NEURAL POINT PROCESS FOR FORECASTING SPATIOTEMPORAL EVENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Forecasting events occurring in space and time is a fundamental problem. Existing neural point process models are only temporal and are limited in spatial inference. We propose a family of deep sequence models that integrate spatiotemporal point processes with deep neural networks. Our novel Neural Spatiotemporal Point Process model is flexible, efficient, and can accurately predict irregularly sampled events. The key construction of our approach is based on space-time separation of temporal intensity function and time-conditioned spatial density function, which is approximated by kernel density estimation. We validate our model on the synthetic spatiotemporal Hawkes process and self-correcting process. On many benchmark spatiotemporal event forecasting datasets, our model demonstrates superior performances. To the best of our knowledge, this is the first neural point process model that can jointly predict the continuous space and time of events.

1 INTRODUCTION

Earthquakes, taxi pickups, mobile check-ins, are all examples of discrete events occurring at different space and time. Accurate forecasting of spatiotemporal events are fundamentally important for disaster response (Veen & Schoenberg, 2008), logistic optimization (Safikhani et al., 2018) and social media analysis (Liang et al., 2019). In sharp contrast to other sequence data such as speech, texts, or sensor measurements, spatiotemporal events pose unique challenges to deep learning.

Spatiotemporal events occur irregularly with uneven time and space intervals. Interpolating an irregular sampled sequence into a regular sequence can introduce significant biases (Rehfeld et al., 2011). Furthermore, event sequences contain both short-range and long-range spatiotemporal dependencies. The rate of an event at a given spacetime point depends on the history of the events preceding it, as well as the events geographically correlated to it.

Discrete-time deep sequence models such as RNNs (Hochreiter & Schmidhuber, 1997; Chung et al., 2014) assume events to be evenly sampled. Other continuous-time sequence models such as NeuralODE (Chen et al., 2018) and Neural Hawkes process (Mei & Eisner, 2017; Zuo et al., 2020) can handle irregular time intervals, but fail to model spatial dependencies. In statistics, spatiotemporal point process (Daley & Vere-Jones, 2007; Reinhart et al., 2018) is a family of models that captures the stochastic process of events that occur in continuous space and time. However, these methods often require feature engineering and are computationally expensive.

In this paper, we model events that occur in continuous space-time. Our model, *Neural Spatiotemporal Point Process* (Neural-STPP), is built on a novel generalization of spatiotemporal point processes, enhanced with the representation power of deep neural networks. In particular, we encode the history of past events into hidden states of the neural networks. We decompose the intensity function to account for the influence from continuous time and space events separately. The event intensity at a given space-time point not only depends on the history of past events but also events in neighboring locations. We then approximate the time-conditional spatial density function with kernel density estimation.

In summary, our work makes the following key contributions:

- **Neural Spatiotemporal Point Process (Neural-STPP).** We propose Neural Spatiotemporal Point Process models for modeling unevenly sampled spatiotemporal events. It in-

tegrates deep neural networks into spatiotemporal point processes to learn hidden states based on past spatial and temporal information.

- **Deep Kernel Density Estimation for Spatial Modeling.** To make the inference tractable, we separate the spatiotemporal intensity into spatial conditional probability and temporal intensity. We use kernel density estimation (KDE) to approximate the spatial conditional probability.
- **Effectiveness.** We demonstrate the effectiveness of the proposed neural spatiotemporal point process model using many synthetic and real-world spatiotemporal event forecasting tasks where it is shown to achieve superior performance and more interpretable predictions.

2 Related Work

Spatiotemporal Forecasting Modeling the spatiotemporal dynamics of a system in order to forecast the future is a fundamental task in many fields. Most work on spatiotemporal forecasting has been focused on spatiotemporal data measured at regular space-time interval, e.g., (Xingjian et al., 2015; Li et al., 2018; Yao et al., 2019; Fang et al., 2019; Park et al., 2019; Geng et al., 2019). For discrete spatiotemporal events, statistical methods include space-time point process, see (Moller & Waagepetersen, 2003; Mohler et al., 2011). (Zhao et al., 2015) propose multi-task feature learning whereas (Yang et al., 2018a;b) propose RNN-based model to predict spatiotemporal check-in events. These discrete-time models assume data are sampled evenly, thus are unsuitable for modeling irregularly sampled events.

Continuous-time Dynamic Models Continuous-time sequence models such as Neural ODE (Chen et al., 2018) assumes the latent dynamics are continuous and can be obtained by an ODE solver. There are many variations (Jia & Benson, 2019; Dupont et al., 2019; Zhang et al., 2019; Liu et al., 2019; Gholami et al., 2019; Fernandes et al., 2019; Finlay et al., 2020; Han et al., 2020). GRU-ODE-Bayes proposed by De Brouwer et al. (2019) introduces a continuous-time version of GRU and a Bayesian update network capable of handling sporadic observations. However, Mozer et al. (2017) shows that there is no significant benefit of using continuous-time RNN for discrete event data. Special treatment is needed for modeling unevenly sampled events.

Neural Point Process Temporal point process is a classic topic in statistics (Moller & Waagepetersen, 2003; Daley & Vere-Jones, 2007; Reinhart et al., 2018). Neural point process couples deep neural networks with point process and has received considerable attention. For example, neural Hawkes process applies RNNs to approximate the intensity function (Du et al., 2016; Mei & Eisner, 2017; Xiao et al., 2017; Zhang et al., 2020), and (Zuo et al., 2020) employs Transformers. (Shang & Sun, 2019) integrates graph convolution structure. However, all existing works focus on temporal point processes without spatial modeling. For datasets with spatial information, they discretize the space and treat them as event "markers". Perhaps the closest to ours is the work by Okawa et al. (2019), which extends Du et al. (2016) for spatiotemporal event prediction but they only predict the density instead of the next location and time of the event.

3 Background

3.1 Temporal Point Processes and Limitation in Spatial Inference

A temporal point process (TPP) $N(t)$ models the number of events that occurred during the interval $(0, t]$. It is characterized by a nonnegative conditional intensity function $\lambda^*(t) := \lambda(t|\mathcal{H}_t)$, where $\mathcal{H}_t = \{t_1, ..., t_n < t\}$ denotes all events up to t. A marked temporal point process (MTPP) models the intensity to also depend on the past events' information other than timing. It describes an event as a two-tuple $\{t, y\}$, where $t \in \mathbb{R}^+$ is the event timing and y is the event's continuous or discrete feature, namely, the event marker. The history condition is then denoted as $\mathcal{H}_t = \{(t_1, y_1), (t_2, y_2), ..., (t_n, y_n) : t_n < t\}$.

While an MTPP utilizes the past event markers, its intensity is uni-variate and can not be used for predicting the next event marker. Most Neural-MTPP models, e.g. Mei & Eisner (2017); Zuo et al. (2020) only consider marker inference when the markers are discrete. Discrete markers allow one to translate MTPP as a multivariate TPP, and predict as a classification problem.

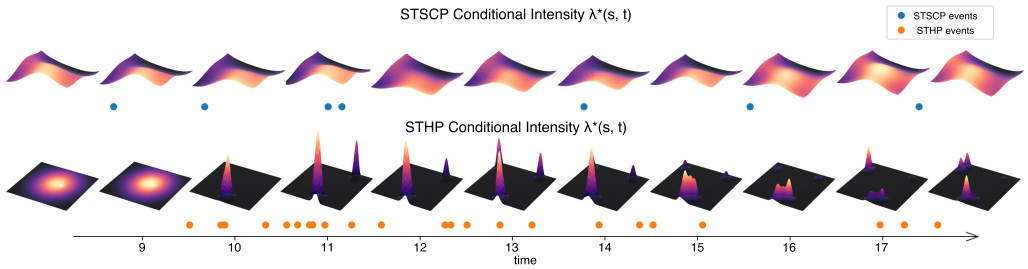


Figure 1: Intensity functions for the spatiotemporal Hawkes process (STHP) and the spatiotemporal self-correcting process (STSCP) $\lambda^*(s,t)$ over time. The dots denote event arrivals.

For spatial inference, the locations are continuous-valued. One option for a Neural-MTPP model to predict locations is to treat the locations as "continuous markers" and perform regression on the locations (Du et al., 2016). The resulting loss function has two terms: a negative log-likelihood (NLL) part for the time-intensity, and a regression loss part for the location. However, such a model cannot learn the space-time intensity and two terms in the loss function can compete with each other, resulting in poor space and time predictions. Consider an earthquake model, in which each event results in an increase in intensity that depends on both the epicenter location and event time. Temporal point process would not be able capture such dynamics.

3.2 Spatiotemporal Point Process

To overcome this limitation, we describe the spatiotemporal point process (STPP), a generalization of the temporal point process to spatiotemporal modeling. It captures the variation of intensity change at different spatial locations and is thus well-suited for spatial inference.

Definition A spatiotemporal point process (STPP) $N(t, \mathcal{S})$ models the number of events that occurred during the interval $(0, t]$ in the spatial domain $\mathcal{S} \subseteq \mathbb{R}^2$. The conditional intensity function $\lambda^*(\mathbf{s}, t) := \lambda(\mathbf{s}, t|\mathcal{H}_t)$ uniquely determines a spatiotemporal point process, where $\mathcal{H}_t := \{(\mathbf{s}_1, t_1), (\mathbf{s}_2, t_1), ..., (\mathbf{s}_n, t_n)\}$. $\lambda^*(\mathbf{s}, t)$ is such defined that the probability of finding an event in an infinitesimal time interval $[t, t+\delta t]$ and an infinitesimal spatial ball $B(\mathbf{s}, \delta\mathbf{s})$ is $\lambda^*(\mathbf{s}, t)|B(\mathbf{s}+\delta\mathbf{s})|\delta t$, where $|B(\mathbf{s}+\delta\mathbf{s})|$ is the Lebesgue measure of the ball.

Relationship to TPP When the locations of the events are disregarded, a STPP characterized by $\lambda^*(\mathbf{s}, t)$ is equivalent to a TPP characterized by $\lambda^*(t) = \int_{\mathcal{S}} \lambda^*(\mathbf{s}, t) d\mathbf{s}$. The conditional density of the new event location is denoted as $f^*(\mathbf{s}|t) := f^*(\mathbf{s}|t, \mathcal{H}_t)$, depending on the new event timing and the past events. As a result, $\lambda^*(\mathbf{s}, t)$ can be alternatively defined as $\lambda^*(\mathbf{s}, t) = f^*(\mathbf{s}|t)\lambda^*(t)$.

