Knowledge-Based Systems 228 (2021) 107247

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Astrologer: Exploiting graph neural Hawkes process for event propagation prediction with spatio-temporal characteristics

Haizhou Du [a,*], Yan Zhou [a], Yunpu Ma [b], Shiwei Wang [a]

[a] Shanghai University of Electric Power, China
[b] Ludwig-Maximilians-Universität München, Germany

ARTICLE INFO

Article history:
Received 10 August 2020
Received in revised form 18 June 2021
Accepted 18 June 2021
Available online 24 June 2021

Keywords:
Hawkes process
Graph neural network
Forecasting
Spatio-temporal events

ABSTRACT

The prediction of event propagation has received extensive attention from the knowledge discovery community for applications such as social network analysis. The data describing these phenomena are multidimensional asynchronous event data that affect each other and show complex dynamic patterns in the continuous-time domain. The study of these dynamic processes and the mining of their potential correlations provide a foundation for the application of event propagation forecasting.

However, conventional forecasting methods often make strong assumptions about the generative processes of the event data that may or may not reflect the reality, and the strong parametric assumptions also restrict the expressive power of the respective processes. Therefore, it is difficult to capture both the temporal and spatial effects of past event sequences.

Most of the existing methods capture the intensity function of the Hawkes processes conditioned only on the historical events while ignoring the spatial information and the influences among different events.

In this work, we propose the Astrologer, a graph neural Hawkes process that can capture the propagation of events from historical events on graph. The underlying idea of Astrologer is to incorporate the conditional intensity function of the Hawkes processes with a graph convolutional recurrent neural network. Using both synthetic and real-world datasets, we show that, Astrologer can learn the dynamics of event propagation without knowing the actual parametric forms. Astrologer can also learn the dynamics and achieve better predictive performance than other parametric alternatives based on particular prior assumptions.

1. Introduction

Event propagation forecasting is currently a rapidly advancing field that is relevant for a wide variety of applications. Many real-world applications generate an abundance of graph data that can be regarded as a set of event sequences with spatio-temporal information. For example, in retweeting social network, the social network can be viewed as a graph and users activities such as retweet, reply, and mention can be regarded as markers. As shown in Fig. 1, each geometric object in sequences represents a marker, and markers in different sequences can influence the appearance of the appearing markers of neighboring nodes. *i.e.*, the timestamps and markers of future events are influenced by past events and the graph structure. The graph structure can carry rich information about the event attributes *e.g.*, marker, content, participator, and the timestamp indicates when the event occurs.

Since event sequences are ubiquitous in applications, forecasting the future events' sequences can help organizations with capacity planning, goal setting, and anomaly detection and have great practical significance. For example, [1] proposed a novel spatio-temporal graph convolutional network that can predict the picking up and returning demand for the public bike-sharing program to solve the chaotic parking problem. Also, [2] captured the spatio-temporal correlations among earthquakes for better forecasting and early warning.

For forecasting event sequences, the Hawkes process is a useful mathematical method with self-excitation characteristics, where the excitations exponentially decaying with time. However, the standard Hawkes process only consider independent event sequences which appear inconsistent with real-world data. Therefore, performance might be deteriorated when real-world event sequences are modeled independently. To address this limitation, Liniger et al. [3] proposed a method that can model events in a sequence that occur prior to time t. Meanwhile, these events may influence the occurrence of new events in other sequences after time t. This approach can capture not only the

* Corresponding author.
E-mail address: duhaizhou@shiep.edu.cn (H. Du).

https://doi.org/10.1016/j.knosys.2021.107247
0950-7051/

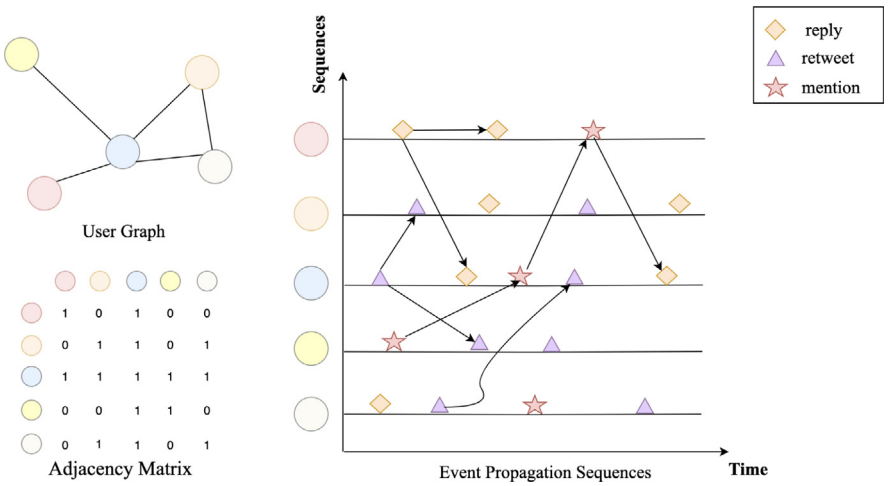

Fig. 1. A real-world example of event propagation forecasting. The left panel depicts the social connection graph of users and the corresponding adjacency matrix. The right panel shows the event propagation sequences where each geometric object in the sequence represents a user activity which can be regarded as a marker. Three different geometric objects represent reply, retweet, and mention, respectively, and they can affect each other. The y-axis represents user nodes in the social network, and each user's activity is characterized by its recurrent event sequences. Given the event propagation sequences, can we capture the spatial–temporal information to predict the next marker and timestamp of each user?.

local self-exciting dynamics but also the global influence from other sequences. Other works [4–6] proposed several parametric forms for the conditional intensity function of the Hawkes processes. Nevertheless, these methods only model the event sequences with small sets and cannot effectively capture the global interactions between several event sequences. Additionally, these methods face computational limitations in calculations for the large-dimensional events [7]. Recently *e.g.*, Shang et al. [8] proposed a novel parametric form for the Hawkes processes' conditional intensity function. This method attributes the advances in this research direction in part to use of the Graph Neural Networks as in [9,10]. However, the use of fixed parametric forms that restrict the expression capability for arbitrarily distributed event data is a major limitation of the Hawkes processes' parametric forms. It poses a risk of model underfitting.

Conventional Hawkes process models, including the state-of-the-art methods proposed in [11–13] use the historical events information as the model input and forecast. The main difference between the state-of-the-art methods and the conventional methods is that the state-of-the-art methods approximate the conditional intensity function by the recurrent neural networks.

Despite the success in modeling events propagation sequences with the Hawkes processes, the following three main challenges for modeling the discrete event sets in continuous time still remain unsolved.

- **Non-parametric forms of the conditional intensity function.** How to design an interpretable model that can replace the fixed parametric forms of the conditional intensity function with the non-parametric forms?
- **Learning from spatio-temporal data.** How to design an effective model that can learn information from large spatio-temporal datasets?
- **Accuracy of forecasting.** How to design a model with good forecasting performance, *i.e.*, how to describe the distributions of the events in the future correctly?

