Spatio-temporal Diffusion Point Processes

Yuan Yuan, Jingtao Ding*, Chenyang Shao, Depeng Jin, Yong Li
Department of Electronic Engineering
Tsinghua University
Beijing, China

ABSTRACT

Spatio-temporal point process (STPP) is a stochastic collection of events accompanied with time and space. Due to computational complexities, existing solutions for STPPs compromise with conditional independence between time and space, which consider the temporal and spatial distributions separately. The failure to model the joint distribution leads to limited capacities in characterizing the spatio-temporal entangled interactions given past events. In this work, we propose a novel parameterization framework for STPPs, which leverages diffusion models to learn complex spatio-temporal joint distributions. We decompose the learning of the target joint distribution into multiple steps, where each step can be faithfully described by a Gaussian distribution. To enhance the learning of each step, an elaborated spatio-temporal co-attention module is proposed to capture the interdependence between the event time and space adaptively. For the first time, we break the restrictions on spatio-temporal dependencies in existing solutions, and enable a flexible and accurate modeling paradigm for STPPs. Extensive experiments from a wide range of fields, such as epidemiology, seismology, crime, and urban mobility, demonstrate that our framework outperforms the state-of-the-art baselines remarkably, with an average improvement of over 50%. Further in-depth analyses validate its ability to capture spatio-temporal interactions, which can learn adaptively for different scenarios. The datasets and source code are available online: https://github.com/tsinghua-fib-lab/Spatio-temporal-Diffusion-Point-Processes.

KEYWORDS

Spatio-temporal point processes, Diffusion models, Co-attention

1 INTRODUCTION

Spatio-temporal point process (STPP) is a stochastic collection of points, where each point denotes an event $x = (t, s)$ associated with time t and location s. STPP is a principled framework for modeling sequences consisting of spatio-temporal events, and have been applied in a wide range of fields, such as earthquakes and aftershocks [3, 34], disease spread [32, 38], urban mobility [9, 52], and emergencies [56, 62].

Spatio-temporal point processes have been widely studied in the literature [2, 3, 7, 13, 43, 58, 63] with rich theoretical foundations [5, 14, 23]. Due to computational complexities, a general approach for STPPs is to characterize the event time and space with distinct models. Conventional STPP models [7, 13, 43] mainly capture relatively simple patterns of spatio-temporal dynamics, where the temporal domain is modeled by temporal point process models, such as poisson process [23], hawkes process [17], and

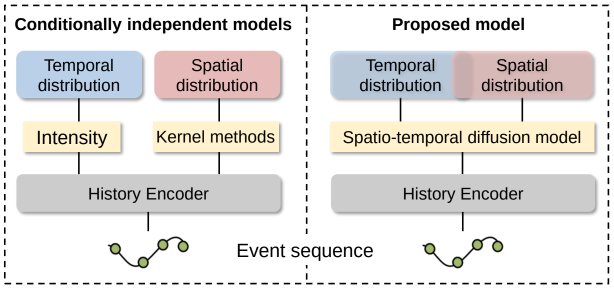


Figure 1: High-level comparison between our proposed framework and conditionally independent solutions for modeling STPPs. Our framework can directly learn the spatio-temporal joint distribution without any model restrictions.

self-correcting process [21], and the spatial domain is usually fitted by kernel density estimators (KDE) [54]. With the advance of neural networks, a series of neural architectures are proposed to improve the fitting accuracy [3, 22, 61], however, they still adopt the approach of separate modeling. For example, Chen et al. [3] use neural ODEs and continuous-time normalizing flows (CNFs) to learn the temporal distribution and spatial distribution, respectively. Zhou et al. [61] apply two independent kernel functions for time and space, whose parameters are obtained from neural networks, to build the density function,

However, for STPPs, the time and space where an event occurs are highly dependent and entangled with each other. For example, in seismology, earthquakes are spatio-temporal correlated due to crust movements [53], which occur with a higher probability close in time and space to previous earthquakes. Take urban mobility as another example, people are more likely to go to work during the day, while tend to go for entertainment at night. Therefore, it is crucial to learn models that can address the spatio-temporal joint distribution conditioned on the event history. However, it is non-trivial due to the following two challenges:

(1) **Spatio-temporal joint distributions for STPPs usually have tremendous sample spaces, which are highly intractable.** Directly fitting requires huge training samples, which is prohibitive in practice. The general approach is to decompose the target distribution into conditionally dependent distributions [3, 5], fitting the temporal density $p^*(t)$[1] and conditional density $p^*(s|t)$ separately. However, the characterization of $p^*(s|t)$ is largely limited to certain model structures, such as KDEs and CNFs, which are less expressive.
(2) **The occurrence of events is usually associated with complex coupling correlations between time and space.** Driven by different generation mechanisms, the occurrence of events

*Jingtao Ding is the corresponding author (dingjt15@tsinghua.org.cn).

[1]We use the common star superscript to denote conditional dependence on the history.

exhibits distinct spatio-temopral dependencies across various fields. How to effectively capture the underlying dependence for an event still remains an open problem.

Solving the above two challenges calls for a new modeling paradigm for STPPs. In this paper, we propose a novel parameterization framework, Spatio-Temporal Diffusion Point Processes (DSTPP), which is capable of leaning spatio-temporal joint distributions effectively. By leveraging denoising diffusion probabilistic modeling, we manage to decompose the original complex distribution into a Markov chain of multiple steps, where each step corresponds to a minor distribution change and can be modeled faithfully by a Gaussian distribution [40, 47]. The target distribution is learned throughout the combination of all steps, where the predicted joint distribution obtained from the previous step acts as the condition for the next-step learning. In this way, conditioned on the already predicted results, the modeling of time and space becomes independent at the current step, i.e., $p^*(t_{\text{current}}|t_{\text{last}}, s_{\text{last}})$ and $p^*(s_{\text{current}}|t_{\text{last}}, s_{\text{last}})$, which successfully solves the intractable problem of the conditional density $p^*(s|t)$. This novel learning paradigm completely removes the constraints of model structure parameterization in existing solutions, allowing accurate and flexible modeling of STPPs.

The multi-step learning process simulates the generation of the spatio-temporal joint distribution, however, the underlying mechanism of each step is still unclear. To further facilitate the learning at each step, we design a spatio-temporal co-attention module to characterize spatio-temporal interactions that contribute to the target joint distribution. Specifically, we simultaneously learn spatial attention and temporal attention to capture their fine-grained interactions adaptively, which characterizes underlying mechanisms of the joint distribution. Table 1 compares the advantages of our framework with existing solutions. Our DSTPP can learn spatio-temporal joint distributions without any dependence restrictions. As no integrals or Monte Carlo approximations are required, it is flexible and can perform sampling in a closed form. It can also be utilized to model a variety of STPPs, where events are accompanied with either a vector of real-valued spatial location or a discrete value, e.g., a class label of the location; thus it is broadly applicable in real-world scenarios.

We summarize our contributions as follows:

- To the best of our knowledge, we are the first to model STPPs within the diffusion model paradigm. By removing integrals and overcoming structural design limitations in existing solutions, it achieves flexible and accurate modeling of STPPs.
- We propose a novel spatio-temporal point process model, DSTPP. On the one hand, the diffusion-based approach decomposes the complex spatio-temporal joint distribution into tractable distributions. On the other hand, the elaborated co-attention module captures the spatio-temporal interdependence adaptively.
- Extensive experiments demonstrate the superior performance of our approach for modeling STPPs using both synthetic and real-world datasets with an average improvement of over 50%. Further in-depth analyses validate that our model successfully captures spatio-temporal interactions for different scenarios in an adaptive manner.

Table 1: Comparison of the proposed model with other point process approaches regarding important properties.

Model	No Asmp.*	No Restr.**	Flexible	Closed-form sampling
Hawkes [17]	✗	✗	✗	✗
Self-correcting [21]	✗	✗	✗	✗
KDE [2]	-	-	✗	✓
CNF [3]	-	-	✗	✓
ST Hawkes [43]	✗	✗	✗	✗
RMTPP [9]	✗	✗	✓	✗
NHP [30]	✗	✗	✓	✗
THP [64]	✗	✗	✓	✗
SNAP [59]	✗	✗	✓	✗
LogNormMix [45]	✗	✗	✗	✓
NJSDE [22]	✗	✗	✓	✗
Neural STPP [3]	✓	✗	✓	✗
DeepSTPP [61]	✗	✗	✓	✗
DSTPP (ours)	✓	✓	✓	✓

* Without assumptions of conditional spatio-temporal independence.
** Without dependence restrictions between time and space.