Spatiotemporal Hawkes Process One of the most well-known STPPs, a spatiotemporal Hawkes (or self-exciting) process (STHP) is characterized by the following intensity function: (Reinhart et al., 2018)

$$\lambda^*(\mathbf{s}, t) := \mu g_0(\mathbf{s}) + \sum_{i:t_i<t} g_1(t, t_i) g_2(\mathbf{s}, \mathbf{s}_i) : \mu > 0 \tag{1}$$

where $g_0(\mathbf{s})$ is the probability density of a distribution over $\mathcal{S}$, g_1 is the triggering kernel and is often implemented as the exponential decay function, $g_1(\Delta t) := \alpha \exp(-\beta \Delta t) : \alpha, \beta > 0$, and $g_2(\mathbf{s}, \mathbf{s}_i)$ is the density of an unimodal distribution over $\mathcal{S}$ centered at $\mathbf{s}_i$. When the locations of the events are disregarded, a STHP is equivalent to a markless Hawkes process. (see Appendix A.1.3). The STHP assumes every past event has an additive, positive, decaying, and spatially local influence over future events. Such a pattern resembles neuronal firing and earthquakes.

Spatiotemporal Self-Correcting Process Compared with self-exciting, self-inhibition characteristic is often overlooked in spatiotemporal point process models. Based on the univariate self-correcting process(Isham & Westcott, 1979), we design spatiotemporal self-correcting process (STSCP), which captures the local self-inhibitory effect of events. It is characterized by

$$\lambda^*(\mathbf{s}, t) = \mu \exp\left(g_0(\mathbf{s})\beta t - \sum_{i:t_i<t} \alpha g_2(\mathbf{s}, \mathbf{s}_i)\right) : \alpha, \beta, \mu > 0 \tag{2}$$

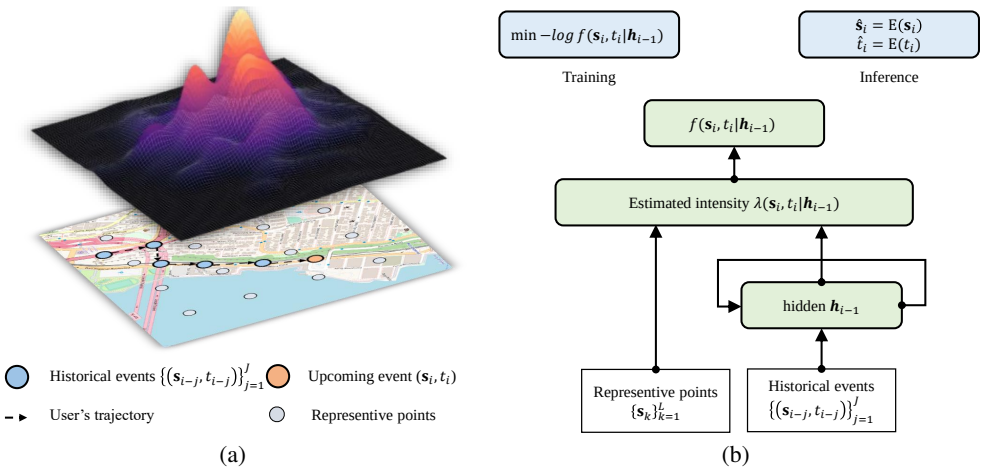


Figure 2: Illustration of the Neural-STPP. (a) We adopt kernel density estimation to approximate the spatial probability with the historical events and representative points. (b) For a given event sequence, we estimate the conditional intensity with a recurrent neural network.

where $g_0(\mathbf{s})$ is the density of a distribution over $\mathcal{S}$, and $g_2(\mathbf{s}, \mathbf{s}_i)$ is the density of an unimodal distribution over $\mathcal{S}$ centered at $\mathbf{s}_i$. Note that the STSCP without spatial markers is not equivalent to a markless self-correcting process. (see Appendix A.1.3)

The STSCP assumes that the background intensity increases steadily with a varying speed at different locations, and the arrival of each event reduces the intensity nearby. (see Figure 1) The next event is likely to be in a high-intensity region with no recent events. Therefore, the STSCP is capable of modeling some regular event sequences, such as an alternating home-to-work travel sequence.

4 Methodology

4.1 Neural Spatiotemporal Point Process (Neural-STPP) Model

In practice, it is difficult to decide which parametric form of the conditional intensity function can fit the complex spatiotemporal events best without sufficient prior knowledge. Therefore, we propose the Neural-STPP model which utilizes a deep neural network to model a general nonlinear dependency over space and time. Figure 2(b) visualizes the overall pipeline for Neural-STPP. At the high-level, it combines a kernel density estimation for spatial modeling and an RNN to model the nonlinear spatial dependency and the timestamps of past events.

Summarize Past Influence by a Neural Network We use a recurrent neural network to learn the dependence of the intensities over past events. As shown in Figure 2(b), past events $\{(\mathbf{s}_{i-j}, t_{i-j})\}_{j=1}^{J}$ are fed into a recurrent neural network with gated recurrent unit (Chung et al. (2014)) to extract meaningful spatotemperal information. $\boldsymbol{h}_{i-j} = \mathrm{GRU}(\boldsymbol{h}_{i-j-1}, (\mathbf{s}_{i-j}, t_{i-j}))$. The hidden state $\boldsymbol{h}_{i-1}$ summarizes the spatiotemperal influence of past events up to $i-1$.

Kernel Density Estimation for Spatial Modeling The space domain $\mathcal{S} \in \mathbb{R}^2$ is divided into $L = M \times N$ grids. The L discrete points $\{\mathbf{s}_k\}_{k=1}^{L}$ within the spatial region are called the space representative points. We further assume the spatial conditional density does not rely on time, $f(\mathbf{s}|t, \mathcal{H}_{t_{i-1}}) = f(\mathbf{s}|\mathcal{H}_{t_{i-1}})$. Given a spatiotemporal sequences $\{(\mathbf{s}_{i-j}, t_{i-j})\}_{j=1}^{J}$, the spatial conditional density function at t is approximated by the weighted kernel density estimation.

$$f(\mathbf{s}|t, \mathcal{H}_{t_{i-1}}) = f(\mathbf{s}|\mathcal{H}_{t_{i-1}}) = \underbrace{\sum_{j=1}^{J} w_j^s g_s(\mathbf{s}, \mathbf{s}_{i-j})}_{\text{past space influence}} + \underbrace{\sum_{k=1}^{L} w_k^s g_s(\mathbf{s}, \mathbf{s}_k)}_{\text{background space influence}} \tag{3}$$

where $g_s(\mathbf{x}, \mathbf{y})$ is a typical kernel function, and $\sum_{j=1}^{J} w_j^s + \sum_{k=1}^{L} w_k^s = 1$. In our experiments, we adopt a Gaussian kernel that $g_s(\mathbf{x}, \mathbf{y}) := \frac{1}{2\pi}|\Sigma|^{-\frac{1}{2}} \exp(-\frac{1}{2}(\mathbf{x}-\mathbf{y})\Sigma^{-1}(\mathbf{x}-\mathbf{y})^T)$. This spatial

probability density function considers two aspects of information: the influence of trajectory information (represented by past locations and $\{w_j^s\}_{j=1}^J$) and background information (represented by representative points and $\{w_k^s\}_{k=1}^L$) on the location of the next event. Given the learned representation $\boldsymbol{h}_{i-1}$, we model the spatial weight $w_:^s$ with a softmax function

$$w_:^s = \frac{\exp \mathbf{W}_: \boldsymbol{h}_{i-1}}{\sum_{k=1}^L \exp \mathbf{W}_k \boldsymbol{h}_{i-1} + \sum_{j=1}^J \exp \mathbf{W}_j \boldsymbol{h}_{i-1}} \tag{4}$$

$\mathbf{W}_:$ is the weight matrix corresponding to spatial weight $w_:^s$.

Conditional Intensity Based on $\boldsymbol{h}_{i-1}$, we formulate the conditional intensity function as

$$\lambda^*(\mathbf{s}, t) = f^*(\mathbf{s}|t)\lambda^*(t) = f^*(\mathbf{s}|t) \exp\Big(\underbrace{\boldsymbol{v}^{t^\top} \cdot \boldsymbol{h}_{i-1}}_{\substack{\text{past time}\\ \text{influence}}} + \underbrace{w^t(t - t_i)}_{\substack{\text{current time}\\ \text{influence}}} + \underbrace{b^t}_{\substack{\text{base time}\\ \text{intensity}}} \Big), \tag{5}$$

where $\boldsymbol{v}^t$ is a column vector, and w^t, b are scalars. Such a design ensures a closed-form conditional intensity and thus avoids the complexity issue in A.2.4. For simplicity purposes, we borrows (Du et al., 2016)'s design of $\lambda^*(t)$; This can be changed to any other Neural-TPP's $\lambda^*(t)$, as long as the conditional intensity is integrable over t, such that the likelihood does not need numerical integration. We also discuss many alternative designs in Appendix D which provide empirical and theoretical motivation for our current design. We derive the joint likelihood that the next event will occur at the location $\mathbf{s}$ and the time t given the history $\mathcal{H}_{t_{i-1}}$ as

$$\begin{aligned} f^*(\mathbf{s}, t) &= \lambda^*(\mathbf{s}, t) \exp\Big(- \int_{t_i}^t \lambda^*(\tau) d\tau \Big) \qquad \text{[see Appendix A.2.1]} \\ &= \lambda^*(\mathbf{s}, t) \exp\Big(-\frac{1}{w_t} (\exp\{\boldsymbol{v}^{t^\top} \cdot \boldsymbol{h}_{i-1} + w^t(t - t_i) + b^t\} - \exp\{\mathbf{v}^T h + b^t\}) \Big). \end{aligned} \tag{6}$$

The expected time and location of next event evaluate to

$$\mathbb{E}[t_i|\mathcal{H}_{t_{i-1}}] = \int_{t_i}^\infty t f^*(t) dt \quad \text{and} \quad \mathbb{E}[\boldsymbol{s}_i|\mathcal{H}_{t_{i-1}}] = \sum_{j=1}^J w_j^s \mathbf{s}_{i-j} + \sum_{k=1}^L w_k^s \mathbf{s}_k. \tag{7}$$

4.2 Parameter Learning

Given spatiotemporal event data $\mathbf{S} = \{(\mathbf{s}_{i-j}, t_{i-j})\}_{j=1}^J$, we perform parameter learning by maximizing the log-likelihood of observing the next event at a given location $(\mathbf{s}_i, t_i)$, namely $\log f^*(\mathbf{s}_i, t_i)$. When learning the parameters of the model, it tends to give more weight to its past space influence while ignoring its base space intensity. This results in $w_j^s \gg w_k^s, \forall j, k$. In order to prevent the model from being biased by historical information, we incorporate a regularization term to minimize the largest value of w^s. More formally, the optimization problem on a batch of N sequences $\{\mathbf{S}^{(n)}\}_{n=1}^N$ is formulated in Equation 10

$$\min -\frac{1}{N} \sum_n \log f^*(\mathbf{s}_i^{(n)}, t_i^{(n)}) + \gamma \max(w^s) \tag{8}$$

where γ defines the weight of the regularization term.

