4: **end for**
5: **return** $(\tau'_{i+1,1}, ..., \tau'_{i+1,K})$

Inspired by [Goodfellow, 2014], the key intuition of this algorithm is that, in the l-th iteration of the training process, the current MTPP model $p_{\boldsymbol{\theta}}$ may not be good enough, so we can use it to generate noise samples:

$$t'_{i+1,k} \sim \hat{t}_{i+1} + \mathcal{N}(0, l\sigma_0^2), \quad \boldsymbol{x}'_{i+1,k} = \hat{\boldsymbol{x}}_{i+1}, \tag{14}$$

where $\hat{t}_{i+1}$ and $\hat{\boldsymbol{x}}_{i+1}$ are the predicted time and mark for the $i+1$-th event based on $\mathcal{H}_i$ and $p_{\boldsymbol{\theta}}$. For example, in conventional MTPP models, we can sample S examples by $\hat{\tau}_{i+1,j} = (\hat{t}_{i+1,j}, \hat{\boldsymbol{x}}_{i+1,j}) \sim p_{\boldsymbol{\theta}}(\tau|\mathcal{H}_i), j = 1, ..., S$ and make predictions by estimating expectations: $\hat{t}_{i+1} = \mathbb{E}[t_{i+1}] = \frac{1}{S}\sum_j \hat{t}_{i+1,j}$ and $\hat{\boldsymbol{x}}_{i+1} = \mathbb{E}[\boldsymbol{x}_{i+1}] = \frac{1}{S}\sum_j \hat{\boldsymbol{x}}_{i+1,j}$. In Section 3.3, we show how the predictions can be made with neural MTPP models. The adaptive Gaussian noise is added to ensure that good predictions are not treated as noise samples. The variance increases w.r.t. the iteration number m because the model $p_{\boldsymbol{\theta}}$ makes better predictions as the training process continues.

3.3 A Neural MTPP Model based on INITIATOR

In this subsection, we introduce an instantiation of the proposed INITIATOR framework with the state-of-the-art deep learning models (a.k.a. neural MTPP). Compared with conventional models (see Section 2.1), neural MTPP models handle sequential event data by vector representations. Specifically, a neural MTPP model maps the observed history of a sequence $\mathcal{H}_i$ to vector representation $\boldsymbol{h}_i$. In the designed model, dense layers are used to project the raw input into a multi-dimensional space. Then, Long Short Term Memory (LSTM) [Hochreiter and Schmidhuber, 1997] is used to capture the nonlinear dependencies between τ_i and $\mathcal{H}_{i-1}, i = 2, ..., N$. Consequently, the output of LSTM is regarded as the vector representation. Given input event τ_i and the corresponding noise samples $\tau'_{i,k}, k = 1, ..., K$, we formulate the

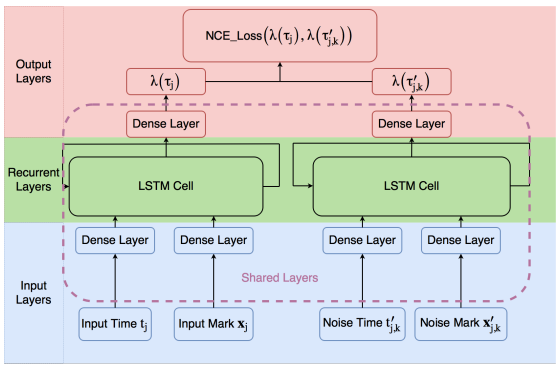


Figure 2: Overview of the MTPP model.

neural MTPP model as below:

$$\begin{aligned} \boldsymbol{s}_i &= [\phi_t(\boldsymbol{w}_t t_i + b_t), \phi_x(\boldsymbol{W}_x \boldsymbol{x}_i + \boldsymbol{b}_x)] \\ (\boldsymbol{h}_i, \boldsymbol{c}_i) &= LSTM(\boldsymbol{s}_i, \boldsymbol{h}_{i-1}, \boldsymbol{c}_{i-1}). \end{aligned} \tag{15}$$