2 PRELIMINARIES

2.1 Spatio-temporal Point Process

A spatio-temporal point process is a stochastic process composed of events with time and space that occur over a domain [33]. These spatio-temporal events are described in continuous time with spatial information. The spatial domain of the event can be recorded in different ways. For example, in earthquakes, it is usually recorded as longitude-latitude coordinates in continuous space. It can also be associated with discrete labels, such as the neighborhoods of crime events. Let $x_i = (t_i, s_i)$ denotes the i_{th} spatio-temporal event written as the pair of occurrence time $t \in \mathbb{T}$ and location $s \in \mathbb{S}$, where $\mathbb{T} \times \mathbb{S} \in \mathbb{R} \times \mathbb{R}^d$. Then a spatio-temporal point process can be defined as a sequence $S = \{x_1, x_2, ..., x_L\}$, and the number of events L is also stochastic. Let $H_t = \{x_i | t_i < t, x_i \in S\}$ denote the event history before time t, modeling STPPs is concerned with parameterizing the conditional probability density function $p(t, s|H_t)$, which denotes the conditional probability density of the next event happening at time t and space s given the history H_t.

Discussion on shortcomings. In existing methods for STPPs, given the event history, space and time are assumed to be conditionally independent [9, 27, 30, 43, 61, 64] or unilaterally dependent [3, 5] i.e., the space is dependent on the time by $p(x|t)$. These dependence restrictions destroy the model's predictive performance on entangled space and time interactions conditioned on history. Besides, most approaches require integration operations when calculating the likelihood, or limit intensity functions to integrable forms, leading to a trade-off between accuracy and efficiency. We compare the shortcomings of existing approaches in Table 1[2], which motivate us to design a more flexible and effective model.

2.2 Denoising Diffusion Probabilistic Models

Diffusion models [19] generate samples by learning a distribution that approximates a data distribution. The distribution is learned by a gradual reverse process of adding noise, which recovers the actual value starting from Gaussian noise. At each step of the denoising process, the model learns to predict a slightly less noisy value.

[2]TPP models can be used for STPPs where the space acts as the marker.

Let $x^0 \sim q(x^0)$ denote a multivariate variable from specific input space $X \in \mathbb{R}^D$, and we consider a probability density function $p_\theta(x^0)$, which aims to approximate $q(x^0)$. Diffusion models are latent variable models, which are defined by two processes: the forward diffusion process and the reverse denoising process. Let X^k for $t = 1, 2, ..., K$ denote a sequence of latent variables of dimension $\in \mathbb{R}^D$, the forward diffusion process is defined by a Markov chain:

$$q(x^{1:K}|x^0) = \prod_{k=1}^{K} q(x^k|x^{k-1}) \ , \tag{1}$$

where $q(x^k|x^{k-1}) := \mathcal{N}(x^k; \sqrt{1-\beta_k}x^k \text{ and } \beta_k \boldsymbol{I}), \beta_1, ..., \beta_K \in (0, 1)$ is a given increasing variance schedule, representing a particular noise level. x^k can be sampled in a closed form as $q(x^k|x^0) = (x^k; \sqrt{\overline{\alpha}_k}x^0, (1-\overline{\alpha}_k)\boldsymbol{I})$, where $\alpha_k := 1-\beta_k$ and $\overline{\alpha}_k = \prod_{k=1}^{K} \alpha_k$. Then a noisy observation at the k_{th} step can be expressed as $x^k = \sqrt{\overline{\alpha}_k}x^0 + (1-\overline{\alpha}_k)\epsilon$, where $\epsilon \sim \mathcal{N}(0, \boldsymbol{I})$ and x^0 is the clean observation.

On the contrary, the reverse denoising process recovers x^0 starting from x^K, where $x^K \sim \mathcal{N}(x^K; 0, \boldsymbol{I})$. It is defined by the following Markov chain with learned Gaussian transitions:

$$\begin{aligned} p_\theta(x^{0:K}) &:= p(x^K) \prod_{k=1}^{K} p_\theta(x^{k-1}|x^k) \ , \\ p_\theta(x^{k-1}|x^k) &:= \mathcal{N}(x^{k-1}; \mu_\theta(x^k, k), \sigma_\theta(x^k, k)\boldsymbol{I}) \ , \end{aligned} \tag{2}$$

$p_\theta(x^{k-1}|x^k)$ aims to remove the Gaussian noise added in the forward diffusion process. The parameter θ can be optimized by minimizing the negative log-likelihood via a variational bound:

$$\min_\theta \mathbb{E}_{q(x^0)} \leq \min_\theta \mathbb{E}_{q(x^{0:K})} [-\log p(x^K) - \sum_{k=1}^{K} \log \frac{p_\theta(x^{k-1}|x^k)}{q(x^k|x^{k-1})}] \ . \tag{3}$$

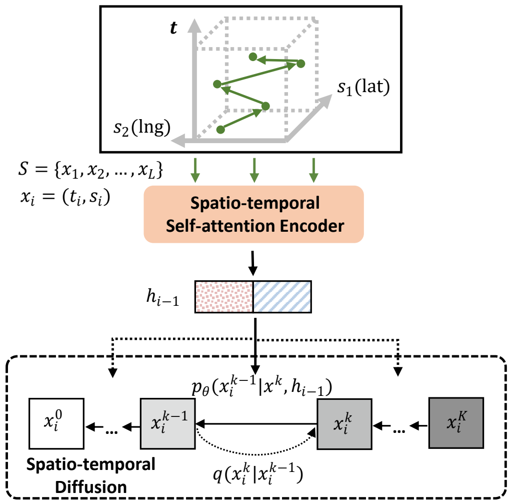
Ho et al. [19] show that the denoising parameterization can be trained by the simplified objective:

$$\mathcal{E}_{x^0 \sim q(x^0), \epsilon \sim \mathcal{N}(0, \boldsymbol{I})} [\|\epsilon - \epsilon_\theta(x_k, k)\|^2] \ , \tag{4}$$

where $x^k = \sqrt{\overline{\alpha}_k}x^0 + (1-\overline{\alpha}_k)\epsilon$. ϵ_θ needs to estimate Gaussian noise added to the input x^k, which is trained by MSE loss between the real noise and predicted noise. Therefore, ϵ_θ acts as the denoising network to transform x^k to x^{k-1}. Once trained, we can sample x^{k-1} from $p_\theta(x^{k-1}|x^k)$ and progressively obtain x^0 according to Equation (2).

3 SPATIO-TEMPORAL DIFFUSION POINT PROCESSES

Figure 2 illustrates the overall framework of DSTPP, which consists of two key modules, the spatio-temporal self-attention encoder and the spatio-temporal diffusion model. The spatio-temporal encoder learns an effective representation of the event history, then it acts as the condition to support the spatio-temporal denoising diffusion process. We first present the spatio-temporal encoder in Section 3.1. Then we formulate the learning of the spatio-temporal joint distribution as a denoising diffusion process, and introduce the diffusion process and inverse denoising process in Section 3.2. We describe how to train this model and perform sampling in Section 3.3. Finally, We demonstrate the detailed architecture of the denoising network parametrization in Section 3.4.



t
s_1(lat)
s_2(lng)
$S = \{x_1, x_2, ..., x_L\}$
$x_i = (t_i, s_i)$
Spatio-temporal Self-attention Encoder
h_{i-1}
$p_\theta(x_i^{k-1}|x^k, h_{i-1})$
x_i^0
x_i^{k-1}
x_i^k
x_i^K
Spatio-temporal Diffusion
$q(x_i^k|x_i^{k-1})$

Figure 2: The overview of the proposed DSTPP framework.