4.3 Variations

All-time Conditioned Spatial PDF Currently, $f^*(\mathbf{s}|t)$ is conditioned only on the past events. We can modify the conditional spatial PDF to further account for the current time influence by formulating w^s to be a function of the current time:

$$w_:^s(t) = \frac{\exp \mathbf{W}_:(\boldsymbol{h}_{i-1} \| t)}{\sum_{k=1}^L \exp \mathbf{W}_k(\boldsymbol{h}_{i-1} \| t) + \sum_{j=1}^J \exp \mathbf{W}_j(\boldsymbol{h}_{i-1} \| t)}, \tag{9}$$

where $\|$ denotes concatenation. Then, in the inference stage, the time prediction has to be evaluated first; the predicted t and the hidden states are then fed to a network to output w^s, which is finally

used for location prediction. However, although such a design increases the model's expressibility, it leads to the propagation of uncertainty, i.e., the prediction error in time amplifies the prediction error in location. Experiments have shown that the past-time conditioned spatial PDF has overall better performance than all-time conditioned spatial PDE on real-world datasets, see Section 5.3.

Regularization The regularization term prevents the model from being biased by historical information. The regularization term $\max(w^s)$ can be changed to the entropy regularization (Mnih et al. (2016)) of w^s, which has been widely applied in reinforcement learning to promote action diversity. The loss function then becomes

$$\min -\frac{1}{N}\sum_{n}\log f^*(\mathbf{s}_i^{(n)}, t_i^{(n)}) - \gamma\sum_{j} w_j^s \log(w_j^s) \tag{10}$$

We have experimented with both regularizations, see Section 5.3. We found that different regularizations yield similar performance over real-world datasets.

Marked Spatiotemporal Point Processes The NSTPP model can also make use of the past non-spatiotemporal event markers, e.g., earthquake magnitudes or event categories, by having the continuous markers and/or the embeddings of the discrete markers as a part of the RNN input. The model is also able to predict such markers by learning an additional linear layer that transforms the hidden states to the marker space. We leave this as a future work as this is not the main focus of this paper.

5 Experiments

We evaluate Neural-STPP for spatiotemporal prediction using both synthetic and real-world data.

Baselines We compare Neural-STPP with five state-of-the-art deep learning models, including

- GRU (Cho et al., 2014): a multi-layer GRU which learns from the sequence ofspatiotemporal events;
- NeuralODE (Chen et al., 2018): an GRU encoder with ODE-based decoder to model continous-time latent dynamics;
- NeuralODE++: NeuralODE with separate encoders for spatial and temporal information.
- RMTPP (Du et al., 2016): it uses GRU to model the temporal intensity function. We modify this model slightly to also predict the event location.
- Neural Hawkes Process (Mei & Eisner, 2017): a continuous-time LSTM to learn a time intensity function based on past events and predict the time of the next event.

All models are implemented in PyTorch. Training is performed using the Adam optimizer with a learning rate initialized to 4e-3 for Neural-STPP and RMTPP, and 1e-3 for other models. Models are trained until convergence up to a maximum of 100 iterations with a minibatch size of 64. The regularization weight γ is set to 1e-4. We also initialize $\Sigma = \mathbf{I}, w^t = 0.1, b = 0.1$ We use Neural Hawkes Process only for time prediction comparison. In practice, this model performs quite badly on real-world datasets compared to others, so we did not include the results in Figure 4(a). For more details, refer to Table 7 in Appendix C.2.

5.1 Synthetic Data

We simulated the STHP and the STSCP, each with three different parameter settings, labeled as DS1, DS2, and DS3 in the tables. (see Appendix B.2, B.3 for detailed simulation algorithms and Table 3 for experimental settings)

Experiment Result In order to better verify Neural-STPP's competence to model the spatiotemporal intensity, we compare it with the maximum likelihood estimator of STHP (ST-Hawkes PP), which is the only model that can learn the spatiotemporal intensity. RMTPP can only learn the temporal intensity. Refer to Appendix A.3.2 for a detailed description of ST-Hawkes PP.

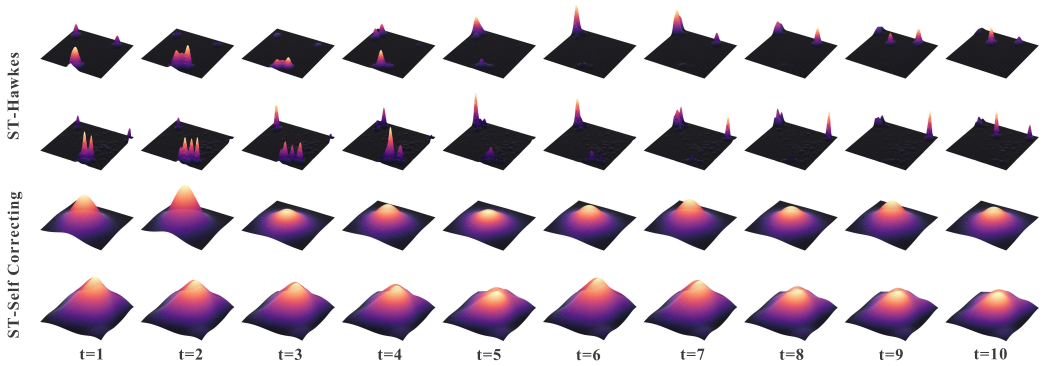


Figure 3: Ground truth and learned intensity on synthetic data. Top row demonstrates the ground-truth intensity, and the second row presents the learned conditional intensity by our Neural-STPP model.

	Spatiotemporal Hawkes process						Spatiotemporal Self Correcting process					
	DS1		DS2		DS3		DS1		DS2		DS3	
	NLL	HD	NLL	HD	NLL	HD	NLL	HD	NLL	HD	NLL	HD
Neural-STPP (ours)	3.47	0.27	5.69	0.41	**3.23**	0.16	**0.62**	**0.20**	**0.53**	**0.13**	**0.78**	**0.15**
ST-Hawkes PP	**3.32**	**0.09**	**2.29**	**0.06**	3.93	**0.05**	1.60	0.34	2.02	0.13	1.59	0.18

Table 2: Negative log likelihood (NLL) and Hellinger distance of distribution (HD) on synthetic data. Comparison between ours and the gold standand ST-Hawkes PP on two type of point processes. Each process contains three datasets with different parameters.

Table 1 shows the mean absolute percentage error (MAPE) between the models' predicted intensity and the ground truth intensity on time. Compared to RMTPP, Neural-STPP generally obtained the same or lower MAPE. It indicates that the extension to spatiotemporal modeling does not affect the temporal prediction.

Table 1: $\lambda^*(t)$ error of models on synthetic data

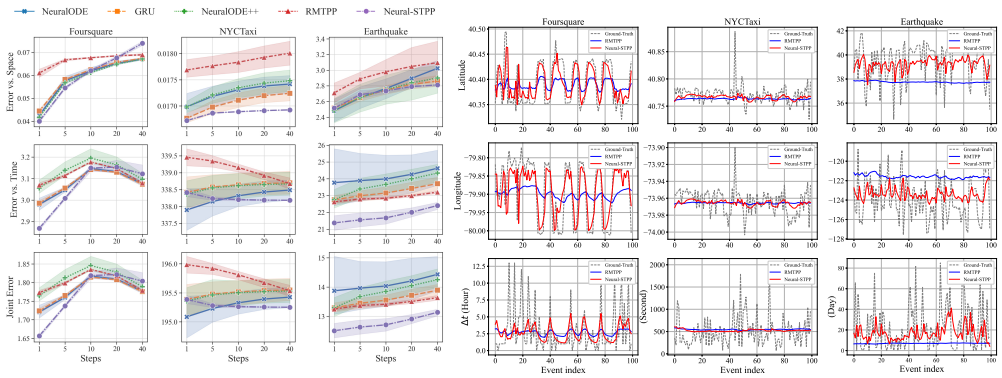
	ST-Hawkes			ST-Self Correcting		
	DS1	DS2	DS3	DS1	DS2	DS3
Neural-STPP	24.98	58.53	15.19	**22.20**	22.53	**19.08**
RMTPP	24.26	90.01	14.02	24.67	20.67	19.20
ST-Hawkes PP	2.98	11.30	4.38	27.38	18.20	20.01

Table 2 measures the qualities of predicted distributions. We report Negative Log Likelihood (NLL) of $f(\mathbf{s}, t|\mathcal{H}_{t_{i-1}})$ and the Hellinger Distance (HD) between the predicted distributions and the ground truth distribution averaged over time. On the ST-Hawkes dataset, since ST-Hawkes PP is the optimal model that has the correct parametric form, it outperforms our Neural-STPP in terms of $\lambda^*(t)$ error, NLL, and HD. In contrast, on ST-Self Correcting dataset, our Neural-STPP performs better than ST-Hawkes PP with much smaller $\lambda^*(t)$ error, NLL, and HD. It shows that Neural-STPP predicts the spatiotemporal intensity more accurately for unknown intensity functions. Figure 3 visualizes the predicted intensity of Neural-STPP and the ground truth, providing strong evidence that Neural-STPP can correctly learn the pattern and dynamics. In general, it proves that Neural-STPP is highly effective in spatiotemporal intensity prediction.