In this paper, we propose a novel forecasting model of event propagation sequences called Astrologer. Our approach transforms the fixed parametric forms of the conditional intensity function to the non-parametric forms to predict the next event's marker and timestamp on graph-based data. The main concept of the model is to capture the spatial information from the graph by geometric deep learning and the temporal information by gated recurrent units. Then we use the neural network's output to approximate the conditional intensity function of the Hawkes process and recall the inverse method by the simulation approach for timestamps forecasting.

More specifically, compared with the conventional forecasting models of event propagation sequences, our work makes the following contributions:

- We conduct a systematic study on the vital problem of event propagation sequences forecasting with spatio-temporal characteristics, and design Astrologer, which is a novel non-parametric model that leverages emerging graph neural Hawkes process techniques to address this problem.
- To the best of our knowledge, Astrologer is the first method to approximate the conditional intensity function by geometric deep learning and recall the inverse method by the simulation approach for timestamps forecasting.
- We use the graph neural network to capture the graph structure *i.e.*, the spatial information, and the learned graph representations can be integrated into the Astrologer.
- To verify our model's efficiency, we evaluate Astrologer on two synthetic datasets and three real-world datasets, including the ATMs dataset, the Stack Overflow dataset, and the Financial dataset. The results show that our model outperforms all of the baseline methods, demonstrating the superiority of Astrologer for event forecasting.

The rest of the paper is organized as follows. We discuss related work in Section 2. Section 3 introduces Hawkes process background and Graph Convolution Network. Section 4 introduces the details of the Astrologer. In Section 5, we evaluate the predictive performance of the Astrologer on both real-world and synthetic datasets, including the design of model parameters, and forecasting results analysis. Finally, we conclude the paper in Section 6.

2. Literature review

The Hawkes processes with conditional intensity functions are stochastic processes with self-excitation characteristics for modeling event propagation sequences. *i.e.*, Hawkes processes supposed that an event that happened could raise some other

events in a finite time interval to occur. Extensive literature has been done to model the dynamic propagation of event sequences by the Hawkes processes. There are three main types of Hawkes processes:

2.1. Conventional Hawkes processes

With the self-excitation characteristics, the initial motivation of the Hawkes processes was to model the earthquakes and their aftershocks [14]. As an effective method for modeling sequential event data, Yang et al. [15] integrated the Hawkes processes with a language model to track the corresponding diffusion network. Li et al. [16] proposed a model for simultaneously identifying and labeling search tasks by integrating topic models with Hawkes processes. Li et al. [17] modeled the influence between householders' energy usage behaviors directly through a novel probabilistic model, which combined topic models with Hawkes processes. In data mining, *e.g.*, social infectivity learning, Stomakh et al. [18] developed a reconstruction model based on Hawkes processes that predicted the unknown participants in a portion of crime events with gang network. Li et al. [19] proposed a probabilistic model based on mixtures of the Hawkes processes that simultaneously tackled event attribution and network parameter inference. In event forecasting tasks, *e.g.*, Ertekin et al. [5] used the Hawkes processes based model to predict power-grid failures over a short-term horizon and to provide a cost/benefit analysis of different proactive maintenance programs. However, there is a major limitation of the Hawkes processes that they often draw various parametric assumptions about the latent dynamics governing the generation of the observed point patterns [12,20].

2.2. Deep learning Hawkes processes

In the conventional Hawkes processes, the conditional intensity function and the density have to be designed in the model and then needs to find a tailor-made learning algorithm for modeling it. Meanwhile, the fixed parameters may inhibit the expression of the model. With the repaid development of deep learning, there are different ways based on the recurrent neural network and Hawkes processes. Du et al. [12] proposed a method that could automatically generate an efficient representation from the events that happened past without a prior assumption about the parameters. The work by Xiao et al. [11] was the first time to approximate the conditional intensity function with fully implicit mapping. Mei et al. [13] discovered the assumptions of Hawkes processes often seem to violate the real-world patterns and proposed a model based on a novel continuous-time LSTM. The model could capture both excitation and inhibition from past events. Meanwhile, the model was designed with a decaying part so that the model did not need to encode the occurrence time intervals as the input of LSTM. Further, the model could avoid the positive probability of extinction. However, these models were costly when predicting large sequences.

To improve the learning efficiency and extend to millions of related marks, Turkmen et al. [21] proposed a model that uses recurrent neural networks to capture the complex temporal dependency patterns among different markers, while self-excitation dynamics within each marker were modeled with the Hawkes processes. Existing literature such as variations of the Multivariate Hawkes processes ignored the spatial information when they modeled the events with spatio-temporal characteristics and became computational inhibitive in most real-world applications involving large dimensions, *e.g.*, these methods were limited by number of events types and the data form. Thus, the Multivariate Hawkes processes cannot model the dynamics of graph sequences, which is the future tendency. To address this limitation, Liu et al. [22] proposed a graph regularization method to integrate the spatial information into the Multivariate Hawkes processes for learning influence matrix between different event sequences. Another method proposed by Wu et al. [23], leveraged the spatial information from graph representation learning, then the learned information was integrated into the embedded event history as side information. In [8], Hawkes processes had been integrated into geometric deep learning to model the correlated individual event sequence better.

2.3. Adversarial learning and reinforcement learning for Hawkes processes

The conventional methods learned via maximum likelihood approaches may prone to failure in multi-modal distributions of sequences. To address this issue, Xiao et al. [24] proposed an intensity-free approach that transformed nuisance processes to a target one and trained the model using a likelihood-free leveraging Wasserstein distance from Hawkes processes. Yan et al. [25] proposed adversarial classification loss, which induced the Wasserstein distance loss to add sharpness to the smooth effect inherently caused by the MSE loss. To solve the MLE disadvantage, Li et al. [26] modeled the generation of each event as the action taken by a stochastic policy, and then the policy was parameterized into a recurrent neural network and gradually improved policy to mimic the observed event distribution. Upadhyay et al. [27] proposed a model based on reinforcement learning, which allowed for arbitrarily complex reward functions. The model is applied to two different *i.e.*, applications in viral marketing and personalized teaching, and the performance is good.

In summary, existing methods either ignore spatial information from graph structure or have a computational consumption with long dependency event sequences. Therefore, we propose a novel forecasting model that can capture the spatial–temporal information and deal with long dependency event sequences.

3. Background

This section describes our method by first introducing the background of the Hawkes process and its self-exciting characteristics. Then, we present the Graph Convolution Network (GCN) that incorporates spatial knowledge by adding graph constraints into the Hawkes process' model.

3.1. Background of the Hawkes process

The Hawkes process [28] is a stochastic process with self-excitation characteristics used to capture the information from sequential asynchronous discrete events occurring in continuous time. The meaning of the asynchronous is that the time interval between consecutive events may not be the same [29]. The conditional intensity function provides the self-excitation characteristics of the Hawkes process. A univariate Hawkes process is suitable for modeling the mutual excitation between events such as the occurrences of earthquakes at a particular location [8]. The conditional intensity function of a univariate Hawkes process is defined as:

$$\lambda(t) = \mu + \sum_{h:t_h<t} \alpha \exp\left(-\delta\left(t - t_h\right)\right), \tag{1}$$

where $\mu > 0$ is the baseline intensity of the event, $\alpha > 0$ is the coefficient that scales the influence of each previous event, and $\delta > 0$ is the decay rate of that excitation of the historical events that can excite future events in a finite time interval.