3.1 Spatio-temporal Encoder

To model the spatio-temporal dynamics of events and obtain effective sequence representations, we design a self-attention-based spatio-temporal encoder. The input of the encoder is made up of events $x = (t, s)$. To obtain a unique representation for each event, we use two embedding layers for the time and space separately. For the space $s \in \mathbb{R}^n$, we utilize a linear embedding layer; for the timestamp, we apply a positional encoding method following [64]:

$$[e_t]_j = \begin{cases} cos(t/10000^{\frac{j-1}{M}}) & \text{if } j \text{ is odd} \\ sin(t/10000^{\frac{j-1}{M}}) & \text{if } j \text{ is even} \end{cases} \ , \tag{5}$$

where e_t denotes the temporal embedding and M is the embedding dimension. For the spatial domain, we use linear projection to convert continuous or discrete space into embeddings as follows:

$$e_s = W_e s \tag{6}$$

where W_e contains learnable parameters. We use $W_e \in \mathcal{R}^{M \times D}$ if the space s is defined in the continuous domain $\mathbb{R}^D, D \in \{1, 2, 3\}$. We use $W_e \in \mathcal{R}^{M \times N}$ if the spatial information is associated with discrete locations represented by one-hot ID encoding $s \in \mathbb{R}^N$, where N is the number of discrete locations. In this way, we obtain real-value vectors e_s for both continuous and discrete spatial domains. For each event $x = (t, s)$, we obtain the spatio-temporal embedding e_{st} by adding the positional encoding e_t and spatial embedding e_s. The embedding of the $S = \{(t_i, s_i)\}_{i=1}^L$ is then specified by $E_{st} = \{e_{st,1}, e_{st,2}, ..., e_{st,L}\} \in \mathbb{R}^{L \times M}$, where $e_{st,i} = e_{s,i} + e_{t,i}$. In the meantime, we also keep the temporal embedding $E_t = \{e_{t,1}, e_{t,2}, ..., e_{t,L}\}$ and spatial embedding $E_s = \{e_{s,1}, e_{s,2}, ..., e_{s,L}\}$, respectively, with the goal of capturing characteristics of different aspects.

After the initial spatial embedding and temporal encoding layers, we pass E_{st}, E_s, and E_t through three self-attention modules. Specifically, the scaled dot-product attention [51] is defined as:

Algorithm 1 Training for each spatio-temporal event $x_i = (\tau_i, s_i)$

```
Input: $h_{i-1}$
Repeat: $x_i^0 \sim q(x_i^0)$,
    $k \sim \text{Uniform}(1, 2, ..., K)$
    $\epsilon \sim \mathcal{N}(0, I)$
    Take gradient descent step on
        $\nabla_{\phi,\theta} \|\epsilon - \epsilon_\theta(\sqrt{\overline{\alpha}_k} x_i^0 + \sqrt{1-\overline{\alpha}_k}\epsilon, h_{i-1}, k)\|^2$
Until: Converged
```

Algorithm 2 Sampling s_i^0 and τ_i^0

```
Input: Noise $s_i^K \sim \mathcal{N}(0, I)$, $\tau_i^K \sim \mathcal{N}(0, I)$ and $h_{i-1}$
  for k = K to 1 do
    $z_s \sim \mathcal{N}(0, I), z_t \sim \mathcal{N}(0, I)$ if k>1 else $z_s = 0, z_t = 0$
    $s_i^{k-1} = \frac{1}{\sqrt{\alpha_k}}(s_i^k - \frac{\beta_k}{\sqrt{1-\overline{\alpha}_k}}\epsilon_\theta(s_i^k, \tau_i^k, h_{i-1}, k)) + \sqrt{\beta_k} z_s$
    $\tau_i^{k-1} = \frac{1}{\sqrt{\alpha_k}}(\tau_i^k - \frac{\beta_k}{\sqrt{1-\overline{\alpha}_k}}\epsilon_\theta(s_i^k, \tau_i^k, h_{i-1}, k)) + \sqrt{\beta_k} z_t$
  end for
Return: $s_i^0, \tau_i^0$
```

$$\begin{aligned} \text{Attention}(Q, K, V) &= \text{Softmax}(\frac{QK^T}{\sqrt{d}}) \ , \\ S &= \text{Attention}(Q, K, V)V \ , \end{aligned} \tag{7}$$

where Q, K, and V represent queries, keys, and values. In our case, the self-attention operation takes the embedding E as input, and then converts it into three matrics by linear projections:

$$Q = EW^Q, \ \ K = EW^K, \ \ V = EW^V \ , \tag{8}$$

where W^Q, W^K, and W^V are weights of lienar projections. Finally, we use a position-wise feed-forward network to transform the attention output S into the hidden representation $h(t)$.

For three embeddings E_s, E_t and E_{st} containing information of different aspects, we all employ the above self-attentive operation to generate hidden spatial representation $h_s(t)$, temporal representation $h_t(t)$, and spatial-temporal representation $h_{st}(t)$. As a result, the hidden representation h_{i-1} in Figure 2 is a collection of the three representations.

3.2 Spatio-temporl Diffusion and Denoising Processes

Conditioned on the hidden representation h_{i-1} generated by the encoder, we aim to learn a model of the spatio-temporal joint distribution of the future event. The learning of such distribution is built on the diffusion model [19], and the values of space and time are diffused and denoised at each event. Specifically, for each event $x_i = (\tau_i, s_i)$ in the sequence, where τ_i denotes the time interval since the last event, we model the diffusion process as a Markov process over the spatial and temporal domains as $(x_i^0, x_i^1, ..., x_i^K)$, where K is the number of diffusion steps. From x_i^0 to x_i^K, we add a little Gaussian noise step by step to the space and time values until they are corrupted into pure Gaussian noise. The process of adding noise is similar to image scenarios, where the noise is applied independently on each pixel [19], we diffuse separately on the spatial and temporal domains by the following probabilities:

$$\begin{aligned} q_{st}(\boldsymbol{x}_i^k|\boldsymbol{x}_i^{k-1}) &\coloneqq (q(\tau_i^k|\tau_i^{k-1}), q(s_i^k|s_i^{k-1})) \ , \\ q(x^k|x^{k-1}) &\coloneqq \mathcal{N}(x^k; \sqrt{1-\beta_k}x^k, \beta_k \boldsymbol{I}) \ , \end{aligned} \tag{9}$$

where $\alpha_k = 1 - \beta_k$ and $\overline{\alpha}_k = \prod_{s=1}^k \alpha_k$.

On the contrary, we formulate the reconstruction of the point $x_i = (\tau_i, s_i)$ as reverse denoising iterations from x_i^K to x_i^0 given the event history. In addition to the history representation h_{i-1}, the denoising processes of time and space are also dependent on each other obtained in the previous step. The predicted values of the next step are modeled in a conditionally independent manner, which is formulated as follows:

$$p_\theta(x_i^{k-1}|x_i^k, h_{i-1}) = p_\theta(\tau_i^{k-1}|\tau_i^k, s_i^k, h_{i-1}) p_\theta(s_i^{k-1}|\tau_i^k, s_i^k, h_{i-1}) \ , \tag{10}$$

In this way, we manage to disentangle the modeling of spatio-temporal joint distribution into conditionally independent modeling, which enables effective and efficient modeling of the observed spatio-temporal distribution. The overall reverse denoising process is formulated as follows:

$$p_\theta(x_i^{0:K}|h_{i-1}) \coloneqq p(x_i^K) \prod_{k=1}^K p_\theta(x_i^{k-1}|x_i^k, h_{i-1}) \ . \tag{11}$$

For the continuous-space domain, the spatio-temporal distribution can be predicted by Equation 11. For the discrete-space domain, we add a rounding step at the end of the reverse process, $p_\theta(s_i|s_i^0)$, which converts the real-valued embedding s_i^0 to discrete location ID s_i.

3.3 Training and Inference

Training. For a spatio-temporal point process, the training should optimize the parameters θ that maximize the log-likelihood:

$$\sum_{i=1}^L \log p_\theta(x_i^0|h_{i-1}) \ , \tag{12}$$

where L is the number of events in the sequence. Based on a similar derivation in the preliminary section, we train the model by a simplified loss function for the i_{th} event and diffusion step k as follows [19]:

$$\mathcal{L} = \mathbb{E}_{x_i^0, \epsilon, k}[\|\epsilon - \epsilon_\theta(\sqrt{\overline{\alpha}_k}x_i^0 + \sqrt{1-\overline{\alpha}_k}\epsilon, h_{i-1}, k)\|^2] \ , \tag{13}$$

where $\epsilon \sim \mathcal{N}(0, I)$. Samples at each diffusion step k for each event are included in the training set. We train the overall framework consisting of ST encoder and ST diffusion in an end-to-end manner. The pseudocode of the training procedure is shown in Algorithm 1.