5.2 Real-World Data

Datasets We use three real-world datasets to evaluate the performance for spatiotemporal event forecasting. (1) **Foursquare Checkin dataset** which contains over 1 million records of users checking in different venues. Each event is a checkin record. We predict the location and time for the next checkin event of the user that has the most checkin records. (2) **NYC Taxi**[1] which contains about 173 million trip records of individual Taxi for consecutive 12 months in 2013. Each event is a pickup record for a taxi. We predict the location and time of the next pickup event of the taxi which has the most trip records. (3) **Earthquake**[2] which contains over 3 million earthquake records during the

[1] http://www.andresmh.com/nyctaxitrips
[2] https://www.kaggle.com/danielpe/earthquakes



(a) (b)

Figure 4: Performance evaluation of different models on three real-world datasets: Foursquare, NYC Taxi and Earthquake. (a) Time, space and joint prediction RMSEs for multi-step forecasting. (b) Time, space and joint prediction visualization.

years 1970-2018. We predict the location and time of the next earthquake above level 4.0 that occur in California state.

Experimental Setup In each experiment, the input is the J precedent events, and the targets are the subsequent K events. For all the datasets, we fix J to be 20. For evaluation, we choose $K \in \{1, 5, 10, 20, 40\}$. We measure the root mean square error (RMSE) of latitude and longitude (averaged across K predicted events) as the space error. Similarly, we measure the RMSE of delta time between 2 consecutive events as the time error and the uniform weight average of all outputs' RMSE as the joint error. Each dataset is split into train/val/test set with the ratio of 8:1:1. We report the average and standard error of each metric across 5 random runs with the same train-test set.

Prediction Performance. Figure 4(a) shows the performance of multi-step prediction for our Neural-STPP and baselines. For **Foursquare**, our model achieves the lowest time, space, and joint error for the initial 10 steps prediction. All models' time prediction errors decrease after the 10th step. This is because, after multiple steps, all models start generating the same time prediction. The predictions are not that reliable so the trend for time error is not very obvious. On **NYC Taxi**, our model reaches the lowest space prediction error for all different steps on the NYCTaxi dataset. Also, it achieves the smallest time error after 10 forecasting steps. However, the time error decreases starting from the 1st step for both Neural-STPP and RMTPP. Our explanation for the similar phenomenon in the Foursquare dataset also works in this case. The only difference is that our Neural-STPP and RMTPP make relatively bad predictions at first which leads to large RMSE. For long-term forecasting, the predicted time value would approach a constant value gradually, which contributes to a smaller time error. For **Earthquake**, our model has a significant advantage in time prediction over other baselines. Additionally, it attains a relatively lower time prediction error across all different step predictions.

We also visualize the predictions for our model compared with RMTPP in Figure 4(b). On **Foursquare**, it is evident that our model makes realistic spatiotemporal predictions while RMTPP fails to capture the trend of human mobility. On the other hand, Neural-STPP generates more dynamical prediction on the **NYCTaxi** and **Earthquake** compared with RMTPP. We can conclude that on datasets with highly irregular moving patterns, Neural-STPP is better at modeling the underlying event dynamics. In summary, Neural-STPP is a highly competitive model for spatiotemporal forecasting.

5.3 Ablation Study

In Section 4.3, we have discussed two variations of NSTPP: NSTPP with an all-time(current time and past event timings) conditioned spatial pdf and with an entropy regularization term.

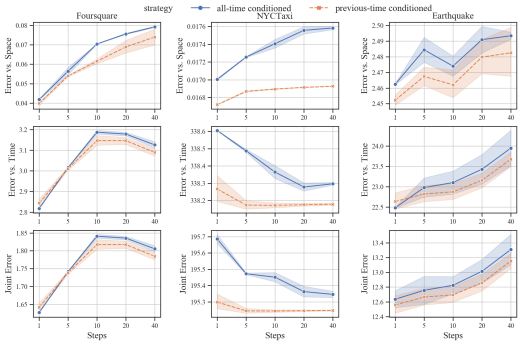
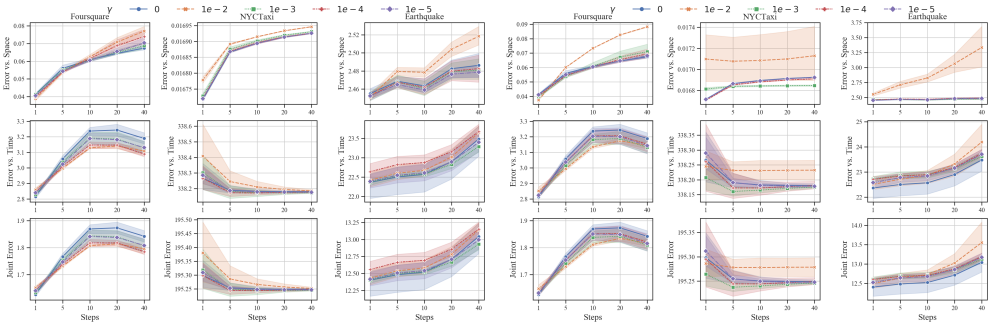


Figure 5: Ablation study of all-time or previous-time conditioned spatial pdf on real-world datasets



(a) Maximum regularization (b) Entropy regularization

Figure 6: Ablation study of regularization term on real-world datasets

All-time Conditioned Spatial PDF We have compared the performance of NSTPP with all-time conditioned spatial pdf and previous-time conditioned spatial pdf over the three real-world datasets, see Figure 5. The hyperparamater settings and initial parameter values are the same as in Appendix Table 4 and 5. In most of the cases, the latter (current design) has better performance.

Entropy Regularization We compare the model performance with different regularization methods. At the same time, we validated our choice of the regularization coefficient γ, see Figure 6. It can be seen that the choice of γ has a non-trivial effects on the prediction accuracy, and γ =1e-4 yields the best performance for maximum regularization. While the two regularization methods have similar best performance, the entropy regularization term is more sensitive to γ. In Figure 6(b), when γ =1e-2, the regularization term becomes dominant and the prediction becomes relatively inaccurate.

6 Conclusion

We propose a family of deep sequence models to model irregularly sampled spatiotemporal event data. Our model, Neural Spatiotemporal Point Process (Neural-STPP), integrates principled spatiotemporal points process with flexible deep neural networks. We derive tractable inference procedure by decomposing the tri-variate spatiotemporal intensity function into temporal intensity and time-conditioned spatial intensity. We approximate the temporal intensity function with an RNN and estimate the spatial intensity with KDE. Using synthetic data from spatiotemporal Hawkes process and self-correcting process, we show that our model can learn the correct spatial and temporal dynamics. We demonstrate superior forecasting performances on many real-world benchmark spatiotemporal event datasets.

References

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in neural information processing systems*, pp. 6571–6583,

2018.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014.

Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014.

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

Edward De Brouwer, Jaak Simm, Adam Arany, and Yves Moreau. Gru-ode-bayes: Continuous modeling of sporadically-observed time series. In *Advances in Neural Information Processing Systems*, pp. 7379–7390, 2019.

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 *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1555–1564, 2016.

Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. Augmented neural odes. In *Advances in Neural Information Processing Systems*, pp. 3140–3150, 2019.

Shen Fang, Qi Zhang, Gaofeng Meng, Shiming Xiang, and Chunhong Pan. Gstnet: Global spatial-temporal network for traffic flow prediction. In *IJCAI*, pp. 2286–2293, 2019.

Steven Fernandes, Sunny Raj, Eddy Ortiz, Iustina Vintila, Margaret Salter, Gordana Urosevic, and Sumit Jha. Predicting heart rate variations of deepfake videos using neural ode. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 0–0, 2019.

Chris Finlay, Jörn-Henrik Jacobsen, Levon Nurbekyan, and Adam M Oberman. How to train your neural ode. *arXiv preprint arXiv:2002.02798*, 2020.

Xu Geng, Yaguang Li, Leye Wang, Lingyu Zhang, Qiang Yang, Jieping Ye, and Yan Liu. Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 3656–3663, 2019.

Amir Gholami, Kurt Keutzer, and George Biros. Anode: Unconditionally accurate memory-efficient gradients for neural odes. *arXiv preprint arXiv:1902.10298*, 2019.

Zhen Han, Yuyi Wang, Yunpu Ma, Stephan Guünnemann, and Volker Tresp. The graph hawkes network for reasoning on temporal knowledge graphs. *arXiv preprint arXiv:2003.13432*, 2020.

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

Valerie Isham and Mark Westcott. A self-correcting point process. *Stochastic processes and their applications*, 8(3):335–347, 1979.

Junteng Jia and Austin R Benson. Neural jump stochastic differential equations. In *Advances in Neural Information Processing Systems*, pp. 9847–9858, 2019.

Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations (ICLR)*, 2018.

Wenwei Liang, Wei Zhang, and Xiaoling Wang. Deep sequential multi-task modeling for next check-in time and location prediction. In *International Conference on Database Systems for Advanced Applications*, pp. 353–357. Springer, 2019.

Xuanqing Liu, Tesi Xiao, Si Si, Qin Cao, Sanjiv Kumar, and Cho-Jui Hsieh. Neural sde: Stabilizing neural ode networks with stochastic noise. *arXiv preprint arXiv:1906.02355*, 2019.

Hongyuan Mei and Jason Eisner. The neural hawkes process: A neurally self-modulating multivariate point process. In *NeurIPS*, 2017.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937, 2016.

George O Mohler, Martin B Short, P Jeffrey Brantingham, Frederic Paik Schoenberg, and George E Tita. Self-exciting point process modeling of crime. *Journal of the American Statistical Association*, 106(493):100–108, 2011.

Jesper Moller and Rasmus Plenge Waagepetersen. *Statistical inference and simulation for spatial point processes*. CRC Press, 2003.

Michael C Mozer, Denis Kazakov, and Robert V Lindsey. Discrete event, continuous time rnns. *arXiv:1710.04110*, 2017.

Maya Okawa, Tomoharu Iwata, Takeshi Kurashima, Yusuke Tanaka, Hiroyuki Toda, and Naonori Ueda. Deep mixture point processes: Spatio-temporal event prediction with rich contextual information. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 373–383, 2019.

Cheonbok Park, Chunggi Lee, Hyojin Bahng, Kihwan Kim, Seungmin Jin, Sungahn Ko, Jaegul Choo, et al. Stgrat: A spatio-temporal graph attention network for traffic forecasting. *arXiv:1911.13181*, 2019.

Kira Rehfeld, Norbert Marwan, Jobst Heitzig, and Jürgen Kurths. Comparison of correlation analysis techniques for irregularly sampled time series. *Nonlinear Processes in Geophysics*, 18(3): 389–404, 2011.