When events occur, the intensity is elevated to various degrees but will decay toward its baseline rates μ. In the real world, it is desirable to model a collection of correlated event sequences such as earthquakes at N locations, *i.e.*, each event sequence in a location is modeled as a Hawkes process:

$$\lambda_k(t) = \boldsymbol{\mu}_k + \sum_{h:t_h<t} \boldsymbol{\alpha}_{k_h,k} \exp\left(-\boldsymbol{\delta}_{k_h,k}\,(t-t_h)\right), \tag{2}$$

where $k = 1, \ldots, N$ is the index of sequences, $\boldsymbol{\mu}_k > 0$ is a vector of size N, which represent the baseline intensity of each sequence. $\boldsymbol{\alpha}_{k_h,k} > 0$ is the degree to which an event of type j initially excites type k, and $\boldsymbol{\delta}_{k_h,k}$ is the decay rate of that excitation [13].

The temporal information in event dynamic propagation sequences can be well captured by the Hawkes process with its conditional intensity function *i.e.*, the stochastic model for the next event coming time given all past events. Given the historical events $\mathcal{H}_t$, $\lambda\,(t)$ represent the probability for the occurrence of a new event in a small window $[t, t+dt)$:

$$\lambda^*(t)dt = \mathbb{P}\{\text{ event in } [t, t+dt) \mid \mathcal{H}_t\} \tag{3}$$

Given the conditional density function $f^*(t)$, the conditional intensity function can be represented as:

$$\lambda(t)dt = \frac{f^*(t)dt}{S^*(t)} = \frac{f^*(t)dt}{1-F^*(t)}. \tag{4}$$

By invoking the elegant relation between the conditional intensity function and the conditional density function, the conditional density function can finally be represented by:

$$f^*(t) = \lambda(t)\exp\left(-\int_{t_{i-1}}^{t}\lambda(s)ds\right). \tag{5}$$

In [30], conditional intensity function of Hawkes process is used to capture the phenomena of interests such as prediction of the next event time.

3.2. Background of graph convolutional network

Graph Convolutional Network(GCN) is also known as shift invariant or space invariant artificial neural networks, based on their shared-weights architecture and translation invariance characteristics. It generalizes the operation of convolution from grid data to graph data. The main idea is to generate a node v's representation by aggregating its own features x_v and neighbors features x_u, where $u \in N(v)$. Specifically, a graph Laplacian L admits a spectral eigendecomposition of the form :

$$L = U\Lambda U^\top \tag{6}$$

where $U = [u_0; \cdots; u_{n-1}] \in R^{n\times n}$ is the orthonormal matrix and is the complete set of the orthonormal eigenvectors $[u_l]_{l=0}^{n-1} \in R^n$, and $\Lambda = diag([\lambda_0, \ldots, \lambda_{n-1}]) \in R^{n\times n}$ is the diagonal matrix with the associated ordered real non-negative eigenvalues $[\lambda_l]_{l=0}^{n-1}$. In particular, eigenvectors are known as the Fourier atoms in classical harmonic analysis and eigenvalues are usually interpreted as the frequencies of the graph. Given a function $x = (x_0, \ldots, x_{n-1})^T \in R^n$ on the vertices of the graph, the graph Fourier transform on graph $\mathcal{G}$ is defined as

$$\widehat{x} = (\widehat{x}(\lambda_0), \ldots, \widehat{x}(\lambda_{n-1}))) = U^T x \in R^n \tag{7}$$

and its inverse is $x = U\widehat{x}$. Thus, the spectral convolutional of function x and convolutional kernel function y on graph $\mathcal{G}$ is given by

$$(x \odot y)_G = U \cdot diag[(\widehat{y}(\lambda_0), \ldots, \widehat{y}(\lambda_{n-1}))] \cdot U^T x, \tag{8}$$

where $\odot$ is the element-wise Hadamard product. It should be mentioned that convolutions are by definition linear operators that diagonalize in the spectral domain, according to the definition of Discrete Fourier Transform and the Convolution Theorem [31]. Therefore, a typical GCN layer can be defined as $x_{output} = \sigma((x \odot y)G)$, where $diag([\widehat{y}(\lambda_0), \ldots, \widehat{y}(\lambda_{n-1})])$ represents parameters of learnable filters in the spectral domain, and σ denotes the activation function (*e.g.*Relu) which is applied on the vertex wise function values [8].

Based on this background information, in this paper, we propose a novel Hawkes process model that can capture the spatial–temporal information from graph data and then can be used for event propagation forecasting tasks.

Our work based on the following two motivations:

- The first motivation is that our model can utilize the spatio-temporal information with a non-parametric form. The methods reported in the literature either ignore the spatio-temporal information or utilize it to learn the fixed parameters of the conditional intensity function. However, fixed parameters may limit the model expression. Since the spatio-temporal data are frequently generated in various fields, it is vital to propose a model to address this limitation.
- Another motivation is to design a model that allows for substantially faster simulation and forecasting. The traditional algorithm for sampling from a Hawkes process is Ogata's thinning method [32]. However, this algorithm specifies the sampling routine and incurs the computational cost of rejected points. Thus prediction by Ogata's thinning method is computationally expensive. We seek to construct a model that can achieve similar results in less time.

4. Astrologer design

In this section, we first define the problem, and then review the proposed model and introduce our model step by step.

4.1. Problem definition

The temporal characteristics of the Hawkes processes are determined by the parametric form of the conditional intensity function. However, it is challenging to choose the parameters correctly when both the spatial and temporal information is taken into account. We propose an efficient model that can capture spatial–temporal information and make predictions without any prior knowledge to address this challenge. The goal of our model is to predict event propagation based on the historical information of the real-world applications with spatio-temporal characteristics. Given a set of event sequences $\mathcal{C} = \left\{\mathcal{S}^1, \mathcal{S}^2, \ldots\right\}$, each $\mathcal{S}^j = \left(\left(t_1^j, y_1^j\right), \left(t_2^j, y_2^j\right), \ldots\right)$ is a sequence with event occurrence time t_i and marker y_i. Instead of learning a fixed parameter of the conditional intensity function for each application, we seek to predict the future event by approximating the conditional intensity function with a geometric neural network and a recurrent neural network that can apply in a diverse range of domains such as financial transactions, and social networks such as Twitter.

4.2. Overview of Astrologer architecture

By carefully investigating the various forms of the conditional intensity function [8,12,22], we first learn the graph's structure and then combine the learned embedding with temporal information. Finally, we forecast the events that will occur in the future based on the spatial–temporal information.