Inference. To predict future spatio-temporal events with trained DSTPP. We first obtain the hidden representation h_i by employing the spatio-temporal self-attention encoder given past $i-1$ events. Then, we can predict the next event starting from Gaussian noise $s_i^K, \tau_i^K \sim \mathcal{N}(0, I)$ conditioned on h_i. Specifically, the reconstruction of x_i^0 from $x_i^K = (s_i^K, \tau_i^K)$ is formulated as follows:

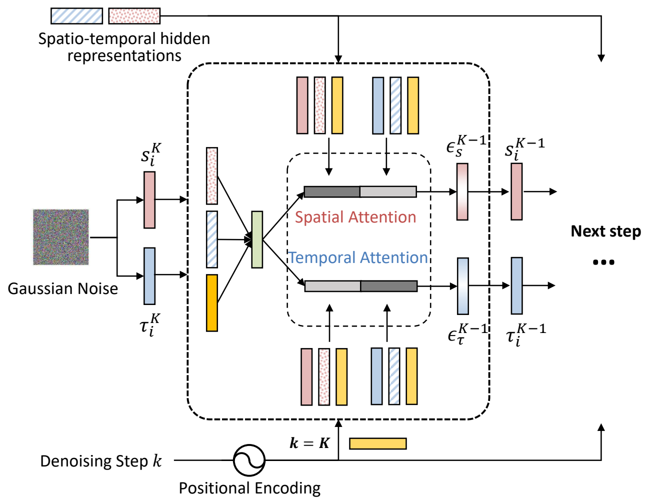


Figure 3: Network architecture of the spatio-temporal co-attention mechanism. Each step in the denoising process shares the same network structure, with spatio-temporal hidden representations as conditions.

$$
\begin{aligned}
s_i^{k-1} &= \frac{1}{\sqrt{\alpha_k}}\left(s_i^k - \frac{\beta_k}{\sqrt{1-\bar{\alpha}_k}}\epsilon_\theta(x_i^k, h_{i-1}, k)\right) + \sqrt{\beta_k} z_s \ , \\
\tau_i^{k-1} &= \frac{1}{\sqrt{\alpha_k}}\left(\tau_i^k - \frac{\beta_k}{\sqrt{1-\bar{\alpha}_k}}\epsilon_\theta(x_i^k, h_{i-1}, k)\right) + \sqrt{\beta_k} z_t \ ,
\end{aligned}
\tag{14}
$$

where z_s and z_t are both stochastic variables sampled from a standard Gaussian distribution. ϵ_θ is the trained reverse denoising network, which takes in the previous denoising result x_i^k, the hidden representation of the sequence history h_{i-1} and the diffusion step k. Algorithm 2 presents the pseudocode of the sampling procedure.

3.4 Co-attention Denoising Network

We design a co-attention denoising network to capture the interdependence between spatial and temporal domains, which facilitates the learning of spatio-temporal joint distributions. Specifically, it performs spatial and temporal attention simultaneously at each denoising step to capture fine-grained interactions. Figure 3 illustrates the detailed network architecture. Each step of the denoising process shares the same structure, which takes in the previously predicted values s_i^{k+1} and τ_i^{k+1}, and the denoising step k with positional encoding. Meanwhile, the network also integrates the hidden representation h_{i-1} to achieve conditional denoising.

Temporal attention aims to generate a context vector by attending to certain parts of the temporal input and certain parts of the spatial input, and so does spatial attention. We calculate the mutual attention weights, i.e., α_s and α_t, for space and time based on the condition h_{i-1} and current denoising step k as follows:

$$
\begin{aligned}
e_k &= \text{SinusoidalPosEmb}(k) \ , \\
\alpha_s &= \text{Softmax}(W_{sa}\text{Concat}(h_{i-1}, e_k) + b_{sa}) \ , \\
\alpha_t &= \text{Softmax}(W_{ta}\text{Concat}(h_{i-1}, e_k) + b_{ta}) \ ,
\end{aligned}
\tag{15}
$$

where $W_{sa}, W_{ta}, b_{sa}, b_{ta}$ are learnable parameters. α_s and α_t measure the mutual dependence between time and space, which are influenced by the event history and current denoising step.

Then we integrate the spatio-temporal condition $h_{i-1} = \{h_{s,i-1}, h_{t,i-1}\}$ into previously predicted values s_i^{k+1} and τ_i^{k+1} by feed-forward neural networks, and each layer is formulated as follows:

$$
\begin{aligned}
x_{s,i} &= \sigma(W_s s_i^{k+1} + b_s + W_{sh} h_{s,i-1} + b_{sh} + e_k) \ , \\
x_{t,i} &= \sigma(W_t \tau_i^{k+1} + b_t + W_{th} h_{t,i-1} + b_{th} + e_k) \ ,
\end{aligned}
\tag{16}
$$

where $W_s \in \mathbb{R}^{M\times D}$, $W_t \in \mathbb{R}^{M\times 1}$, $W_{sh}, W_{th} \in \mathbb{R}^{M\times M}$, and $b_s, b_t, b_{sh}, b_{th} \in \mathbb{R}^{M\times 1}$ are learnable parameters of the linear projection, and σ denotes the ReLU activation function. Finally, the outputs of spatial attention and temporal attention are calculated as follows:

$$
\begin{aligned}
x_i &= [x_{s,i}, x_{t,i}] \ , \\
\epsilon_{s,i}^k &= \sum \alpha_s x_i \ , \quad \epsilon_{t,i}^k = \sum \alpha_t x_i,
\end{aligned}
\tag{17}
$$

where $\epsilon_{s,i}^k$ and $\epsilon_{t,i}^k$ are the predicted noise at step k for the i_{th} event. We can obtain the predicted values s_i^k and τ_i^k at step k according to Equation (14). Then the predicted values s_i^k and τ_i^k are fed into the denoising network again to iteratively predict the results towards the clean values of space and time. In this way, the interdependence between time and space is captured adaptively and dynamically, facilitating the learning of the spatio-temporal joint distribution.

4 EXPERIMENTS

In this section, we perform experiments to answer the following research questions:

- **RQ1:** How does the proposed model perform compared with existing baseline approaches?
- **RQ2:** Is the joint modeling of spatial and temporal dimensions effective for STPPs, and what's the spatio-temporal interdependence like during the denoising process?
- **RQ3:** How does the total number of diffusion steps affect the performance?
- **RQ4:** How to gain a deeper understanding of the reverse denoising diffusion process?

4.1 Experimental Setup

4.1.1 ***Datasets.*** We perform extensive experiments on synthetic datasets and real-world datasets in the STPP literature. All datasets are obtained from open sources which contain up to thousands of spatio-temporal events. Varying across a wide range of fields, we use two synthetic datasets and four real-world datasets for continuous-space cases, including human mobility, earthquakes in Japan, COVID-19 spread, bike sharing in New York City, and simulated Hawkes Gaussian Mixture Model process and pinwheel structure data [3]. Besides, we use two real-world datasets, Atlanta Crime Data and NYC Taxi Data, the spatial locations of which are discrete neighborhoods. We briefly introduce the used datasets here, and further details can be found in Appendix A.

Mobility [57]. Check-in behaviors in Tokyo collected by Foursquare for two months. Each check-in is associated with timestamps and GPS coordinates.

Earthquakes. Earthquakes in Japan with a magnitude of at least 2.5 from 1990 to 2020 recorded by the U.S. Geological Survey[3].

COVID-19. Publicly released by The New York Times (2020), which records daily infected cases of COVID-19 in New Jersey state[4]. We aggregate the data at the county level.

Citibike. Bike sharing in New York City collected by a bike sharing service. The start of each trip is considered as an event.

HawkesGMM[5]. This synthetic data uses Gaussian Mixture Model to generate spatial locations. Events are sampled from a multivariate Hawkes process.

Pinewheel [3]. This synthetic data is generated from multi-modal and non-Gaussian spatial distributions containing 10 clusters in a pinwheel structure.

Crime [6]. It is provided by the Atlanta Police Department, recording robbery crime events. Each event is associated with the time and the neighborhood.

Taxi. It consists of pick-up records of taxis in New York City. Each pick-up is associated with the time and the neighborhood.