To train this model, an output model is required to map $\boldsymbol{h}_i$ to a scalar y which can be the conditional intensity[3] $\lambda(\tau_i|\mathcal{H}_{i-1})$, the ground conditional intensity $\lambda^t(t_i|\mathcal{H}_{i-1})$, the predicted time $\hat{t}_{i+1}$ or the predicted mark $\hat{\boldsymbol{x}}_{i+1}$. Hence, the CIF of a neural MTPP model can be decomposed as $\lambda_{\boldsymbol{\theta}_\lambda}(\tau|\mathcal{H}_{i-1}) = g^\lambda(f(\mathcal{H}_i))$. In the designed model, a dense layer plays the role of output model, mapping representation to conditional intensity. Then the output model can be framed as:

$$\lambda(\tau_i) = g^\lambda(\boldsymbol{h}_i) = \phi_{out}(\boldsymbol{w}_{out}\boldsymbol{h}_i + b_{out}) \tag{16}$$

To compute the loss function of INITIATOR, similar to the Siamese structure [Bromley *et al.*, 1994], we share dense layers and recurrent layers between inputs - observed event τ_i and its noise $\tau'_{i,k}$. Finally, the conditional intensity of a true event $\lambda(\tau_i)$ and that for its noise samples $\lambda(\tau'_{i,k})$ are fed into the loss function of INITIATOR (Eq. (13)).

Then, we cover how adaptive noise generation is employed in the proposed model. According to Algorithm 1, given the vector representation $\boldsymbol{h}_i$ and output models $g^t, g^{\boldsymbol{x}} = [g^{x^1}, ..., g^{x^d}]$ trained to predict the i+1-th event based on $\boldsymbol{h}_i$, we generate a noise sample $\tau'_{i+1,k}$ by:

$$t'_{i+1,k} \sim g^t(\boldsymbol{h}_i) + \mathcal{N}(0, m\sigma_0^2), \quad \boldsymbol{x}'_{i+1,k} = g^{\boldsymbol{x}}(\boldsymbol{h}_i). \tag{17}$$

4 Experiments

In this section, we conduct experiments to evaluate the performance of the proposed INITIATOR framework. In particular, we attempt to answer the following two research questions: how accurate can the proposed INITIATOR predict (1) the exact timestamp of the event; and (2) the type of that may occur in the near future? Before the details of experiments, the datasets and experimental settings are introduced.

4.1 Dataset Description

We collect three real-world sequential event datasets to answer the above two proposed research questions.

Citi Bike. Citi Bike[4] shares bikes at stations across New York and New Jersey. The activities for a certain bike form a sequence of events. The training set and test set contain the records of the bikes in Jersey City from January to August 2017 and that of September 2017, respectively. Our task is to predict destination of the next ride and its arrival time.

Retweet. We randomly sample 10,000 retweet streams from the Seimic dataset [Zhao *et al.*, 2015] and perform a 5-fold cross-validation. Each stream of retweets for a novel tweet is a sequence of events. The task is to predict the retweet time and the associated class label[5].

[3] y can be predicted by the conditional intensity $\lambda(\tau_{i+1}|\mathcal{H}_i)$ as in [Du *et al.*, 2016] where g takes an extra input τ_{i+1} besides $\mathcal{H}_i$.

[4] https://www.citibikenyc.com/

[5] Following [Mei and Eisner, 2016], retweeters are grouped into three classes: normal user (degree lower than median), influencer (higher than median but lower than 95% percentile) and celebrity (degree higher than 95% percentile).

Dataset	Citi Bike	Retweet	Financial
μ_t	8.135e2(s)	3.228e4(s)	0.619(ms)
σ_t	1.157e4(s)	7.839e4(s)	3.117(ms)
$\lvert\mathcal{X}\rvert$	132	3	2
μ_M	1.839e2	1.458e2	8.000e2
training events	1.873e5	1.468e6	6.400e5
test events	3.299e4	3.716e5	1.600e5

Table 1: Statistics of the datasets, time is in seconds(s) or milliseconds(ms).