Alex Reinhart et al. A review of self-exciting spatio-temporal point processes and their applications. *Statistical Science*, 33(3):299–318, 2018.

Abolfazl Safikhani, Camille Kamga, Sandeep Mudigonda, Sabiheh Sadat Faghih, and Bahman Moghimi. Spatio-temporal modeling of yellow taxi demands in new york city using generalized star models. *International Journal of Forecasting*, 2018.

Jin Shang and Mingxuan Sun. Geometric hawkes processes with graph convolutional recurrent neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 4878–4885, 2019.

Alejandro Veen and Frederic P Schoenberg. Estimation of space–time branching process models in seismology using an em–type algorithm. *Journal of the American Statistical Association*, 103 (482):614–624, 2008.

Shuai Xiao, Mehrdad Farajtabar, Xiaojing Ye, Junchi Yan, Le Song, and Hongyuan Zha. Wasserstein learning of deep generative point process models. In *Advances in Neural Information Processing Systems*, pp. 3247–3257, 2017.

SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pp. 802–810, 2015.

Guolei Yang, Ying Cai, and Chandan K. Reddy. Recurrent spatio-temporal point process for check-in time prediction. In *CIKM*, pp. 2203–2211. ACM, 2018a.

Guolei Yang, Ying Cai, and Chandan K Reddy. Spatio-temporal check-in time prediction with recurrent neural network based survival analysis. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, 2018b.

Huaxiu Yao, Xianfeng Tang, Hua Wei, Guanjie Zheng, and Zhenhui Li. Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 5668–5675, 2019.

Qiang Zhang, Aldo Lipani, Omer Kirnap, and Emine Yilmaz. Self-attentive hawkes processes. In *International Conference on Machine Learning (ICML)*, 2020.

Tianjun Zhang, Zhewei Yao, Amir Gholami, Kurt Keutzer, Joseph Gonzalez, George Biros, and Michael Mahoney. Anodev2: A coupled neural ode evolution framework. *arXiv preprint arXiv:1906.04596*, 2019.

Liang Zhao, Qian Sun, Jieping Ye, Feng Chen, Chang-Tien Lu, and Naren Ramakrishnan. Multi-task learning for spatio-temporal event forecasting. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1503–1512, 2015.

Simiao Zuo, Haoming Jiang, Zichong Li, Tuo Zhao, and Hongyuan Zha. Transformer hawkes process. *International Conference on Machine Learning (ICML)*, 2020.

A MODEL DETAILS

A.1 TEMPORAL POINT PROCESS

In this appendix, we elaborate on the markless TPP model.

A.1.1 DEFINITION

An infinitesimal definition of a TPP's conditional intensity function $\lambda^*(t)$ is

$$\lambda^*(t)\delta t = \mathbb{E}[dN(t)|\mathcal{H}_{t-}], \delta t \to 0.$$

In other words, the probability of finding an event in the time interval $[t, t+\delta t)$ of infinitesimal width δt is $\lambda^*(t)\delta t$. Given that $\mathcal{H}_t = \{t_1, ..., t_n\}$, let $f^*(t) := f(t|\mathcal{H}_t)$ denotes the conditional probability density function that the next event will be at $t : t > t_n$, and $F^*(t) := F(t|\mathcal{H}_t) := \int_{t_n}^{t} f^*(t)$ to be the cumulative density function. The relationship $f^*(t)\delta t = \dfrac{\lambda^*(t)\delta t}{1 - F^*(t)}$ is straightforward since that the next event is at t implies that no event happens between the final event and t. Solving the equation yields

$$\begin{aligned}
& \lambda^*(t)dt = \frac{dF^*(t)}{dt}\frac{dt}{1 - F^*(t)} = d(-\log(1 - F^*(t))) \\
\to \quad & \int_{t_n}^{t} \lambda^*(\tau)d\tau = -\log(1 - F^*(\tau))|_{t_n}^{t} = -\log(1 - F^*(t)) + 0 \\
\to \quad & \exp\left(-\int_{t_n}^{t} \lambda^*(\tau)d\tau\right) = 1 - F^*(t) \\
\to \quad & f^*(t) = \lambda^*(t)(1 - F^*(t)) = \lambda^*(t)\exp\left(-\int_{t_n}^{t} \lambda^*(\tau)d\tau\right), \qquad (11)
\end{aligned}$$

which is an alternative definition of $f^*(t)$.

A.1.2 LIKELIHOOD

Given a TPP, the log likelihood of observing a sequence $\{t_1, t_2, ...t_n\}$ is

$$\begin{aligned}
\mathcal{L} &= \log\left[(1 - F^*(t))\prod_{i=1}^{n} f(t_i|\mathcal{H}_{t_i-})\right] \\
&= \sum_{i=1}^{n}\left[\log\lambda^*(t_i) - \int_{t_{i-1}}^{t_i} \lambda^*(\tau)d\tau\right] + \log(1 - F^*(t)) \\
&= \sum_{i=1}^{n}\log\lambda^*(t_i) - \int_{0}^{t_n} \lambda^*(\tau)d\tau - \int_{t_n}^{T} \lambda^*(\tau)d\tau \\
&= \sum_{i=1}^{n}\log\lambda^*(t_i) - \int_{0}^{T} \lambda^*(\tau)d\tau
\end{aligned}$$

where T refers to the end time of the observation, such that we know there is no event in $[t_n, T]$. In practice, T is often simply considered as t_n. Notice that the first term is the probability of events occurring at $t_1, t_2, ..., t_n$, while the second term is the probability of no event occurring elsewhere.

A.1.3 EXAMPLES

Poisson Process The stationary Poisson process is the most basic temporal point process, with its intensity defined as a constant. A nonhomogenous Poisson process is defined by a time-varying but not history-dependent intensity. Both are memoryless, since the $\mathcal{H}_s : s < t$ has no influence on $N(t) - N(s)$.

Hawkes Process The Hawkes process is characterized by the intensity $\lambda^*(t) = \mu + \sum_{i:t_i<t} g_1(t - t_i) : \mu > 0$, where g_1 is the triggering kernel and is often implemented as the exponential decay function, $g_1(\Delta t) := \alpha \exp(-\beta \Delta t) : \alpha, \beta > 0$. The Hawkes process captures self-excitation. It could be seen as the summation of a background process with constant intensity μ and the offspring process triggered by each historical event i, starting at t_i with the decreasing intensity $g_1(t - t_i)$. Therefore, each event could be thought of as either being a background event or having a parent event. This hierarchical clustering structure makes the Hawkes Process a useful tool for modeling earthquakes and aftershocks, or gang-related crimes and retaliations.

The Hawkes process is the markless version of the STHP, as

$$\int_{\mathcal{S}} \lambda^*(\mathbf{s}, t) d\mathbf{s} = \mu \int_{\mathcal{S}} g_0(\mathbf{s}) d\mathbf{s} + \sum_{i:t_i<t} g_1(t, t_i) \int_{\mathcal{S}} g_2(\mathbf{s}, \mathbf{s}_i) d\mathbf{s} = \mu + \sum_{i:t_i<t} g_1(t, t_i)$$

Self-Correcting Process The self-correcting process is characterized by the intensity $\lambda^*(t) = \exp(\mu t - \sum_{i:t_i<t} \alpha) : \alpha, \mu > 0$. It captures self-inhibition. While the background intensity is increasing over time, the arrival of a new event leads to a decrement of the intensity by a factor of $\exp(\alpha)$. The past events' influence is not additive but multiplicative, hence no clustering structure.

The event sequences generated by a self-correcting process feature regular inter-event intervals. Each interval is unlikely to be too short, as the intensity after an event is low; it is also unlikely to be too long, as the intensity grows exponentially. Real-world applications of the self-correcting process include modeling machine failure. After each failure, a repair will reduce the machine's failure rate, which increases as time goes.

Renewal Process The renewal process is a family of the temporal point process whose conditional intensity function only depends on the time since the most recent event, i.e., $\lambda^*(t) = g(t - t_n)$. The renewal process captures self-reset; the arrival of a new event resets the intensity to $g(0)$.

A.2 Spatiotemporal Point Process

In this section, we elaborate on the STPP model.

A.2.1 Conditional Density

$$\begin{aligned} f(\mathbf{s}, t|\mathcal{H}_t) &= f(t|\mathcal{H}_t) f(\mathbf{s}|t, \mathcal{H}_t) \\ &= \lambda^*(t) \exp\left(-\int_{t_n}^{t} \lambda^*(\tau) d\tau\right) f(\mathbf{s}|t, \mathcal{H}_t) \qquad \text{[see Appendix A.1.1]} \\ &= \lambda^*(\mathbf{s}, t) \exp\left(-\int_{t_n}^{t} \lambda^*(\tau) d\tau\right) \end{aligned}$$

A.2.2 Likelihood

Given a STPP, the log-likelihood of observing a sequence $\{(\mathbf{s}_1, t_1), (\mathbf{s}_2, t_2), ...(\mathbf{s}_n, t_n)\}$ is

$$\begin{aligned} \mathcal{L} &= \log\left[(1 - F^*(t)) \prod_{i=1}^{n} f(\mathbf{s}_i, t_i|\mathcal{H}_{t_i-})\right] \\ &= \sum_{i=1}^{n} \left[\log \lambda^*(\mathbf{s}_i, t_i) - \int_{t_{i-1}}^{t_i} \lambda^*(\tau) d\tau\right] + \log(1 - F^*(t)) \\ &= \sum_{i=1}^{n} \log \lambda^*(\mathbf{s}_i, t_i) - \int_{0}^{t_n} \lambda^*(\tau) d\tau - \int_{t_n}^{T} \lambda^*(\tau) d\tau \\ &= \sum_{i=1}^{n} \log \lambda^*(\mathbf{s}_i, t_i) - \int_{0}^{T} \lambda^*(\tau) d\tau \end{aligned}$$