To model the event dynamic propagation and make predictions regarding the marker y_i and timestamp t_i, we propose a novel model that consists of the following two stages:

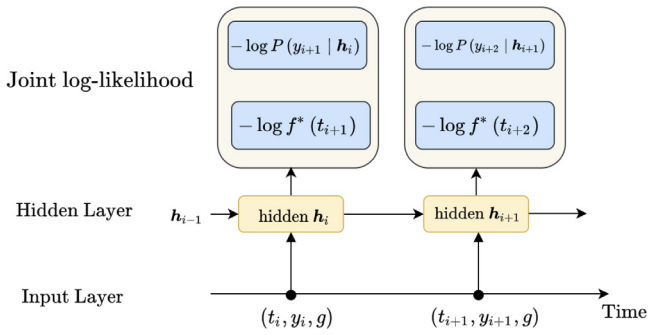

Fig. 2. The illustration of Astrologer model. Given the sequence $\boldsymbol{S} = \left((t_i, y_i)_{i=1}^n\right)$, each pair of sequence and spatial information learned from GCN are fed into Astrologer model where $\boldsymbol{h}_i$ up to the timestamp t_i learns a nonlinear dependency over spatial–temporal information from past events.

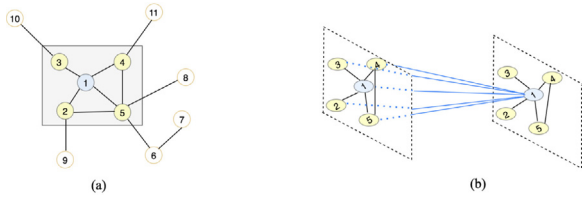

Fig. 3. Node 1 is assumed as a user in a social network. (a) The yellow nodes indicate the users connected to node 1. (b) We obtain the spatial information by the graph convolution network from the topological relationship between node 1 and the surrounding users.

- **Graph Embedding Stage.** The first step is the graph embedding, where the graph convolutional neural network of the model learns the information from the input data, and then the embedding vectors are fed into the second stage.
- **Graph Neural Hawkes Process Stage.** Here, an Astrologer model is trained based on the embedding vectors from the first step, event markers y_i and the temporal information. Then, the model's output is used to approximate the Hawkes processes' conditional intensity function, and then the inverse method is recalled by the simulation approach to make predictions.

An illustration of Astrologer is shown in Fig. 2. We describe the details of the Graph Embedding and Graph Neural Hawkes Process (GNHP) below.

4.3. Graph embedding stage

In the Fourier domain, the GCN model constructs a filter used to capture the graph's spatial features between the nodes and act on each node of the graph and its first-order neighborhood. After that, the GCN model can be constructed by stacking multiple convolutional layers.

As shown in Fig. 3, we assume that node 1 is a user in a social network whose behavior may be influenced by interpersonal relationships. The GCN can obtain spatial information via multiple convolutions from topology.

Generally, we adopt 2-layer GCN to capture the graph spatial features, with the GCN layer defined as [10]:

$$H^{(0)} = \widehat{A}XW^{(0)}, \tag{9}$$

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right), \tag{10}$$

where $H^{(0)}$ is the output of the first GCN layer, $\tilde{A}$ is the adjacency matrix A of the input graph $\mathcal{G}$ with added self-connections and X represents the features matrix, $\widetilde{D}$ is the degree matrix of the graph $\mathcal{G}$, $\sigma(\cdot)$ represents the activation function, and $W^{(l)}$ represents the weight matrix.

4.4. Graph neural Hawkes process stage

Given the past dynamic propagation sequence $\boldsymbol{S} = \left((t_i, y_i)_{i=1}^n\right)$ and current spatial information from GCN, we propose a stage of GNHP based on Gated Recurrent Units (GRU) to predict the next timestamp and marker by iterating the following five components.

- **Input Embedding** As shown in Fig. 2, the information from past events includes $\left\{\boldsymbol{y}_{i-1}, \boldsymbol{g}, \boldsymbol{t}_{i-1}\right\}$ as a triple. The first term is a one-hot vector of a node in $\mathcal{G} = (V, E)$, *i.e.*, the marker of the event. The second term is a vector, learned from the GCN, that represents the corresponding node. The last term is the embedding of the occurrence time of the current event t_{i-1}.
  The one-hot vector is projected into a latent space, in order to make the representation of the node more compact and efficient, by an embedding layer with the weight matrix $\boldsymbol{W}_{em}$ and bias $\boldsymbol{b}_{em}$, *i.e.*, $\boldsymbol{y}_i = \boldsymbol{W}_{em}^\top y_i + \boldsymbol{b}_{em}$. Then, node spatial features and temporal features are embedded into a common feature space $\mathbb{R}^H$ with the weight matrices $\boldsymbol{W}_g$ and $\boldsymbol{W}_t$, respectively.
- **Hidden Layer** As shown in Fig. 4, the right side shows the details of the GNHP cell, where $\boldsymbol{h}_{i-1}$ denotes the hidden state of t_{i-1}, $\boldsymbol{r}_t$ and $\boldsymbol{u}_t$ is the reset gate and update of the set of GRU. The computational processes performed after receiving the current input and the hidden state of $\boldsymbol{h}_{i-1}$ are described as follows:

$$\boldsymbol{u_i} = \sigma\left(\boldsymbol{W_u}\left[\boldsymbol{g}, \boldsymbol{y_{i-1}}, t_{i-1}, \boldsymbol{h_{i-1}}\right] + \boldsymbol{b_u}\right) \tag{11}$$

$$\boldsymbol{r_i} = \sigma\left(\boldsymbol{W_r}\left[\boldsymbol{g}, \boldsymbol{y_{i-1}}, t_{i-1}, \boldsymbol{h_{i-1}}\right] + \boldsymbol{b_r}\right) \tag{12}$$

$$\boldsymbol{c}_i = \tanh\left(\boldsymbol{W}_c\left[\boldsymbol{g}, \boldsymbol{y}_{i-1}, t_{i-1}, (\boldsymbol{r_i} * \boldsymbol{h_{i-1}})\right] + \boldsymbol{b_c}\right) \tag{13}$$

$$\boldsymbol{h}_i = \boldsymbol{u}_i * \boldsymbol{h}_{i-1} + (\boldsymbol{1} - \boldsymbol{u}_i) * \boldsymbol{c}_i, \tag{14}$$

  where $\boldsymbol{g}$ denotes the spatial information captured by GCN, $\boldsymbol{k}$ denotes the marker embedding of k, $\boldsymbol{t}$ denotes the time interval from the previous event, $\boldsymbol{W}$ denotes the weights and $\boldsymbol{b}$ denotes the bias.
  Given the event dynamic propagation historical embedding, we can compute the event propagation probability $P(y_{i+1}|\mathcal{H}_n)$ and approximate the conditional intensity function of the Hawkes process.
- **Event Propagation Probability** Given $\boldsymbol{h}_i$, we compute the $P(y_{i+1}|\mathcal{H}_i)$ as follows:

$$P(y_{i+1} = k|\boldsymbol{h}_i) = \frac{\exp\left(\boldsymbol{V}_{k,:}^h \boldsymbol{h}_i + b_k^h\right)}{\sum_{k=1}^N \exp\left(\boldsymbol{V}_{k,:}^h \boldsymbol{h}_i + b_k^h\right)} \tag{15}$$

  where N is the number of the nodes of graph $\mathcal{G}$, b_k^h is the bias and $\boldsymbol{V}_{k,:}^h$ is the kth row of matrix $\boldsymbol{V}$.
- **Conditional Intensity Function** Given $\boldsymbol{h}_i$, we approximate the conditional intensity function as follows:

$$\lambda^*(t) = \exp\Big(\underbrace{\boldsymbol{v}^{t^\top} \cdot \boldsymbol{h}_i}_{\text{mutual-excitation}} + \underbrace{w^t\,(t - t_i) + \mu^t}_{\text{local self-excitation}}\Big), \tag{16}$$

  where $\boldsymbol{v}^{t^\top}$ is a column vector, w^t and μ^t are scalars. In greater detail:
  - The exponential function acts as a nonlinear transfer function to obtain a nonnegative conditional intensity function.