4.1.2 *Baselines.*

To evaluate the performance of our proposed model, we compare it with commonly-used methods and state-of-the-art models. The baselines can be divided into three groups: spatial baselines, temporal baselines, and spatio-temporal baselines. It is common for previous methods to model the spatial domain and temporal domain separately, so spatial baselines and temporal baselines can be combined freely for STPPs. Appendix B provides more details of the used baselines. We summarize the three groups as follows:

- **Spatial baselines:** We use conditional kernel density estimation (Condition KDE) [4], Continuous normalizing flow (CNF), and Time-varying CNF [4] (TVCNF) [4]. The three methods all model continuous spatial distributions.
- **Temporal baselines:** We include commonly used TPP models. Classical TPP models include the Poisson process [41], Hawkes Process [18], and Self-correcting process [21]. We also incorporate neural TPP models, including Recurrent Marked Temporal Point Process (RMTPP) [9], Neural Hawkes Process (NHP) [30], Transformer Hawkes Process (THP) [64], Self-attentive Hawkes Process (SAHP) [59]. Beisdes, we also compare with intensity-free approaches: Log Normal Mixture model (LogNormMix) [45], and Wasserstein GAN (WGAN) [55].
- **Spatio-temporal baseliens.** We include state-of-the-art spatio-temporal baselines, including Neural Jump Stochastic Differential Equations (NJSDE) [45], Neural Spatio-temporal Point Process (NSTPP) [3], and Deep Spatiotemporal Point Process (DeepSTPP) [61].

4.1.3 *Evaluation Metrics.*

We evaluate the performance of models from two perspectives: likelihood comparison and event prediction comparison. We use negative log-loglikelihood (NLL) as metrics, and the time and space are evaluated, respectively. Although the exact likelihood cannot be obtained, we can write the variational lower bound (VLB) according to Equation (3) and utilize it as the NLL metric instead. Thus, the performance on exact likelihood is even better than the reported variational lower bound. The models' predictive ability for time and space is also important in practical applications [35]. Since time intervals are real values, we use a common metric, Root Mean Square Error (RMSE), to evaluate time prediction. The spatial location can be defined in D-dimensional space, so we use Euclidean distance to measure the spatial prediction error. We refer the readers to Appendix C.1 for more details of the used evaluation metrics.

4.2 Overall performance (RQ1)

Table 2 and Table 3 show the overall performance of models on NLL and prediction, respectively. Figure 4 shows the prediction performance of models in discrete-space scenarios. Due to the space limit, some experiment results are presented in Appendix D. From these results, we have the following observations:

- **Unreasonable parametric assumptions for point processes destroy the performance severely.** The worst performance of the self-correcting process indicates the assumption that the occurrence of past events inhibits the occurrence of future events, does not match realities. On the contrary, the Hawkes process, which assumes the occurrence of an event increases the probability of event occurrence in the future, outperforms other classical models (Poisson and Self-correcting), with an obvious reduction of temporal NLL. Nevertheless, the self-exciting assumption can still fail when faced with cases where previous events prevent subsequent events. Therefore, classical models that require certain assumptions, cannot cover all situations with different dynamics.
- **It is necessary to capture the interdependence between spatial and temporal domains.** NSTPP models the dependence of space on time by $p(s|t)$, and the performance regarding spatial metrics is improved compared with independent modeling, including Conditional KDE, CNF and Time-varying CNF, with an average relative improvement of 17.8%. However, it does not outperform other TPP models in the temporal domain, suggesting that modeling the distribution $p(t|H_t)$ without conditioning on the space s fails to learn the temporal domain sufficiently.
- **DSTPP achieves the best performance across all datasets with multiple evaluation metrics.** In the continuous-space scenarios, regarding NLL, our model performs the best on both temporal and spatial domains. Compared with the second-best model, our model reduces the spatial NLL by over 50% on average. The performance on temporal NLL also achieves remarkably significant improvement across all datasets. In terms of models' predictive power, our model also achieves optimal performance, with remarkable improvements compared to second-best models: 56% to 90% in the spatial domain and 55% to 94% in the temporal domain. In addition, as Figure 4 shows, DSTPP delivers better predictive performance compared with other solutions in modeling discrete-space scenarios, especially the spatial prediction on the Crime dataset with an eight-fold performance boost. Our model can capture the spatio-temporal interdependence in an adaptive manner and further learn the joint distribution through a flexible framework that requires no parameter assumptions and MC estimations. These advantages enable it to achieve superior performance for STPPs.

[3]https://earthquake.usgs.gov/earthquakes/search/
[4]https://github.com/nytimes/covid-19-data
[5]https://github.com/facebookresearch/neural_stpp/blob/main/toy_datasets.py
[6]http://www.atlantapd.org/i-want-to/crime-data-downloads

Table 2: Performance evaluation for negative log-likelihood per event on test data. ↓ means lower is better. Bold denotes the best results and underline denotes the second-best results.

Model	Earthquake		COVID-19		Citibike		HawkesGMM	
	Spatial ↓	Temporal ↓	Spatial ↓	Temporal ↓	Spatial ↓	Temporal ↓	Spatial ↓	Temporal ↓
Conditional KDE	2.21±0.105	-[(1)]	2.31±0.084	-	2.74±0.001	-	0.236±0.001	-
CNF	1.35±0.000	-	2.05±0.014	-	2.15±0.000	-	0.427±0.002	-
TVCNF	1.34±0.008	-	2.04±0.004	-	2.19±0.025	-	0.431±0.008	-
Possion	-	-0.146±0.000	-	-0.876±0.021	-	-0.626±0.000	-	1.34±0.000
Hawkes	-	-0.514±0.000	-	-2.06±0.000	-	-1.06±0.001	-	0.880±0.000
Self-correcting	-	13.8±0.533	-	7.13±0.062	-	7.11±0.010	-	4.59±0.135
RMTPP	-	0.0930±0.051	-	-1.30±0.022	-	1.24±0.001	-	1.52±0.002
NHP	-	-0.676±0.001	-	-2.30±0.001	-	-1.14±0.001	-	0.580±0.000
THP	-	-0.976±0.011	-	-2.12±0.002	-	-1.49±0.003	-	-0.402±0.001
SAHP	-	-0.229±0.007	-	-1.37±0.118	-	-1.02±0.067	-	-1.25±0.136
LogNormMix	-	-0.341±0.071	-	-2.01±0.025	-	-1.06±0.005	-	0.630±0.004
NJSDE	1.65±0.012	0.0950±0.203	2.21±0.005	-1.82±0.002	2.63±0.001	-0.804±0.059	0.395±0.001	1.77±0.030
NSTPP	0.885±0.037	-0.623±0.004	1.90±0.017	-2.25±0.002	2.38±0.053	-1.09±0.004	0.285±0.011	0.824±0.005
DeepSTPP	4.92±0.007	-0.174±0.001	0.361±0.01	-1.09±0.01	-4.94±0.016	-1.13±0.002	0.519±0.001	0.322±0.002
DSTPP (ours)	**0.308±0.006**	**-1.96±0.020**	**-1.02±0.029**	**-3.08±0.003**	**-5.41±0.011**	**-3.36±0.010**	**-2.95±0.047**	**-3.42±0.002**

[(1)] **Spatial baselines and temporal baselines can be combined freely for modeling spatio-temporal domains.**

Table 3: Performance evaluation for predicting both time and space of the next event. We use Euclidean distance to measure the prediction error of the spatial domain and use RMSE between real intervals and predicted intervals for time prediction.