Financial. This dataset contains sequences of financial events from a stock traded in US [Du *et al.*, 2016]. To avoid bias in the original dataset, we ensure the length of sequences to be the same by using the first 800 events of each sequence. Then a 5-fold cross-validation is carried out. The task is to predict time and mark (buy or sell) for the next event.

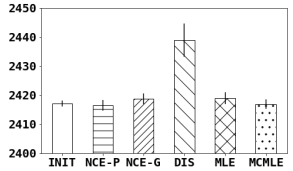
In these datasets, each event only comes with a discrete unidimensional mark. We show the statistics of them in Table 1: mean and standard deviation of time interval between consecutive events (μ_t and σ_t), the number of unique values for a mark ($|\mathcal{X}|$), average sequence lengths (μ_M) and the number of events for training and test.

4.2 Experimental Settings

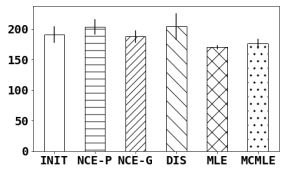
Training is carried out with mini-batches while experimental results of the whole test set are reported. All experiments are repeated 10 times. ADAM [Kingma and Ba, 2014] is the optimizer we use. In addition, we select ReLU as the activation function (ϕ_t, ϕ_x and ϕ_{out}). In terms of the initialization, the cell state of LSTM, weights of LSTM and weights of dense layers are set to be 0, the truncated normal distribution and the Xavier initialization [Glorot and Bengio, 2010], respectively. Grid search is used for optimal hyperparameters. Specifically, we search learning rate in $\{0.01, 0.001, 0.0001\}$, number of units in dense layers in $\{1, 10, 100\}$, LSTM state cell size in $\{32, 64, 128, 256\}$, batch size in $\{16, 32, 64\}$ and the number of noise samples per true event in $\{1, 2, 5, 10\}$. Similar to [Mnih and Teh, 2012], we adopt three strategies for the reparametrized normalizer z': (1) we set $z' = 1$ as constant; (2) we set z' as a single parameter to learn, which is also independent of $\mathcal{H}_i$; (3) we learn $z' = g^z(\boldsymbol{h}_i)$ as a function of the vector representation of $\mathcal{H}_i$.

Baselines To assess the effectiveness of the proposed framework, we compare INITIATOR with the following variants and state-of-the-art frameworks for training neural MTPP models. For a fair comparison, we use the same input layers, recurrent layers and output layers on vector representations for time, mark, CIF, and ground CIF. It worths to note that TPP models such as seismic [Zhao *et al.*, 2015] cannot be considered as baselines as their inability to model mark types along with timing information.

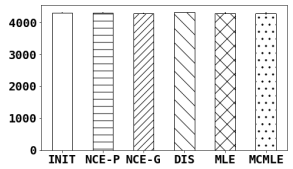
- **NCE-P**: A variant of INITIATOR in which we sample $t'_{i,k}$ from homogeneous Poisson process.
- **NCE-G**: A modified INITIATOR in which $t'_{i,k}$ are *i.i.d.* samples from Gaussian distributions.
- **DIS**: Similar to [Xiao *et al.*, 2017], DIS trains a MTPP model with discriminative loss functions, i.e., MSE on time and cross entropy on marks.



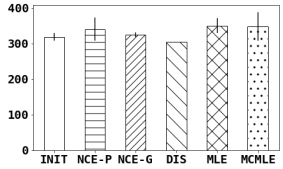
(a) Citi Bike: RMSE



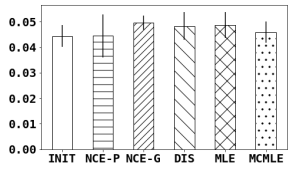
(b) Citi Bike: MAE



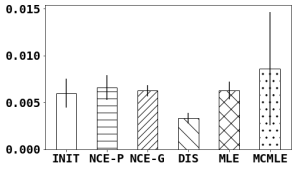
(c) Retweet: RMSE



(d) Retweet: MAE

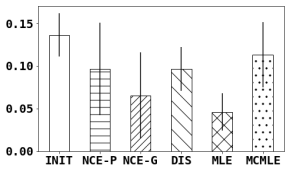


(e) Financial: RMSE

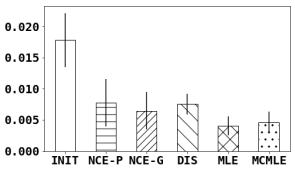


(f) Financial: MAE

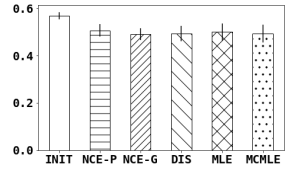
Figure 3: Time prediction results of the three datasets, the error bars represent one standard deviation.