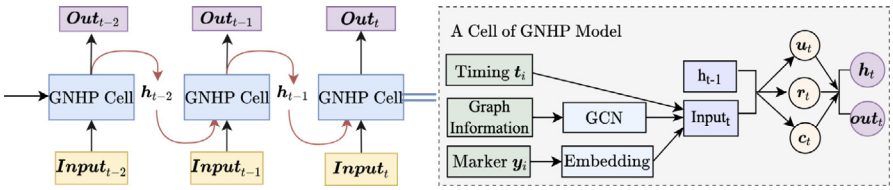

Fig. 4. Schematic of the overall process of the Astrologer model. The left part represents the structure of the GNHP cell, and the right part represents the specific workflow of the model cell.

  - The first term $\boldsymbol{v}^{t^\top} \cdot \boldsymbol{h}_i$ represents the accumulative influence from the temporal information and markers of the past dynamic propagation. It is equivalent to the influence of the occurred events in the traditional intensity function, *i.e.*, replacing the fixed parametric formulations with a nonlinear function.
  - The second term $w^t\,(t - t_i) + \mu^t$ gives a base intensity level for the Hawkes processes.

- **Prediction** *Markers prediction.* Given the probability in (15), the next marker y_{i+1} can be predicted as follows:

$$y_{i+1} = \arg\max_{k \in M} P\left(y_{i+1} = k | \boldsymbol{h}_i\right), \tag{17}$$

where M is the marker set for the event sequences.
Timestamps prediction. For a Hawkes process with the conditional intensity function $\lambda^*(t)$, the integrated conditional intensity can be defined as follows:

$$\Lambda(t) = \int_0^t \lambda^*(\tau) d\tau. \tag{18}$$

When we obtain the timestamp t_j and $\Lambda_{t_i}\left(t_{i+1}\right)$, we can recall the inverse method by the simulation approach for timestamps prediction [33]. That is, given $s \sim \text{Exp}(1)(i.e. x = -\log(1-u), u \sim$ uniform $(0, 1))$ then according to $\hat{t}_{i+1} = \Lambda_{t_i}^{-1}(x)$, we can calculate the $\hat{t}_{i+1}$ as:

$$\begin{aligned} t_{i+1} =& t_i - \frac{1}{w}\left(\boldsymbol{v}^{t^\top} \cdot \boldsymbol{h}_i + \mu\right) \\ &+ \frac{1}{w}\left(\log\left(\exp\left(\boldsymbol{v}^{t^\top} \cdot \boldsymbol{h}_i + \mu\right) - w \log(1-u)\right)\right) \end{aligned}. \tag{19}$$

For example, given a significance $\alpha = 0.1$, in theorem , it is 90% certain that $\hat{t}_{i+1}$ is in the interval $\left[\hat{t}_{i+1}^{\left(\frac{\alpha}{2}\right)}, \hat{t}_{i+1}^{\left(1-\frac{\alpha}{2}\right)}\right]$. We present the following time prediction Algorithm 1. In our model, we chose $u = 0.5$ which is the median of the interval $(0, 1)$.

Algorithm 1 Time Prediction

Require: Hidden state; Model parameters and output; Timestamp t_{i-1} and integrated conditional intensity $\Lambda(t)$.
Ensure: The occur timestamp of the next event.
1: Initialize $T \leftarrow 0$
2: $\Lambda\left(t_{i-1}, t_i\right) = \Lambda\left(t_i\right) - \Lambda\left(t_{i-1}\right)$
3: $s = -\log(1-u)$
4: $\hat{t}_i = \Lambda_{t_{i-1}}^{-1}(s)$
5: **for** *Nsamples* **do**
6: $\quad draw\ \mu \sim Unif(0, 1)$
7: $\quad T+ = \Lambda_{t_{i-1}}^{-1}(s)$
8: **end for**
9: $t_i \leftarrow T/N$
10: **return** t_i

4.5. Parameters learning

Given a collection of event sequences $\mathcal{C} = \left\{\mathcal{S}^j\right\}$, where $\mathcal{S}^j = \left(\left(t_i^j, y_i^j\right)_{i=1}^{n_j}\right)$, we can train the parameters by maximizing the joint objective function:

$$OPT = \sum_j \sum_i \left(\log P\left(y_{i+1}^j | \boldsymbol{h}_i\right) + \log f\left(t_{i+1}^j | \boldsymbol{h}_i\right)\right), \tag{20}$$

where the first term is the cross-entropy of the predicted marker and the second term is the temporal log-likelihood of the predicted timestamp. To maximize the objective function, we adopt the Back Propagation Through Time to train Astrologer. First, we chose the size b of Back Propagation Through Time(BPTT) and then we unroll Astrologer in Fig. 2 by b steps. In each training process, the sequences are truncated by b and every consecutive sample $\left\{\left(t_k^j, y_k^j\right)_{k=j}^{i+b}\right\}$ is provided to apply the feed-forward operation through the network and the parameters to the loss function are updated, respectively. After we unroll the model for b steps through time, the parameters of Astrologer will be updated sequentially in the back propagation stage. Meanwhile, the outputs are fed into joint loss functions that include the NLL(negative log-likelihood) of the predicted timestamp and the cross-entropy of the predicted marker. Finally, we apply Adaptive Moment Estimation (ADMA) [34] with mini-batch and several other methods that have been applied to train neural networks in [35]. The details of parameter learning are shown in Algorithm 2.

Algorithm 2 Parameters Learning

Require: All the training events $\mathcal{C} = \left\{\mathcal{S}^1, \mathcal{S}^2, \ldots\right\}$; parameters $\boldsymbol{v}^{t^\top}$, w^t and μ^t
Ensure: The conditional intensity of Hawkes process $\left\{\boldsymbol{x}^{(T)}\right\}$.
1: Initialize $\left\{\boldsymbol{x}^{(0)}\right\}$
2: **for** t := 0 to T **do**
3: $\quad$ Forward Propagation:
4: $\quad$ Apply SVD to generate feature matrix V_t and apply k-neighbors graph to generate adjacency matrix A_t
5: $\quad$ Apply 2-layers GCN on A_t producing output matrix O_t.
6: $\quad$ Apply GRU with a fully connected layer on the output matrix O_t producing small incremental update $\left\{\boldsymbol{dx}^{(t)}\right\}$.
7: $\quad$ Update $\left\{\boldsymbol{x}^{(t+1)} \leftarrow \boldsymbol{x}^{(t)} + \boldsymbol{dx}^{(t)}\right\}$
8: $\quad$ Back Propagation:
9: $\quad$ Apply Adam stochastic optimization algorithm to optimize joint objective function and update parameters $\boldsymbol{v}^{t^\top}$, w^t and μ^t
10: **end for**
11: **return** parameters $\boldsymbol{v}^{t^\top}$, w^t and μ^t to calculate the Hawkes process' conditional intensity function.