A.2.3 INFERENCE

With a trained STPP and a sequence of history events, we can predict the next event timing and location using their expectations, which evaluate to

$$\begin{aligned}\mathbb{E}[t_i|\mathcal{H}_{t_{i-1}}] &= \int_{t_{i-1}}^{\infty} t \int_{\mathcal{S}} f(\mathbf{s}, t|\mathcal{H}_{t_{i-1}}) d\mathbf{s}dt = \int_{t_{i-1}}^{\infty} t \exp\left(-\int_{t_{i-1}}^{t} \lambda^*(\tau)d\tau\right) \int_{\mathcal{S}} \lambda^*(\mathbf{s}, t) d\mathbf{s}dt, \\ &= \int_{t_{i-1}}^{\infty} t \exp\left(-\int_{t_{i-1}}^{t} \lambda^*(\tau)d\tau\right) \lambda^*(t) d\mathbf{s}dt \text{ and}\end{aligned}$$

$$\begin{aligned}\mathbb{E}[\mathbf{s}_i|\mathcal{H}_{t_{i-1}}] &= \int_{t_{i-1}}^{\infty} \int_{\mathcal{S}} \mathbf{s}\lambda^*(\mathbf{s}, t) \exp\left(-\int_{t_{i-1}}^{t} \lambda^*(\tau)d\tau\right) d\mathbf{s}dt \\ &= \int_{t_{i-1}}^{\infty} \exp\left(-\int_{t_{i-1}}^{t} \lambda^*(\tau)d\tau\right) \int_{\mathcal{S}} \mathbf{s}\lambda^*(\mathbf{s}, t) d\mathbf{s}dt, \text{ respectively.}\end{aligned}$$

A.2.4 COMPLEXITY ISSUE

It is worth noting that both learning and inference require conditional intensity. If the conditional intensity has no analytic expression, then it has to be evaluated by numerical integration over $\mathcal{S}$. Then, evaluating the likelihood or either expectation requires at least triple integral. Note that $\mathbb{E}[t_i|\mathcal{H}_{t_{i-1}}]$ and $\mathbb{E}[\mathbf{s}_i|\mathcal{H}_{t_{i-1}}]$ actually are sextuple integrals, but we can memorize all $\lambda^*(\mathbf{s}, t)$ from $t = t_{i-1}$ to $t \gg t_{i-1}$ to avoid re-compute the intensities. However, memorization leads to high space complexity. As a result, we generally want to avoid an intractable conditional intensity in the model.

A.3 SPATIOTEMPORAL HAWKES PROCESS

In this section, we present the details of the spatiotemporal Hawkes process MLE learner.

A.3.1 LIKELIHOOD

Specifically for the STHP, the second term in the STPP likelihood evaluates to

$$\begin{aligned}\int_0^T \lambda^*(\tau)d\tau &= \mu T + \alpha \int_0^T \int_0^\tau e^{-\beta(\tau-u)} dN(u)d\tau \\ &\quad (0 \le u \le \tau, 0 \le \tau \le T) \to (u \le \tau \le T, 0 \le u \le T) \\ &= \mu T + \alpha \int_0^T \int_u^T e^{-\beta(\tau-u)} d\tau dN(u) \\ &= \mu T - \frac{\alpha}{\beta} \int_0^T \left[e^{-\beta(T-u)} - 1\right] dN(u) \\ &= \mu T - \frac{\alpha}{\beta} \sum_{i=0}^{N} \left[e^{-\beta(T-t_i)} - 1\right]\end{aligned}$$

Finally, the STHP log-likelihood is

$$\mathcal{L} = \sum_{i=1}^{n} \log \lambda^*(\mathbf{s}_i, t_i) - \mu T + \frac{\alpha}{\beta} \sum_{i=0}^{N} \left[e^{-\beta(T-t_i)} - 1\right]$$

A.3.2 MLE SETTINGS

We pre-specified the model kernels $g_0(\mathbf{s})$ and $g_2(\mathbf{s}, \mathbf{s}_j)$ to be Gaussian:

$$g_0(\mathbf{s}) := \frac{1}{2\pi}|\Sigma_{g0}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{s}-\mathbf{s}_\mu)\Sigma_{g0}^{-1}(\mathbf{s}-\mathbf{s}_\mu)^T\right)$$
$$g_2(\mathbf{s}, \mathbf{s}_j) := \frac{1}{2\pi}|\Sigma_{g2}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{s}-\mathbf{s}_j)\Sigma_{g2}^{-1}(\mathbf{s}-\mathbf{s}_j)^T\right)$$

Therefore, the model has 11 scalar parameters: 2 for $\mathbf{s}_\mu$, 3 for Σ_{g0}, 3 for Σ_{g2}, α, β, and μ. We directly estimate $\mathbf{s}_\mu$ as the mean of $\{s_i\}_0^n$, and then estimate the other 9 parameters by minimizing the negative log-likelihood using the BFGS algorithm. T in the likelihood function is treated as t_n.

A.3.3 INFERENCE

Based on the general inference formulas in Appendix A.2.3, and also note that for an STHP,

$$\begin{aligned}
\int_{t_{i-1}}^{t} \lambda^*(\tau)d\tau &= \int_0^t \lambda^*(\tau)d\tau - \int_0^{t_{i-1}} \lambda^*(\tau)d\tau \\
&= \left\{\mu t - \frac{\alpha}{\beta}\sum_{j=0}^{i-1}\left[e^{-\beta(t-t_j)} - 1\right]\right\} - \left\{\mu t_{i-1} - \frac{\alpha}{\beta}\sum_{j=0}^{i-1}\left[e^{-\beta(t_{i-1}-t_j)} - 1\right]\right\} \\
&= \mu(t-t_{i-1}) - \frac{\alpha}{\beta}\sum_{j=0}^{i-1}\left[e^{-\beta(t-t_{i-1}+t_{i-1}-t_j)} - e^{-\beta(t_{i-1}-t_j)}\right] \\
&= \mu(t-t_{i-1}) - \frac{\alpha}{\beta}\left(e^{-\beta(t-t_{i-1})} - 1\right)\sum_{j=0}^{i-1}\left[e^{-\beta(t_{i-1}-t_j)}\right] \text{ and}
\end{aligned}$$

$$\int_{\mathcal{S}} \mathbf{s}\mu g_2(\mathbf{s}, \mathbf{s}_\mu)d\mathbf{s} = \mu\mathbf{s}_\mu$$
$$\int_{\mathcal{S}} \mathbf{s}\sum_{i=0}^{n} g_1(t,t_i)g_2(\mathbf{s},\mathbf{s}_i)d\mathbf{s} = \sum_{i=0}^{n} g_1(t,t_i)\int_{\mathcal{S}} \mathbf{s}g_2(\mathbf{s},\mathbf{s}_i)d\mathbf{s} = \sum_{i=0}^{n} g_1(t,t_i)\mathbf{s}_i$$
$$\int_{\mathcal{S}} \mathbf{s}\lambda^*(\mathbf{s},t)d\mathbf{s} = \mu\mathbf{s}_\mu + \sum_{i=0}^{n} g_1(t,t_i)\mathbf{s}_i,$$

we have

$$\begin{aligned}
\mathbb{E}[t_i|\mathcal{H}_{t_{i-1}}] = \int_{t_{i-1}}^{\infty} t &\left(\mu + \alpha\sum_{j=0}^{i-1} e^{-\beta(t-t_j)}\right) \\
&\exp\left(\frac{\alpha}{\beta}\left(e^{-\beta(t-t_{i-1})} - 1\right)\sum_{j=0}^{i-1}\left[e^{-\beta(t_{i-1}-t_j)}\right] - \mu(t-t_{i-1})\right) dt \text{ and}
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}[\mathbf{s}_i|\mathcal{H}_{t_{i-1}}] = \int_{t_{i-1}}^{\infty} &\left(\mu\mathbf{s}_\mu + \alpha\sum_{j=0}^{i-1} e^{-\beta(t-t_j)}\mathbf{s}_j\right) \\
&\exp\left(\frac{\alpha}{\beta}\left(e^{-\beta(t-t_{i-1})} - 1\right)\sum_{j=0}^{i-1}\left[e^{-\beta(t_{i-1}-t_j)}\right] - \mu(t-t_{i-1})\right) dt
\end{aligned}$$

Both require 1D numerical integration.

B SIMULATION DETAILS

In this appendix, we discuss a general algorithm for simulating any STPP, and a specialized algorithm for simulating an STHP. Both are based on an algorithm for simulating any TPP.

B.1 TPP SIMULATION

The most widely used technique to simulate a temporal point process is Ogata's modified thinning algorithm, as shown in Algorithm 1. (Daley & Vere-Jones (2007)) It is a rejection technique; it samples points from a stationary Poisson process whose intensity is always higher than the ground truth intensity, and then randomly discards some samples to get back to the ground truth intensity.

The algorithm requires picking the forms of $M^*(t)$ and $L^*(t)$ such that

$$\sup(\lambda^*(t+\Delta t), \Delta t \in [0, L(t)]) \leq M^*(t).$$

In other words, $M^*(t)$ is an upper bound of the actual intensity in $[t, t+L(t)]$. It is noteworthy that if $M^*(t)$ is chosen to be too high, most sampled points would be rejected and would lead to an inefficient simulation.

When simulating a process with decreasing inter-event intensity, such as the Hawkes process, $M^*(t)$ and $L^*(t)$ can be simply chosen to be $\lambda^*(t)$ and ∞. When simulating a process with increasing inter-event intensity, such as the self-correcting process, $L^*(t)$ is often empirically chosen to be $2/\lambda^*(t)$, since the next event is very likely to arrive before twice the mean interval length at the beginning of the interval. $M^*(t)$ is therefore $\lambda^*(t+L^*(t))$.

Algorithm 1 Otaga Modified Thinning Algorithm for Simulating a TPP

Input: Interval $[0, T]$, model parameters
$t \leftarrow 0, \mathcal{H} \leftarrow \emptyset$
while true do
  Compute $m \leftarrow M(t|\mathcal{H})$, $l \leftarrow L(t|\mathcal{H})$
  Draw $\Delta t \sim \text{Exp}(m)$ (exponential distribution with mean $1/m$)
  if $t + \Delta t > T$ **then**
    return $\mathcal{H}$
  end if
  if $\Delta t > l$ **then**
    $t \leftarrow t + l$
  else
    $t \leftarrow t + \Delta t$
    Compute $\lambda = \lambda^*(t)$
    Draw $u \sim \text{Unif}(0, 1)$
    if $\lambda/m > u$ **then**
      $\mathcal{H} = \mathcal{H} \cup t$
    end if
  end if
end while

B.2 STPP SIMULATION

It has been mentioned in Section 3.2 that an STPP can be seen as attaching the locations sampled from $f^*(\mathbf{s}|t)$ to the events generated by a TPP. Simulating an STPP is basically adding one step to Algorithm 1: sample a new location from $f^*(\mathbf{s}|t)$ after retaining a new event at t.