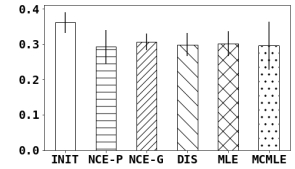
(a) Citi Bike: Micro-F1



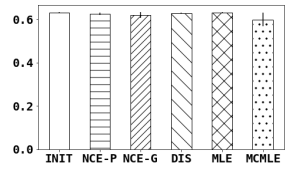
(b) Citi Bike: Macro-F1



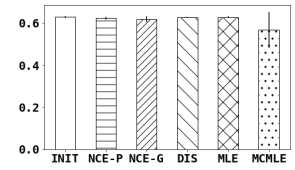
(c) Retweet: Micro-F1



(d) Retweet: Macro-F1



(e) Financial: Micro-F1



(f) Financial: Macro-F1

Figure 4: Mark prediction results of the three datasets, the error bars represent one standard deviation.

- **MLE**: Similar to [Du *et al.*, 2016], with an integrable ground CIF, MLE maximizes the likelihood of time exactly and minimizes the cross entropy on marks.
- **MCMLE**: As in [Mei and Eisner, 2016], MCMLE trains a MTPP model by maximizing likelihood approximately through Monte Carlo sampling.

Evaluation metrics: For time prediction, we evaluate different methods by the root mean squared error (RMSE) and the mean absolute error (MAE). For mark prediction, only unidimensional discrete marks are in the datasets. We use two metrics for classification: micro-F1 and macro-F1.

4.3 Experimental Results and Discussion

We conduct experiments with the three datasets on two research tasks: (1) time prediction; and (2) mark prediction. The comparison results w.r.t. the time prediction are shown in Figure 3 and the results w.r.t. the mark prediction are presented in Figure 4. We observe the followings:

- In nearly all cases, the proposed training framework INITIATOR outperforms its variants and the state-of-the-art approaches for both prediction tasks measured by the four metrics. We conduct one-tailed T-test to compare the performance of INITIATOR and other methods. T-test results indicate that INITIATOR is significantly better than the baselines with a significant level of 0.05.
- Benefiting from the adaptive noise generation, INITIATOR performs better than NCE-P and NCE-G as the noise samples generated by INITIATOR forces the MTPP model to capture more from the data distribution.
- INITIATOR outperforms MLE in most of the cases as MLE specifies an integrable function as its ground CIF, which limits the functional space MLE can search.
- Results show that INITIATOR is better than MCMLE. This is because Monte Carlo sampling can lead to biased estimations of the likelihood function, while Theorem 1 shows that INITIATOR estimates the likelihood in a more principled way.

5 Conclusion

In this work, we propose INITIATOR, a novel training framework to address the limitations of existing approaches based on MLE for MTPP models. Specifically, INITIATOR leverages the principle of NCE to compute the intractable likelihood functions (a bottleneck of existing training framework) by learning a set of parameters. Theoretically, we show that INITIATOR has strong connections to exact MLE. Practically, we introduce an algorithm of adaptive noise sample generation for INITIATOR and examine the effectiveness of INITIATOR with an instantiation by the state-of-the-art deep learning techniques. Experimental results show the superior performance of INITIATOR in predicting both time and mark for the upcoming events. For future work, one direction is to develop novel frameworks to train complex MTPP models for correlated or heterogeneous event sequences. Another direction is to interpret the predictions made by the black-box neural MTPP models.

Acknowledgments

This material is based upon work supported by the Natural Science Foundation (NSF) grant 1614576 and the Office of Naval Research (ONR) grant N00014-17-1-2605.