5. Evaluation

In this section, we evaluate the forecasting performance of the Astrologer on various synthetic and real-world datasets. We

train the model without any prior knowledge and compare the Astrologer with a baseline method and three state-of-the-art methods, *i.e.* RMTPP [12], NHP [13], and Intensity RNN [11]. In more detail, first, we introduce the datasets and experiments setup of our experiment. Then, we describe the details of the competing methods and model settings. Finally, we present and analyze the experimental results.

5.1. Experiments setup

All the experiments were executed on a single computer with the following specifications:

CPU: 2.8 GHz Intel Core i7 processor, 16GB 2133 MHz LPDDR3

Used software details are shown below:

Model implementation: Pytorch 1.1.0.post2, Numpy 1.16.6 and scipy 1.1.0 Operating System: macOS Catalina 10.15.4.

5.2. Experiment datasets

We evaluate our model on two synthetic datasets and three real-world datasets, both of which have time interactions between a set with markers and a set with timestamps. More details:

- **Synthetic Dataset**
  To prove the proposed model can be extended to other fields, we simulated a multivariate Hawkes process to generate a synthetic dataset. The parameters μ are generated from a uniform distribution on [0,0.001]. The infectivity matrix A is generated by $\mathbf{A} = \mathbf{U}\mathbf{V}^T$. $\mathbf{U}$ and $\mathbf{V}$ are both 1000×9 matrices with entries in $[(i-1)+1 : (i+1), i], i = 1, \ldots, 9$ sampled randomly from [0, 0.01] and all other entries were set to zero.
- **Self-correcting Dataset**
  We generated a synthetic dataset by the self-correcting process with the conditional intensity function

$$\lambda^*(t) = \exp\left(\mu t - \sum_{t_i<t} \alpha\right), \tag{21}$$

  where $\mu > 0, \alpha > 0$.
- **Financial Transaction Dataset**
  The dataset consists of collected a raw limited order book data from the New York Stock Exchange of the high-frequency transactions for a stock in one day [12]. It contains 0.7 million transaction records, each of which records the time (in millisecond) and the possible action (buy and sell). The types of activities are treated as markers.
- **ATMs (Automated Teller Machines) Dataset**
  The ATMs dataset [11] consists of 1554 ATMs, and is splited with two parts, one is the training data with 1085 ATMs and another is testing part with 469 ATMs. It recorded two main types of error and ticket from Sep.2014 to Mar.2015 in North America. Moreover, types of error are further divided into 6 subtypes: 1) printer (PRT), 2) cash dispenser module (CNG), 3) internet data center (IDC), 4) communication part (COMM), 5) printer monitor (LMTP), 6) miscellaneous *e.g.*hip card module, USB (MISC).
- **Stack OverFlow Dataset**
  Stack OverFlow is a website that provides a platform for users to ask for coding assistance. The website exploits badges to encourage user engagement and guide behaviors [36]. The badges are divided into 81 types. The data process is the same as [12]. Finally, the data from 2012-01-01 to 2014-01-01 consists of about 6 thousand users with about 480 thousand events where the badges is treated as markers.

5.3. Competing models

To evaluate the predictive performance of forecasting, we compare Astrologer with the following discrete-time models:

- **Intensity RNN.** The Intensity RNN is proposed in [11], which captures the past information by two recurrent neural networks. When an event sequence occurs, the model embeds the event type into the latent space and then inputs the embedding into an event-rnn. Meanwhile, the temporal information is input into another time-rnn, which is finally associated with the output of two recurrent neural networks or their variants, *e.g.*, LSTM.
- **The Recurrent Marked Temporal Point Processes.** The Recurrent Marked Temporal Point Process (RMTPP) is proposed by [12]. The authors proposed a unified model capable of modeling a general nonlinear dependency over the historical events of both the timestamps and the marker information by deep neural network.
- **The Neural Hawkes process.** The Neural Hawkes Process removed the restriction that the happened events have an independent influence on $\lambda_k(t)$. Rather than predict $\lambda_k(t)$ as a simple summation, the model used a deep neural network. It allows learning a complex dependence of the intensities on the number, order, and timing of events. In the new Hawkes process, these dynamics are caught by a hidden state vector $\mathbf{h}(t) \in (-1, 1)^D$, which in turn depends on a vector $\mathbf{c}(t) \in \mathbb{R}^D$ of memory cells in a continuous-time LSTM.
- **Long Short Term Memory with NLL loss.** We use the Long Short Term Memory(LSTM), which is trained by Astrologer loss with markers and timestamps information, to predict marker and timestamps respectively. There are two hidden layers of LSTM with 16 hidden units.

5.4. Generate graph of datasets

As suggested in [37], a graph can be constructed as an unweighted k-nearest neighbor graph in the space of features such as users features in Stack Overflow. In cases where features are not available, we build a two-dimensional matrix from the event sequences. Each entry indicates the total count of user and event interactions and applies SVD(Singular Value Decomposition) to get a latent feature vector for each user.

5.5. Model setting

We use a single layer GRU with Sigmoid gate activations and tanh activation for hidden representation. The embedding layer of event markers is fully connected, and it uses tanh activation, outputs a 16-dimensional vector. For graph structure embedding, we set the 2-layers GCN to capture the spatial information, and the size of the hidden layers is 16. We used the first ten events to predict the next event marker, and timestamp. Therefore, the length of the event sequence can be arbitrarily long. The hyperparameters of Astrologer we chose across all these datasets are as following: hidden layer size in {16, 32, 64}; learning rate in {0.1, 0.01, 0.001}; time length of inputs in {5, 10, 15, 30} and batch-size in {64, 256, 512, 1024}.

5.6. Evaluation metrics

We chose several prediction metrics to show the model performance. For time prediction, we use the RMSE (Root Mean Squared Error):

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(Y_t - \widehat{Y}_t\right)^2}, \tag{22}$$

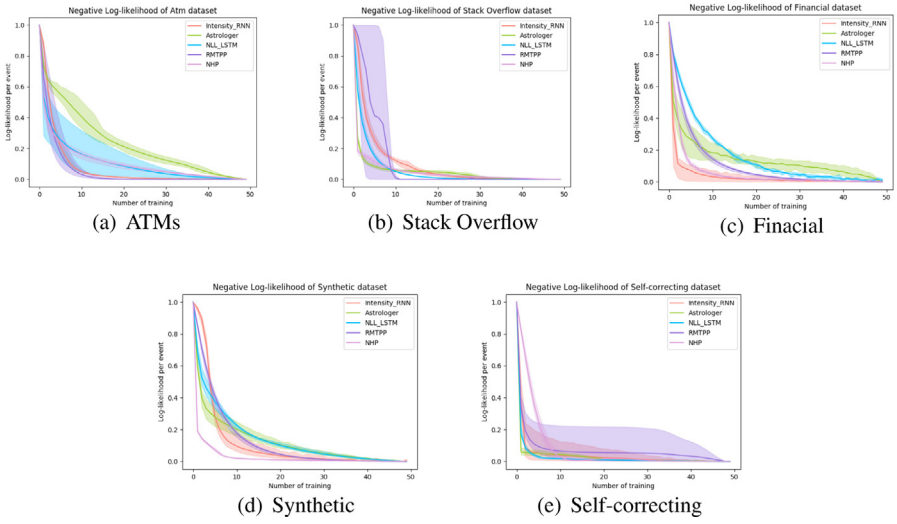

Fig. 5. The Negative log-likelihood of training with five times. The shaded part is the confidence interval of negative log-likelihood.