As for a spatiotemporal self-correcting process, neither $f^*(\mathbf{s}, t)$ nor $\lambda^*(t)$ has a closed form, so the process's spatial domain has to be discretized for simulation. $\lambda^*(t)$ can be approximated by $\sum_{\mathbf{s}\in\mathcal{S}} \lambda^*(\mathbf{s}, t)/|\mathcal{S}|$, where $\mathcal{S}$ is the set of discretized coordinates. $L^*(t)$ and $M^*(t)$ are chosen to be $2/\lambda^*(t)$ and $\lambda^*(t + L^*(t))$. Since $f^*(\mathbf{s}|t)$ is proportional to $\lambda^*(\mathbf{s}, t)$, sampling a location from $f^*(\mathbf{s}|t)$ is implemented as sampling from a multinomial distribution whose probability mass function is the normalized $\lambda^*(\mathbf{s}, t)$.

Table 3: The Parameter Settings for the Synthetic Dataset

		α	β	μ	g0 covariance	g2 covariance
ST-Hawkes	DS1	.5	1	.2	[.2 0; 0 .2]	[0.5 0; 0 0.5]
	DS2	.5	.6	.15	[5 0; 0 5]	[.1 0; 0 .1]
	DS3	.3	2	1	[1 0; 0 1]	[.1 0; 0 .1]
ST-Self Correcting	DS1	.2	.2	1	[1 0; 0 1]	[0.85 0; 0 0.85]
	DS2	.3	.2	1	[.4 0; 0 .4]	[.3 0; 0 .3]
	DS3	.4	.2	1	[.25 0; 0 .25]	[.2 0; 0 .2]

B.3 STHP Simulation

To simulate a spatiotemporal Hawkes process with Gaussian kernel, we mainly followed an efficient procedure proposed by Zhuang (2004), that makes use of the clustering structure of the Hawkes process and thus do not require repeated calculations of $\lambda^*(\mathbf{s}, t)$.

Algorithm 2 Simulating spatiotemporal Hawkes process with Gaussian kernel

1: Generate the background events $G^{(0)}$ with the intensity $\lambda^*(\mathbf{s}, t) = \mu g_0(\mathbf{s})$, i.e., simulate a homogenous Poisson process $\text{Pois}(\mu)$ and sample each event's location from a bivariate Gaussian distribution $\mathcal{N}(\mathbf{s}_\mu, \Sigma)$
2: $\ell = 0, S = G^{(\ell)}$
3: **while** $G^\ell \neq \emptyset$ **do**
4: **for** $i \in G^\ell$ **do**
5: Simulate event i's offsprings $O_i^{(\ell)}$ with the intensity $\lambda^*(\mathbf{s}, t) = g_1(t, t_i) g_2(\mathbf{s}, \mathbf{s}_i)$, i.e., simulate a non-homogenous stationary Poisson process $\text{Pois}(g_1(t, t_i))$ by Algorithm 1 and sample each event's location from a bivariate Gaussian distribution $\mathcal{N}(\mathbf{s}_i, \Sigma)$
6: $G^{(\ell+1)} = \bigcup_i O_i^{(\ell)}, S = S \cup G^{(\ell+1)}, \ell = \ell + 1$
7: **end for**
8: **end while**
9: **return** S

B.4 Parameter Settings

For the synthetic dataset, we pre-specified both the STSCP's and the STHP's kernels $g_0(\mathbf{s})$ and $g_2(\mathbf{s}, \mathbf{s}_j)$ to be Gaussian:

$$g_0(\mathbf{s}) := \frac{1}{2\pi} |\Sigma_{g0}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{s} - [0,0])\Sigma_{g0}^{-1}(\mathbf{s} - [0,0])^T\right)$$

$$g_2(\mathbf{s}, \mathbf{s}_j) := \frac{1}{2\pi} |\Sigma_{g2}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{s} - \mathbf{s}_j)\Sigma_{g2}^{-1}(\mathbf{s} - \mathbf{s}_j)^T\right)$$

The STSCP is defined on $\mathcal{S} = [0, 1] \times [0, 1]$, while the STHP is defined on $\mathcal{S} = \mathbb{R}^2$. The STSCP's kernel functions are normalized according to their cumulative probability in $\mathcal{S}$. Table B.4 shows the simulation parameters. The STSCP's spatial domain is discretized as an 101×101 grid during the simulation.

C Experiment Details

In this appendix, we include more details of our Neural-STPP parameter settings and some additional experiment results.

C.1 Model Setup Details

For a better understanding of our Neural-STPP, we list out the detailed hyperparameter setting and initial parameter values of the model in Table 5.

Name	Value	Description
Optimizer	Adam	Optimizer of the rnn is set to Adam
Learning rate	3e-4	Decay the learning rate by 0.2 every 10 epochs
Momentum	0.9	-
Epoch	100	Train the rnn for 100 epochs for 1 step prediction
Batch size	64	-
Hidden size	128	The rnn has 128 features in the hidden state
$M \times N$	40×40	The number of grids for kernel density estimation in Section 4.1
γ	1e-4	A regularization term for historical information in Section 4.2

Table 4: Hyperparameter settings for training Neural-STPP on Foursquare Dataset.

Name	Value	Description
w^t	0.1	Scalar for current time influence in Section 4.1
b	0.1	Base time intensity in Section 4.1

Table 5: Initial parameter values for training Neural-STPP on Foursquare Dataset.

All of the hyperparameters are subject to change for different datasets and fine-tune these parameters can always give out better results.

C.1.1 Validation of Grid Size

We validated our choice of the grid size $M \times N = 40 \times 40$ by grid search, see Figure 7.

C.2 Additional Experiment Result

	ST-Hawkes			ST-Self Correcting		
	DS1	DS2	DS3	DS1	DS2	DS3
GRU	**2.70**$\pm$0.01	2.15$\pm$0.08	**0.97**$\pm$0.00	0.59$\pm$0.00	0.67$\pm$0.00	0.88$\pm$0.01
Neural ODE	2.75$\pm$0.04	2.16$\pm$0.11	**0.97**$\pm$0.00	0.59$\pm$0.00	0.68$\pm$0.00	0.87$\pm$0.01
Neural ODE++	2.76$\pm$0.02	2.15$\pm$0.09	**0.97**$\pm$ 0.00	**0.58**$\pm$0.00	**0.65**$\pm$0.01	0.85$\pm$0.01
RMTPP	2.73$\pm$0.02	2.42$\pm$0.06	**0.97**$\pm$0.00	0.59$\pm$0.00	0.67$\pm$0.00	0.86$\pm$0.00
Neural-STPP	2.73$\pm$0.02	**2.10**$\pm$0.04	**0.97**$\pm$0.00	**0.58**$\pm$0.00	0.67$\pm$0.00	**0.82**$\pm$0.00

Table 6: Performance comparison of one-step prediction for models on synthetic data. Each column corresponds to a row in Appendix B.4 Table 3, which describes all these synthetic datasets in detail.

Synthetic Data We compare our Neural-STPP with other alternative baselines mentioned in Section 5 on the synthetic data simulated by the synthetic models in Table 3. Table 6 shows the space and time joint RMSE of one step prediction of our Neural-STPP and baselines on synthetic data where lower RMSE indicates better performance. Each column represents a different synthetic data and each row represents a prediction model. When evaluating on the same dataset, our Neural-STPP is among the best two models in all cases which indicates its generally great performance on synthetic data. In addition, Neural-STPP always has the smallest standard deviation in comparison to other baselines, which implies its better stability.

Real-world data As is mentioned in Section 5, we use Neural Hawkes Process as a time prediction baseline. Table 7 describes the error of multi-step time prediction of Neural Hawkes Process. It is evident that Neural Hawkes Process performs the worst as the time error is much larger than that of the other models on all datasets.

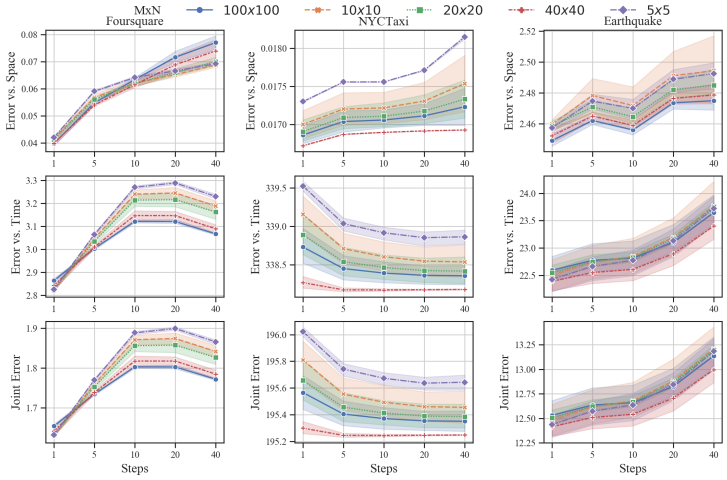


Figure 7: Ablation study of grid size $M \times N$ on real-world datasets.

	1	5	10	20	40
Foursquare	3.81	3.87	4.02	4.04	4.00
NYCTaxi	551.08	566.04	583.86	605.50	623.28
Earthquake	23.77	24.45	25.00	25.64	26.25

Table 7: Performance of multi-step time prediction for Neural Hawkes Process on real-world data. Each row represents a different dataset. Each column represents a specific number of steps ahead.

D Alternative Design and Negative Results

Before reaching at our current design, we had attempted two alternative designs which both failed. These negative results provide evidence on why we favor our current design. We present the negative results here in the hope of reducing others' efforts to duplicate similar experiments.

D.1 NSTPPv1: moving s into the power of exp

Intuitively, moving the space and time representations into the power of exp would allow the space and time representations to interact more closely.

Our first proposed model is as below,

$$\lambda^*(\mathbf{s}, t) = \exp\left(\mu g_0(\mathbf{s})(t - t_i) + \sum_{t_i < t} \boldsymbol{v}^{t^T} \mathbf{h}_{i-1} g_2(\mathbf{s}, \mathbf{s}_i)\right)$$

where the variable definitions are the same as in the main text. This model is more flexible. When the spatial domain $\mathcal{S}$ is bounded, the kernel density function $g_0(\mathbf{s})$ and $g_2(\mathbf{s}, \mathbf{s}_i)$ can be replaced by neural networks of $\mathbf{s}$ to further increase its expressivity.