Model	Earthquake		COVID-19		Citibike		HawkesGMM	
	Spatial ↓	Temporal ↓	Spatial ↓	Temporal ↓	Spatial ↓	Temporal ↓	Spatial ↓	Temporal ↓
Conditional KDE	11.3±0.658	-	0.688±0.047	-	0.718±0.001	-	1.54±0.006	-
CNF	8.48±0.054	-	0.559±0.000	-	0.722±0.000	-	71663±60516	-
TVCNF	8.11±0.001	-	0.560±0.000	-	0.705±0.000	-	2.03±0.000	-
Possion	-	0.631±0.017	-	0.463±0.021	-	0.438±0.001	-	2.81±0.070
Hawkes	-	0.544±0.010	-	0.672±0.088	-	0.534±0.011	-	2.63±0.002
Self-correcting	-	11.2±0.486	-	2.83±0.141	-	10.7±0.169	-	9.72±0.159
RMTPP	-	0.424±0.009	-	1.32±0.024	-	2.07±0.015	-	3.38±0.012
NHP	-	1.86±0.023	-	2.13±0.100	-	2.36±0.056	-	2.82±0.028
THP	-	2.44±0.021	-	0.611±0.008	-	1.46±0.009	-	5.35±0.002
SAHP	-	0.409±0.002	-	0.184±0.024	-	0.203±0.010	-	2.75±0.049
LogNormMix	-	0.593±0.005	-	0.168±0.011	-	0.350±0.013	-	2.79±0.021
WGAN	-	0.481±0.007	-	0.124±0.002	-	0.238±0.003	-	2.83±0.048
NJSDE	9.98±0.024	0.465±0.009	0.641±0.009	0.137±0.001	0.707±0.001	0.264±0.005	1.62±0.003	2.25±0.007
NSTPP	8.11±0.000	0.547±0.010	0.560±0.000	0.145±0.002	0.705±0.000	0.355±0.013	2.02±0.000	3.30±0.201
DeepSTPP	6.51±0.000	0.341±0.000	0.486±0.000	0.197±0.000	0.0312±0.000	0.234±0.000	1.38±0.000	1.46±0.000
DSTPP (ours)	**2.84±0.193**	**0.149±0.001**	**0.170±0.001**	**0.0243±0.000**	**0.00495±0.000**	**0.0301±0.002**	**0.136±0.013**	**0.0891±0.009**

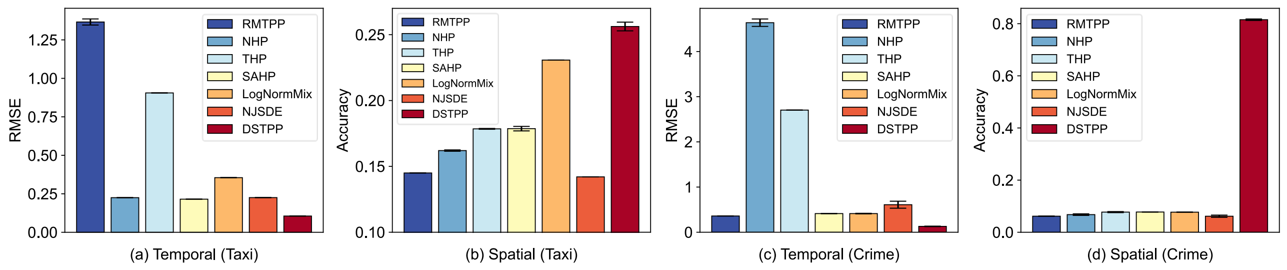


Figure 4: Prediction performance on discrete-space datasets for both time and space of the next event. The left column gives the temporal RMSE (lower is better), and the right column presents the spatial accuracy (higher is better).

4.3 Analysis of Spatio-temporal Interdependence (RQ2)

To gain a deeper understanding of the spatio-temporal interdependence in the denoising process, we perform an in-depth analysis of

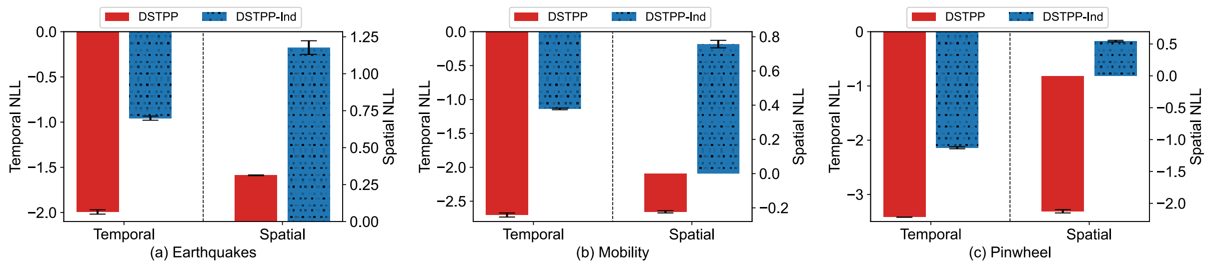


Figure 5: Ablation study on the joint spatio-temporal modeling. DSTPP-Ind denotes the degraded version of DSTPP, where spatial and temporal domains are independent.

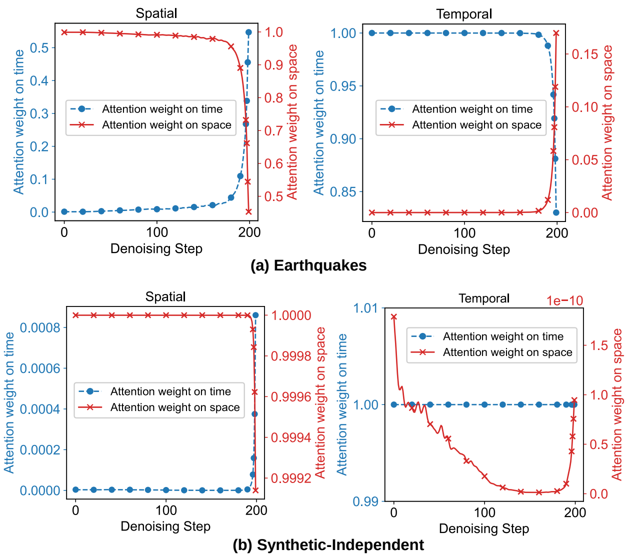


Figure 6: Spatial and temporal attention weights in the denoising iterations for two datasets with different spatio-temporal interdependence. Best viewed in color.

co-attention weights. Specifically, the analysis is conducted on two representative datasets: Earthquake and Synthetic-Independent, where the Earthquake dataset is highly spatio-temporal entangled, and the Synthetic-Independent dataset is totally spatio-temporal independent. Appendix A provides the generation details of the synthetic dataset. We use these two datasets to validate whether the designed co-attention mechanism can learn different interdependence between time and space. At each step of the denoising process, we calculate attention weights of the temporal and spatial dimensions on themselves and each other. Figure 6 shows how attention weights change as denoising proceeds.

As shown in Figure 6(a), at the early stage, temporal and spatial domains do not assign attention weights to each other, and the attention weights on themselves are close to one. At the final stage (step ≥ 150), the two domains start to assign attention weights to each other. At last, for the temporal domain, the attention weights on time and space are approximately 0.83 and 0.17; for the spatial domain, the attention weights are close to evenly divided (0.52 and 0.48), suggesting that the spatial domain is more dependent on the temporal domain. In the later stage of the denoising iterations, the model learns a distribution closer to the real case, thus, it is reasonable that the spatial and temporal domains assign more attention weights to each other. Figure 6(b) displays different results: the two domains share almost no attention weights to each other, indicating that the model has successfully learned the independent relationship. Figure 6(a) and (b) together validate the effectiveness of the co-attention mechanism, which can adaptively learn various interaction mechanisms between time and space.

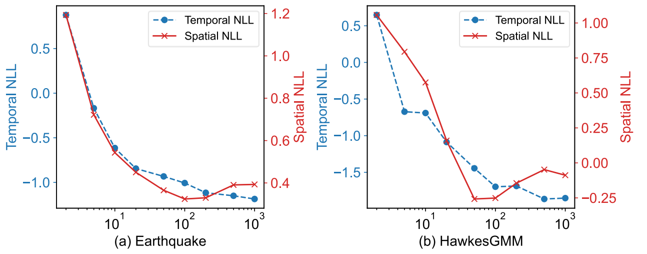


Figure 7: Ablation studies on the total number of diffusion steps for Earthquake and HawkesGMM data. We observe similar results with other datasets.

4.4 Ablation Studies (RQ2, RQ3)

Co-attention Mechanism. In order to examine the effectiveness of the co-attention mechanism, we degrade our DSTPP into a base framework, DSTPP-Ind, which models the distributions of space and time independently in the denoising process. To be specific, we replace $p_\theta(t_i^{k-1}|t_i^k, s_i^k, h_{i-1})$ and $p_\theta(s_i^{k-1}|t_i^k, s_i^k, h_{i-1})$ in Equation (10) with $p_\theta(t_i^{k-1}|t_i^k, h_{i-1})$, $p_\theta(s_i^{k-1}|s_i^k, h_{i-1})$, where space and time are not conditionally dependent on each other. Figure 5 shows the performance comparison of DSTPP and DSTPP-Ind in continuous-space settings. We can observe that DSTPP trained by incorporating the joint modeling of time and space performs consistently better than DSTPP-Ind with independent modeling. These results indicate the necessity to capture the interdependence between time and space, and meanwhile, validate the effectiveness of the spatio-temporal co-attention design.