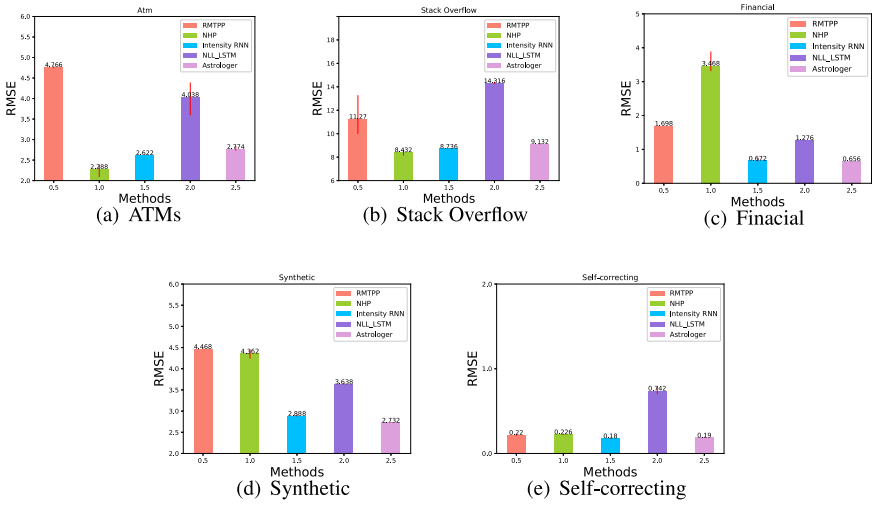

Fig. 6. Performance evaluation for predicting timestamp of the next event based on the graph information and past events information. The top row presents the RMSE of the real world datasets timestamps prediction, and the bottom row presents the RMSE of the synthetic datasets timestamps prediction.

which calculates the absolute difference between the actual values and the values of the model prediction.

5.7. Result and discussion

We compare our model with some state-of-the-art methods by evaluating the metrics of training time, the result is shown in Table 1. We trained five times to get the average training time and calculated the variance of training time. In the Synthetic dataset, our model gets better performance in training time than other state-of-the-art methods. In other datasets, the convergence of the RMTPP is the fastest. However, the performance of time prediction, Astrologer is better than RMTPP.

We recorded the negative log-likelihood of training with batch size 1024. Fig. 5 shows the negative log-likelihood. The above-mentioned models' predictive performance on the testing data of each dataset is shown in Fig. 6. We use 0.8 for training, and the rest 0.2 is divided into a test set and a validation set. Fig. 6 compares the predictive performance of predicting the event timestamps based on the past event sequences across five datasets. In the Stack overflow dataset, the model can reach the optimal value only by three steps. In the synthetic dataset, the model needs 40 steps to reach the optimal value. There are 0.7 million transaction records in the Financial dataset, and the max length of the sequence is 3325. It may cost fluctuate in loss function when the batch size is 1024 and the learning rate is 0.001. In general, in the real-world datasets and the self-correcting dataset, without graph structure, the Astrologer's performance is similar to the state-of-the-art methods. However, in the synthetic dataset with a graph structure, the Astrologer outperforms all models.

The bottom of Fig. 6 shows the time prediction performance of all the models mentioned in the five datasets. The Astrologer outperforms the other alternatives with lower RMSE for predicting timings in synthetic datasets. Meanwhile, the performance of our model is similar to the state-of-the-art method in real-world datasets without graph structure. This is because the synthetic dataset is simulated events by multi-dimensional Hawkes process, and we can get graph structure to the training model.

In the synthetic dataset, Astrologer has 2.732 RMSE, which is better than all models. In the ATMs dataset, the state-of-the-art method [13] has 2.228 RMSE , while Astrologer has achieves a 2.774 RMSE in prediction. Astrologer has 9.132 RMSE on the Stack Overflow datasets, which is similar to the state-of-the-art methods. Although we try to construct a graph structure by an unweighted k-nearest neighbor graph, the actual graph structure cannot be simulated like the real-world graph data. Therefore,

Table 1
The training time of each model (With 5 training).

Datasets	Metrics	Models				
		Astrologer	RMTPP	NHP	Intensity RNN	NLL LSTM
ATM	Training time	963.04	**278.12**	5444.14	955.20	1174.31
	(variance)	68.96	**8.12**	58583.78	570.08	129.95
Stack overflow	Training time	870.89	2100.46	1898.77	1027.23	**755.20**
	(variance)	**139.20**	1784.86	7428.34	232.58	301.80
Financial	Training time	803.40	**99.30**	1547.90	945.04	838.74
	(variance)	90.91	**5.69**	592.55	11250.53	77.60
Synthetic	Training time	**98.78**	104.73	1869.71	139.28	293.15
	(variance)	9.79	**2.06**	911.17	9.62	54.95
Self-correcting	Training time	198.29	**47.51**	1492.18	416.78	184.20
	(variance)	53.04	**0.14**	41.77	149.68	16.06

when we add the spatial graph information to the model, its performance may be similar to the state-of-the-art methods. However, we verify that adding graph information can improve the performance of timestamps prediction in graph-based datasets.

In summary, these results show that the Astrologer performs better than other state-of-the-art methods when applied on graph-based datasets. In other datasets that use K-nearest neighbor graphs to construct graph structures, the graph information may be the noise deviation of the proposed model, which will reduce the model's prediction performance.

6. Conclusion

It is well-known that the Hawkes process is widely used for event propagation forecasting. However, the Hawkes process' spatial information is always ignored in related datasets though it is essential. In this paper, we propose a novel non-parametric Hawkes process model, named Astrologer, based on the geometric deep learning. Astrologer inherits the advantages from both the graph convolutional neural network and the Hawkes process. Therefore, the model can predict markers and the timestamps of next events based on the information that is learned from the model. Compared with the baseline models and the state-of-the-art methods, we have two innovations: i) This is the first time that approximated the conditional intensity function by geometric deep learning and recalled inverse method to make a prediction. ii)The model used the graph neural network to learn the embedding from graph structure, and then the learned embedding will be integrated into the model. To prove the proposed model's effectiveness, we evaluate Astrologer on both real-world datasets and synthetic datasets. We predict the badge awarded of website users on the stack overflow dataset, the ATMs failures on the ATMs dataset, and the users' action(Buy or Sell) on the financial transaction dataset. We also evaluate Astrologer on two synthetic datasets. Then, we may get a better representation of the nodes and improve the model's performance.