References

[Arjovsky and Bottou, 2017] Martin Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862*, 2017.

[Bromley *et al.*, 1994] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a" siamese" time delay neural network. In *NIPS*, pages 737–744, 1994.

[Du *et al.*, 2016] 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 *KDD*, pages 1555–1564. ACM, 2016.

[Flanders, 1973] Harley Flanders. Differentiation under the integral sign. *The American Mathematical Monthly*, 80(6):615–627, 1973.

[Gao and Liu, 2015] Huiji Gao and Huan Liu. Mining human mobility in location-based social networks. *Synthesis Lectures on Data Mining and Knowledge Discovery*, 7(2):1–115, 2015.

[Gao *et al.*, 2015] Shuai Gao, Jun Ma, and Zhumin Chen. Modeling and predicting retweeting dynamics on microblogging platforms. In *WSDM*, pages 107–116. ACM, 2015.

[Glorot and Bengio, 2010] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, pages 249–256, 2010.

[Goodfellow, 2014] Ian J Goodfellow. On distinguishability criteria for estimating generative models. *arXiv preprint arXiv:1412.6515*, 2014.

[Guo and Shakarian, 2016] Ruocheng Guo and Paulo Shakarian. A comparison of methods for cascade prediction. In *ASONAM*, pages 591–598. IEEE, 2016.

[Gutmann and Hyvärinen, 2010] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *AISTATS*, pages 297–304, 2010.

[Hawkes, 1971] Alan G Hawkes. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1):83–90, 1971.

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

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

[Kingman, 1993] John Frank Charles Kingman. *Poisson processes*. Wiley Online Library, 1993.

[Lian *et al.*, 2015] Wenzhao Lian, Ricardo Henao, Vinayak Rao, Joseph Lucas, and Lawrence Carin. A multitask point process predictive model. In *ICML*, pages 2030–2038, 2015.

[Liu *et al.*, 2017] Xin Liu, Junchi Yan, Shuai Xiao, Xiangfeng Wang, Hongyuan Zha, and Stephen M Chu. On predictive patent valuation: Forecasting patent citations and their types. In *AAAI*, pages 1438–1444, 2017.

[Mei and Eisner, 2016] Hongyuan Mei and Jason Eisner. The neural hawkes process: A neurally self-modulating multivariate point process. *arXiv preprint arXiv:1612.09328*, 2016.

[Mikolov *et al.*, 2013] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119, 2013.

[Mnih and Teh, 2012] Andriy Mnih and Yee Whye Teh. A fast and simple algorithm for training neural probabilistic language models. In *ICML*, pages 419–426. Omnipress, 2012.

[Rasmussen, 2011] Jakob Gulddahl Rasmussen. Temporal point processes the conditional intensity function. *Lecture Notes, Jan*, 2011.

[Shen *et al.*, 2014] Hua-Wei Shen, Dashun Wang, Chaoming Song, and Albert-László Barabási. Modeling and predicting popularity dynamics via reinforced poisson processes. In *AAAI*, volume 14, pages 291–297, 2014.

[Snyder and Miller, 2012] Donald L Snyder and Michael I Miller. *Random point processes in time and space*. Springer Science & Business Media, 2012.

[Tabibian *et al.*, 2017] Behzad Tabibian, Isabel Valera, Mehrdad Farajtabar, Le Song, Bernhard Schölkopf, and Manuel Gomez-Rodriguez. Distilling information reliability and source trustworthiness from digital traces. In *WWW*, pages 847–855, 2017.

[Valera and Gomez-Rodriguez, 2015] Isabel Valera and Manuel Gomez-Rodriguez. Modeling adoption and usage of competing products. In *ICDM*, pages 409–418. IEEE, 2015.

[Xiao *et al.*, 2017] Shuai Xiao, Junchi Yan, Xiaokang Yang, Hongyuan Zha, and Stephen M Chu. Modeling the intensity function of point process via recurrent neural networks. In *AAAI*, pages 1597–1603, 2017.

[Zhao *et al.*, 2015] 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 *KDD*, pages 1513–1522. ACM, 2015.