Diffusion Steps. The number of total steps K in the diffusion process is a crucial hyperparameter. With the increase of diffusion steps, the denoising network approximates more minimal changes between steps. A bigger K allows the reverse denoising process to be adequately approximated by Gaussian distribution [47]. However, too many diffusion steps will vastly reduce the efficiency of training and sampling. Therefore, it is essential to explore to what extent, a larger diffusion step K improves the model's performance. Specifically, we perform ablation studies on

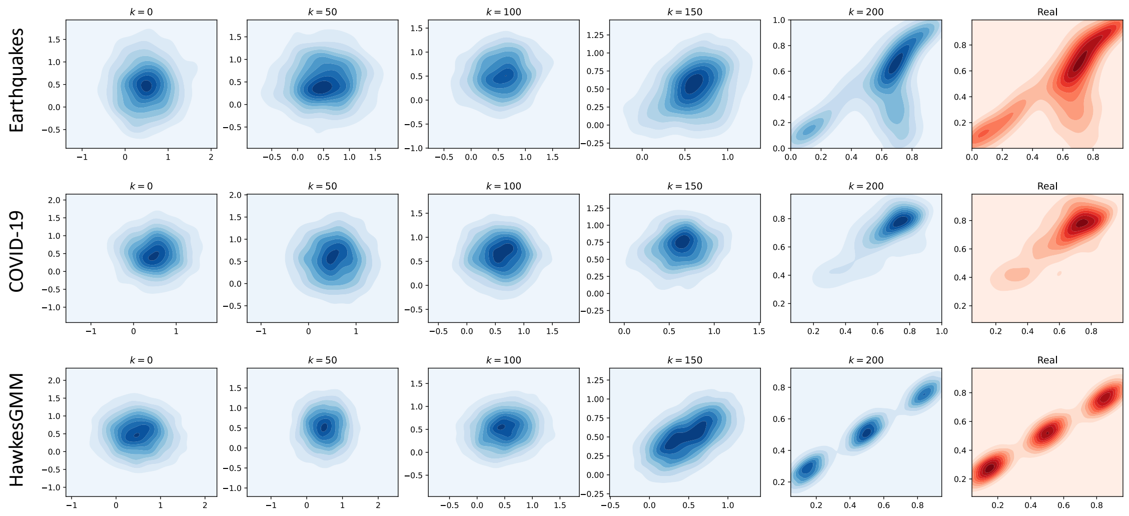


Figure 8: Visualization of the spatial distribution at different stages in the denoising process (the first five columns in blue color). The last column in red color presents the real distribution. Starting from Gaussian noise, our DSTPP model gradually fits the spatial distribution of ground truth. Best viewed in color.

Earthquakes and HawkesGMM datasets with varying total diffusion steps $K = \{2, 5, 10, 20, 50, 100, 200, 500, 1000\}$ and keep all other hyperparameters fixed. The results of temporal NLL and spatial NLL are plotted in Figure 7. We can observe that temporal NLL and spatial NLL both achieve the best values at $K \approx 200$, suggesting that the diffusion step K can be reduced to 200 without significant performance loss.

4.5 Analysis of Reverse Diffusion Process (RQ4)

To gain a deeper understanding of the denoising process, We visualize the spatial distribution during the reverse denoising iterations in Figure 8. As we can observe, at the beginning of the denoising process, the spatial distribution displays a Gaussian noise. With progressive denoising iterations, the data distribution deforms gradually and becomes more concentrated. Finally, at the last step, the spatial distribution fits perfectly with the ground truth distribution. It indicates that our DSTPP is able to learn the generative process of spatial distribution successfully. Besides, the denoising process is not a linear change, where the distribution changes during the last 50 steps are more significant than the previous steps. Combined with results in Section 4.3, where the interdependence between spatial and temporal domains is effectively captured in the latter stage, it is reasonable that the denoising effect is improved significantly during this period.

5 RELATED WORK

5.1 Spatio-temporal Point Processes

Temporal point process models [9, 27, 30, 59, 64] can be directly used for STPPs, where the space is considered as the event marker. Kernel density estimation methods are also used to model continuous-space distributions in STPP models [2, 3, 22, 33, 43, 61]. Most existing solutions follow an intensity-based paradigm, and their main challenge is how to choose a good parametric form for the intensity function. There exists a trade-off between the modeling capability of the intensity function and the cost to compute the log-likelihood. Some intensity-free models [36, 45] are proposed to tackle this problem, however, the probability density function either is unavailable [36] or still has certain model restrictions [45]. Another drawback of existing models is that they can only model either the continuous-space domain or the discrete-space domain, which largely limits their usability in real-world scenarios.

Recently, a line of advances have been developed for the generative modeling of point processes. For example, generative adversarial networks [8, 55] are used to learn to generate point processes in a likelihood-free manner. Reinforcement learning [25, 50] approaches and variational autoencoders [29, 37] are also included to explore the generative performance of TPPs. Some works also use noise contrastive learning [16, 31] instead of MLE. We are the first to learn point processes within the paradigm of diffusion models, which successfully address limitations in previous existing solutions.

5.2 Denoising Diffusion Probabilistic Models

Denoising diffusion probabilistic models (DDPM) [19, 47, 48], are a class of deep generative models, which are inspired by non-equilibrium thermodynamics. Due to their powerful generative capabilities, diffusion models have been used in a wide range of applications including image generation [1, 6, 44, 46], time series prediction and imputation [42, 49], audio generation [11, 20, 24], text generation [10, 12, 26], 3D point cloud generation [28, 60], and trajectory generation [15, 39]. In this paper, we first introduce the diffusion model to the domain of spatio-temporal point processes.

6 CONCLUSION

In this paper, we propose a novel framework to directly learn spatio-temporal joint distributions with no requirement for independence assumption and Monte Carlo sampling, which has addressed the structural shortcomings of existing solutions. The proposed framework also poses desired properties like easy training and closed-form sampling. Extensive experiments on diverse datasets with multiple evaluation metrics highlight the impact of our framework against state-of-the-art STPP models. As for future work, it is promising to apply our model in large-scale natural systems, such as climate changes and ocean currents, which are concerned with highly complex spatio-temporal data.

REFERENCES

[1] Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. 2021. Structured denoising diffusion models in discrete state-spaces. *Advances in Neural Information Processing Systems* 34 (2021), 17981–17993.

[2] Adrian Baddeley, Imre Bárány, and Rolf Schneider. 2007. Spatial point processes and their applications. *Stochastic Geometry: Lectures Given at the CIME Summer School Held in Martina Franca, Italy, September 13–18, 2004* (2007), 1–75.

[3] Ricky TQ Chen, Brandon Amos, and Maximilian Nickel. 2021. Neural spatio-temporal point processes. (2021).

[4] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. 2018. Neural ordinary differential equations. *Advances in neural information processing systems* 31 (2018).

[5] Daryl J Daley, David Vere-Jones, et al. 2003. *An introduction to the theory of point processes: volume I: elementary theory and methods*. Springer.

[6] Prafulla Dhariwal and Alexander Nichol. 2021. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems* 34 (2021), 8780–8794.

[7] Peter J Diggle. 2006. Spatio-temporal point processes: methods and applications. *Monographs on Statistics and Applied Probability* 107 (2006), 1.

[8] S Haleh S Dizaji, Saeid Pashazadeh, and Javad Musevi Niya. 2022. Wasserstein generative adversarial networks for modeling marked events. *The Journal of Supercomputing* (2022), 1–23.

[9] Nan Du, Hanjun Dai, Rakshit Trivedi, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and Le Song. 2016. 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*. 1555–1564.

[10] Zhujin Gao, Junliang Guo, Xu Tan, Yongxin Zhu, Fang Zhang, Jiang Bian, and Linli Xu. 2022. Difformer: Empowering Diffusion Model on Embedding Space for Text Generation. *arXiv preprint arXiv:2212.09412* (2022).

[11] Karan Goel, Albert Gu, Chris Donahue, and Christopher Ré. 2022. It's raw! audio generation with state-space models. In *International Conference on Machine Learning*. PMLR, 7616–7633.

[12] Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and LingPeng Kong. 2022. Diffuseq: Sequence to sequence text generation with diffusion models. *arXiv preprint arXiv:2210.08933* (2022).

[13] Jonatan A González, Francisco J Rodríguez-Cortés, Ottmar Cronie, and Jorge Mateu. 2016. Spatio-temporal point process statistics: a review. *Spatial Statistics* 18 (2016), 505–544.