CRediT authorship contribution statement

Haizhou Du: Conceptualization, Methodology, Data curation. **Yan Zhou:** Writing - original draft, Software, Validation. **Yunpu Ma:** Methodology, Writing - review & editing. **Shiwei Wang:** Visualization, Investigation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] G. Xiao, R. Wang, C. Zhang, A. Ni, Demand prediction for a public bike sharing program based on spatio-temporal graph convolutional networks, Multimedia Tools Appl. (2020) 1–19.
[2] Q. Wang, Y. Guo, L. Yu, P. Li, Earthquake prediction based on spatio-temporal data mining: an LSTM network approach, IEEE Trans. Emerg. Top. Comput. (2017).
[3] T.J. Liniger, Multivariate Hawkes Processes (Ph.D. thesis), ETH Zurich, 2009.
[4] H. Xu, W. Wu, S. Nemati, H. Zha, Patient flow prediction via discriminative learning of mutually-correcting processes, IEEE Trans. Knowl. Data Eng. 29 (1) (2016) 157–171.
[5] Ş. Ertekin, C. Rudin, T.H. McCormick, et al., Reactive point processes: A new approach to predicting power failures in underground electrical systems, Ann. Appl. Stat. 9 (1) (2015) 122–144.
[6] R. Lemonnier, K. Scaman, A. Kalogeratos, Multivariate Hawkes processes for large-scale inference, in: Thirty-First AAAI Conference on Artificial Intelligence, 2017.
[7] M. Eichler, R. Dahlhaus, J. Dueck, Graphical modeling for multivariate Hawkes processes with nonparametric link functions, J. Time Series Anal. 38 (2) (2017) 225–242.
[8] J. Shang, M. Sun, Geometric Hawkes processes with graph convolutional recurrent neural networks, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 2019, pp. 4878–4885.
[9] M. Defferrard, X. Bresson, P. Vandergheynst, Convolutional neural networks on graphs with fast localized spectral filtering, in: Advances in Neural Information Processing Systems, 2016, pp. 3844–3852.
[10] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, 2016, arXiv preprint arXiv:1609.02907.
[11] S. Xiao, J. Yan, X. Yang, H. Zha, S.M. Chu, Modeling the intensity function of point process via recurrent neural networks, in: AAAI, 2017, pp. 1597–1603.
[12] N. Du, H. Dai, R. Trivedi, U. Upadhyay, M. Gomez-Rodriguez, L. Song, Recurrent marked temporal point processes: Embedding event history to vector, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 1555–1564.
[13] H. Mei, J.M. Eisner, The neural Hawkes process: A neurally self-modulating multivariate point process, in: Advances in Neural Information Processing Systems, 2017, pp. 6754–6764.
[14] Y. Ogata, Statistical models for earthquake occurrences and residual analysis for point processes, J. Amer. Statist. Assoc. 83 (401) (1988) 9–27.
[15] S.-H. Yang, H. Zha, Mixture of mutually exciting processes for viral diffusion, in: International Conference on Machine Learning, 2013, pp. 1–9.
[16] L. Li, H. Deng, A. Dong, Y. Chang, H. Zha, Identifying and labeling search tasks via query-based hawkes processes, in: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2014, pp. 731–740.
[17] L. Li, H. Zha, Household structure analysis via hawkes processes for enhancing energy disaggregation, in: IJCAI, 2016, pp. 2553–2559.
[18] A. Stomakhin, M.B. Short, A.L. Bertozzi, Reconstruction of missing data in social networks based on temporal patterns of interactions, Inverse Problems 27 (11) (2011) 115013.
[19] L. Li, H. Zha, Dyadic event attribution in social networks with mixtures of hawkes processes, in: Proceedings of the 22nd ACM International Conference on Information & Knowledge Management, 2013, pp. 1667–1672.
[20] F. Salehi, W. Trouleau, M. Grossglauser, P. Thiran, Learning Hawkes processes from a handful of events, in: Advances in Neural Information Processing Systems, 2019, pp. 12694–12704.
[21] A.C. Türkmen, Y. Wang, A.J. Smola, Fastpoint: Scalable deep point processes, in: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, 2019, pp. 465–480.

[22] Y. Liu, T. Yan, H. Chen, Exploiting graph regularized multi-dimensional Hawkes processes for modeling events with spatio-temporal characteristics, in: IJCAI, 2018, pp. 2475–2482.
[23] W. Wu, H. Liu, X. Zhang, Y. Liu, H. Zha, Modeling event propagation via graph biased temporal point process, 2019, arXiv preprint arXiv:1908.01623.
[24] S. Xiao, M. Farajtabar, X. Ye, J. Yan, L. Song, H. Zha, Wasserstein learning of deep generative point process models, in: I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems, vol. 30, Curran Associates, Inc., 2017, pp. 3247–3257.
[25] J. Yan, X. Liu, L. Shi, C. Li, H. Zha, Improving maximum likelihood estimation of temporal point process via discriminative and adversarial learning, in: IJCAI, 2018, pp. 2948–2954.
[26] S. Li, S. Xiao, S. Zhu, N. Du, Y. Xie, L. Song, Learning temporal point processes via reinforcement learning, in: Advances in Neural Information Processing Systems, 2018, pp. 10781–10791.
[27] U. Upadhyay, A. De, M.G. Rodriguez, Deep reinforcement learning of marked temporal point processes, in: Advances in Neural Information Processing Systems, 2018, pp. 3168–3178.
[28] A.G. Hawkes, Spectra of some self-exciting and mutually exciting point processes, Biometrika 58 (1) (1971) 83–90.
[29] S.M. Kazemi, R. Goel, K. Jain, I. Kobyzev, A. Sethi, P. Forsyth, P. Poupart, Relational representation learning for dynamic (knowledge) graphs: A survey, 2019, arXiv preprint arXiv:1905.11485.
[30] O. Aalen, O. Borgan, H. Gjessing, Survival and Event History Analysis: A Process Point of View, Springer Science & Business Media, 2008.
[31] S. Mallat, A Wavelet Tour of Signal Processing, Elsevier, 1999.
[32] Y. Ogata, On Lewis' simulation method for point processes, IEEE Trans. Inform. Theory 27 (1) (1981) 23–31.
[33] S. Ross, Chapter 4 - generating discrete random variables, in: S. Ross (Ed.), Simulation, fifth ed., Academic Press, 2013, pp. 47–68, https://doi.org/10.1016/B978-0-12-415825-2.00004-8, URL http://www.sciencedirect.com/science/article/pii/B9780124158252000048.
[34] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2014, arXiv preprint arXiv:1412.6980.
[35] I. Sutskever, J. Martens, G. Dahl, G. Hinton, On the importance of initialization and momentum in deep learning, in: International Conference on Machine Learning, 2013, pp. 1139–1147.
[36] S. Grant, B. Betts, Encouraging user behaviour with achievements: an empirical study, in: 2013 10th Working Conference on Mining Software Repositories, MSR, IEEE, 2013, pp. 65–68.
[37] F. Monti, M. Bronstein, X. Bresson, Geometric matrix completion with recurrent multi-graph neural networks, in: Advances in Neural Information Processing Systems, 2017, pp. 3697–3707.