However, the key problem of the model is that the negative likelihood, $\exp(-\int_T^\infty \int_\mathcal{S} \lambda^*(\mathbf{s}, t))$, has to be approximated using expensive triple numerical integration (see Appendix A.2.4). While the integral evaluated via the trapezoidal rule supports backpropagation, the model needs to discretize the space, losing its decisive advantage over marked temporal point process models. Additionally, triple summations at a reasonable spatial resolution are extremely costly. If the spatial resolution is not high enough, the peak of the kernel function cannot be precisely learned.

The upper bound of the time (integrating from t to ∞) has to be determined empirically for different datasets. The integral estimate via Monte Carlo sampling is not differentiable, as the simulation requires rejection sampling and the conditional intensity (see Appendix B.3). Inference also requires numerical integration.

In our design, we can apply the sum rule of integration and have a tractable $\lambda^*(t)$. Our experiments show that such a numerically convenient design also enjoys a favorable empirical performance.

D.2 NSTPPv2: MODEL THE CHANGE IN INTENSITY BEFORE AND AFTER THE EVENT

We have seen the importance of having a tractable conditional intensity. Another design that is tractable is to model the conditional intensity before and after the event separately. Denote the conditional intensity before the event t_i as λ^*_{i-} and that after the event t_i as λ^*_{i+}. The spatiotemporal intensity is as follows:

$$\lambda^*(\mathbf{s}, t) = g_0(s)\lambda_{0-} + \sum_{i=0}^{j} g_2(\mathbf{s}, \mathbf{s}_i)(\lambda^*_{i+} - \lambda^*_{i-}) \exp(w^t(t - t_i)) \tag{12}$$

Using the same conditional intensity in the main text:

$$\begin{aligned}
\lambda^*_{i+} &= \exp({\boldsymbol{v}^t}^T h_{i+1} + b^t), \\
\lambda^*_{i-} &= \exp({\boldsymbol{v}^t}^T h_i + w^t(t_i - t_{i-1}) + b^t).
\end{aligned}$$

The spatial integral of $\lambda(s, t)$ is a telescoping series:

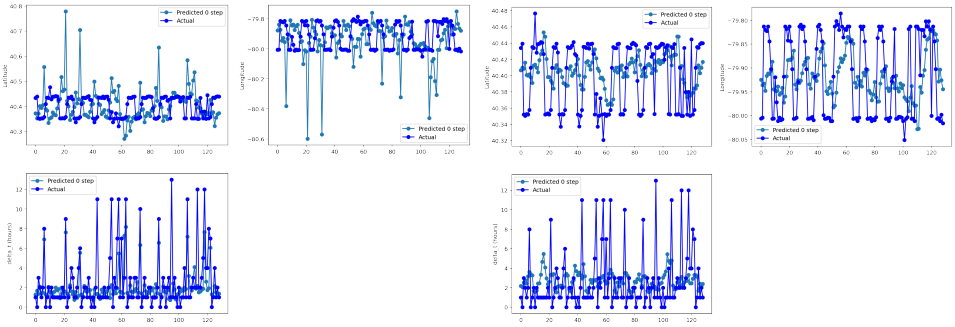
$$\begin{aligned}
\int_{\mathcal{S}} \lambda^*(s, t) &= \exp({\boldsymbol{v}^t}^T h_0 + w^t(t - t_{-1}) + b^t) + \sum_{i=0}^{j} (\lambda^*_{i+} - \lambda^*_{i-}) \exp(w^t(t - t_i)) \\
&= \exp({\boldsymbol{v}^t}^T h_0 + w^t(t - t_{-1}) + b^t) \\
&\quad + \exp({\boldsymbol{v}^t}^T h_1 + w^t(t - t_0) + b^t) - \exp({\boldsymbol{v}^t}^T h_0 + w^t(t - t_{-1}) + b^t) \\
&\quad + \exp({\boldsymbol{v}^t}^T h_2 + w^t(t - t_1) + b^t) - \exp({\boldsymbol{v}^t}^T h_1 + w^t(t - t_0) + b^t) + ... \\
&\quad + \exp({\boldsymbol{v}^t}^T h_{j+1} + w^t(t - t_j) + b^t) \\
&= \exp({\boldsymbol{v}^t}^T h_{j+1} + w^t(t - t_j) + b^t) = \lambda^*(t)
\end{aligned}$$

which evaluates to a tractable conditional intensity.

Essentially, NSTPPv2 directly multiplies each event's temporal intensity change by the event's spatial pdf. In contrast, in the current design, an event's location wouldn't interact with the event's time until being combined with other events' locations ($f^*(\mathbf{s}|t)$). NSTPPv2 features a more straightforward spatiotemporal interaction while retaining a tractable conditional intensity. The spatial and temporal inferences also only need 1-dimensional numerical integration, as shown below:

$$\begin{aligned}
&\int_{t_j}^{\infty} \int_{\mathcal{S}} t f^*(\mathbf{s}, t) = \int_{t_j}^{\infty} t f^*(t) \\
&= \int_{t_j}^{\infty} t \exp({\boldsymbol{v}^t}^T h_j + w^t(t - t_j) + b^t + \frac{1}{w}(\exp({\boldsymbol{v}^t}^T h_j + b^t) - \exp({\boldsymbol{v}^t}^T h_j + w^t(t - t_j) + b^t)))
\end{aligned}$$

$$\begin{aligned}
&\int_{t_j}^{\infty} \int_{\mathcal{S}} \mathbf{s} f^*(\mathbf{s}, t) \\
&= \int_{t_j}^{\infty} \exp\left(-\int_{t_j}^{t} \lambda^*(\tau) d\tau\right) \int_{\mathcal{S}} \mathbf{s} \lambda^*(\mathbf{s}, t) \\
&= \int_{t_j}^{\infty} \exp\left\{\frac{1}{w^t}[\exp({\boldsymbol{v}^t}^T h_j + b^t) - \exp({\boldsymbol{v}^t}^T h_j + w^t(t - t_j) + b^t)]\right\} \left(\int_{\mathcal{S}} s \lambda^*(\mathbf{s}, t)\right)
\end{aligned}$$



(a) NSTPPv2, $g_2(\mathbf{s}, \mathbf{s}_i) \sim \mathcal{N}(\mathbf{s}_i, \mathbf{I})$ (b) NSTPPv2, $g_2(\mathbf{s}, \mathbf{s}_i) \sim \mathcal{N}(\mathbf{s}_i, 0.01\mathbf{I})$

Figure 8: NSTPPv2 lat/lon/time predictions on Foursquare with different initial variances

$$
\begin{aligned}
&\int_{\mathcal{S}} \mathbf{s}\lambda^*(\mathbf{s}, t) \\
&= \mathbf{s}_\mu \exp({\boldsymbol{v}^t}^T h_0 + w^t(t - t_{-1}) + b^t) + \sum_i \mathbf{s}_i(\lambda_{i+} - \lambda_{i-}) \exp(w^t(t - t_i)) \\
&= \mathbf{s}_\mu \exp({\boldsymbol{v}^t}^T h_0 + w^t(t - t_{-1}) + b^t) \\
&\quad + \mathbf{s}_0 \exp({\boldsymbol{v}^t}^T h_1 + w^t(t - t_0) + b^t) - \mathbf{s}_0 \exp({\boldsymbol{v}^t}^T h_0 + w^t(t - t_{-1}) + b^t) \\
&\quad + \mathbf{s}_1 \exp({\boldsymbol{v}^t}^T h_2 + w^t(t - t_1) + b^t) - \mathbf{s}_1 \exp({\boldsymbol{v}^t}^T h_1 + w^t(t - t_0) + b^t) + ... \\
&\quad + \mathbf{s}_j \exp({\boldsymbol{v}^t}^T h_{j+1} + w^t(t - t_j) + b^t) \\
&= (\mathbf{s}_\mu - \mathbf{s}_0) \exp({\boldsymbol{v}^t}^T h_0 + w^t(t - t_{-1}) + b^t) \\
&\quad + (\mathbf{s}_0 - \mathbf{s}_1) \exp({\boldsymbol{v}^t}^T h_1 + w^t(t - t_0) + b^t) + ... + \mathbf{s}_j \exp({\boldsymbol{v}^t}^T h_{j+1} + w^t(t - t_j) + b^t) \\
&= (\mathbf{s}_\mu - \mathbf{s}_0) \exp({\boldsymbol{v}^t}^T h_0 + w^t(t - t_{-1}) + b^t) \\
&\quad + \sum_{i=1}^{j} (\mathbf{s}_{i-1} - \mathbf{s}_i) \exp({\boldsymbol{v}^t}^T h_i + w^t(t - t_{i-1}) + b^t) + \mathbf{s}_j \exp({\boldsymbol{v}^t}^T h_{j+1} + w^t(t - t_j) + b^t)
\end{aligned}
$$

However, NSTPPv2 has a major weakness: the intensity's positivity is not guaranteed. If we cast a rectifier over the intensity function and $\lambda^*(\mathbf{s}, t)$ is negative somewhere in $\mathcal{S}$, the nice relationship $\int_{\mathcal{S}} \lambda^*(\mathbf{s}, t) = \exp({\boldsymbol{v}^t}^T h_{j+1} + w^t(t - t_j) + b^t)$ would no longer hold. Accordingly, we have to constrain the model's parameters to stay positive during the training, which significantly limits the model's expressivity. We have tested the model on the Foursquare dataset, see Figure 8. We can see that when the kernel variance is initialized to be 1 (relatively large when the locations are normalized), the model is not able to reduce the spatial kernel variance, since a sharper kernel is likely to produce negative intensity at its peak. The resulting spatial prediction is inaccurate, submitting to the temporal prediction.

When the kernel variance is relatively small, e.g. 0.01, the model is not able to learn the change in temporal intensity, since a substantial change in the temporal intensity is likely to break the constraint given sharp spatial kernels. The resulting temporal prediction is flat and inaccurate.

D.3 Summary

We have considered various alternative designs. While it is desirable to have a more flexible space-time intensity function, we are largely constrained by the difficulty of numerical integration in NSTPPv1 and the non-negativity constraints in NSTPPv2.

Consequently, to the best of our knowledge, a tractable model of the spatiotemporal intensity function shall summate outside the exponential, and separate the temporal and spatial parts of the intensity. Hence, our current design is developed based on these two conditions.