[14] Jan Grandell. 2012. *Aspects of risk theory*. Springer Science & Business Media.

[15] Tianpei Gu, Guangyi Chen, Junlong Li, Chunze Lin, Yongming Rao, Jie Zhou, and Jiwen Lu. 2022. Stochastic trajectory prediction via motion indeterminacy diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 17113–17122.

[16] Ruocheng Guo, Jundong Li, and Huan Liu. 2018. INITIATOR: Noise-contrastive Estimation for Marked Temporal Point Process.. In *IJCAI*. 2191–2197.

[17] Alan G Hawkes. 1971. Point spectra of some mutually exciting point processes. *Journal of the Royal Statistical Society: Series B (Methodological)* 33, 3 (1971), 438–443.

[18] Alan G Hawkes. 2018. Hawkes processes and their applications to finance: a review. *Quantitative Finance* 18, 2 (2018), 193–198.

[19] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems* 33 (2020), 6840–6851.

[20] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. 2022. Video diffusion models. *arXiv preprint arXiv:2204.03458* (2022).

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

[22] Junteng Jia and Austin R Benson. 2019. Neural jump stochastic differential equations. *Advances in Neural Information Processing Systems* 32 (2019).

[23] John Frank Charles Kingman. 1992. *Poisson processes*. Vol. 3. Clarendon Press.

[24] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. 2020. Diffwave: A versatile diffusion model for audio synthesis. *arXiv preprint arXiv:2009.09761* (2020).

[25] Shuang Li, Shuai Xiao, Shixiang Zhu, Nan Du, Yao Xie, and Le Song. 2018. Learning temporal point processes via reinforcement learning. *Advances in neural information processing systems* 31 (2018).

[26] Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori B Hashimoto. 2022. Diffusion-lm improves controllable text generation. *arXiv preprint arXiv:2205.14217* (2022).

[27] Haitao Lin, Lirong Wu, Guojiang Zhao, Pai Liu, and Stan Z Li. 2022. Exploring Generative Neural Temporal Point Process. *Transactions on Machine Learning Research* (2022).

[28] Shitong Luo and Wei Hu. 2021. Diffusion probabilistic models for 3d point cloud generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2837–2845.

[29] Nazanin Mehrasa, Akash Abdu Jyothi, Thibaut Durand, Jiawei He, Leonid Sigal, and Greg Mori. 2019. A variational auto-encoder model for stochastic point processes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 3165–3174.

[30] Hongyuan Mei and Jason M Eisner. 2017. The neural hawkes process: A neurally self-modulating multivariate point process. *Advances in neural information processing systems* 30 (2017).

[31] Hongyuan Mei, Tom Wan, and Jason Eisner. 2020. Noise-contrastive estimation for multivariate point processes. *Advances in neural information processing systems* 33 (2020), 5204–5214.

[32] Sebastian Meyer, Johannes Elias, and Michael Höhle. 2012. A space–time conditional intensity model for invasive meningococcal disease occurrence. *Biometrics* 68, 2 (2012), 607–616.

[33] Jesper Moller and Rasmus Plenge Waagepetersen. 2003. *Statistical inference and simulation for spatial point processes*. CRC press.

[34] Yosihiko Ogata. 1988. Statistical models for earthquake occurrences and residual analysis for point processes. *Journal of the American Statistical association* 83, 401 (1988), 9–27.

[35] Maya Okawa, Tomoharu Iwata, Takeshi Kurashima, Yusuke Tanaka, Hiroyuki Toda, and Naonori Ueda. 2019. 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*. 373–383.

[36] Takahiro Omi, Kazuyuki Aihara, et al. 2019. Fully neural network based model for general temporal point processes. *Advances in neural information processing systems* 32 (2019).

[37] Zhen Pan, Zhenya Huang, Defu Lian, and Enhong Chen. 2020. A variational point process model for social event sequences. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 173–180.

[38] Junhyung Park, Adam W Chaffee, Ryan J Harrigan, and Frederic Paik Schoenberg. 2022. A non-parametric hawkes model of the spread of ebola in west africa. *Journal of Applied Statistics* 49, 3 (2022), 621–637.

[39] Jami Pekkanen, Oscar Terence Giles, Yee Mun Lee, Ruth Madigan, Tatsuru Daimon, Natasha Merat, and Gustav Markkula. 2021. Variable-drift diffusion models of pedestrian road-crossing decisions. *Computational Brain & Behavior* (2021), 1–21.

[40] Rüdiger Rackwitz and Bernd Flessler. 1978. Structural reliability under combined random load sequences. *Computers & structures* 9, 5 (1978), 489–494.

[41] Jakob Gulddahl Rasmussen. 2018. Lecture notes: Temporal point processes and the conditional intensity function. *arXiv preprint arXiv:1806.00221* (2018).

[42] Kashif Rasul, Calvin Seward, Ingmar Schuster, and Roland Vollgraf. 2021. Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. In *International Conference on Machine Learning*. PMLR, 8857–8868.

[43] Alex Reinhart. 2018. A review of self-exciting spatio-temporal point processes and their applications. *Statist. Sci.* 33, 3 (2018), 299–318.

[44] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10684–10695.

[45] Oleksandr Shchur, Marin Biloš, and Stephan Günnemann. 2019. Intensity-free learning of temporal point processes. (2019).

[46] Abhishek Sinha, Jiaming Song, Chenlin Meng, and Stefano Ermon. 2021. D2c: Diffusion-decoding models for few-shot conditional generation. *Advances in Neural Information Processing Systems* 34 (2021), 12533–12548.

[47] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. 2015. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*. PMLR, 2256–2265.

[48] Jiaming Song, Chenlin Meng, and Stefano Ermon. 2020. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502* (2020).

[49] Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. 2021. CSDI: Conditional score-based diffusion models for probabilistic time series imputation. *Advances in Neural Information Processing Systems* 34 (2021), 24804–24816.

[50] Utkarsh Upadhyay, Abir De, and Manuel Gomez Rodriguez. 2018. Deep reinforcement learning of marked temporal point processes. *Advances in Neural Information Processing Systems* 31 (2018).

[51] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[52] Huandong Wang, Qiaohong Yu, Yu Liu, Depeng Jin, and Yong Li. 2021. Spatio-temporal urban knowledge graph enabled mobility prediction. *Proceedings of the ACM on interactive, mobile, wearable and ubiquitous technologies* 5, 4 (2021), 1–24.

[53] Qianlong Wang, Yifan Guo, Lixing Yu, and Pan Li. 2017. Earthquake prediction based on spatio-temporal data mining: an LSTM network approach. *IEEE Transactions on Emerging Topics in Computing* 8, 1 (2017), 148–158.

[54] Stanisław Węglarczyk. 2018. Kernel density estimation and its application. In *ITM Web of Conferences*, Vol. 23. EDP Sciences, 00037.

[55] Shuai Xiao, Mehrdad Farajtabar, Xiaojing Ye, Junchi Yan, Le Song, and Hongyuan Zha. 2017. Wasserstein learning of deep generative point process models. *Advances in neural information processing systems* 30 (2017).

[56] Zheng Xu, Yunhuai Liu, Neil Y Yen, Lin Mei, Xiangfeng Luo, Xiao Wei, and Chuanping Hu. 2016. Crowdsourcing based description of urban emergency events using social media big data. *IEEE Transactions on Cloud Computing* 8, 2 (2016), 387–397.

[57] Dingqi Yang, Daqing Zhang, Vincent W Zheng, and Zhiyong Yu. 2014. Modeling user activity preference by leveraging user spatial temporal characteristics in LBSNs. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 45, 1 (2014), 129–142.

[58] Guolei Yang, Ying Cai, and Chandan K Reddy. 2018. Recurrent spatio-temporal point process for check-in time prediction. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management.* 2203–2211.

[59] Qiang Zhang, Aldo Lipani, Omer Kirnap, and Emine Yilmaz. 2020. Self-attentive Hawkes process. In *International conference on machine learning.* PMLR, 11183–11193.

[60] Linqi Zhou, Yilun Du, and Jiajun Wu. 2021. 3d shape generation and completion through point-voxel diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision.* 5826–5835.

[61] Zihao Zhou, Xingyi Yang, Ryan Rossi, Handong Zhao, and Rose Yu. 2022. Neural Point Process for Learning Spatiotemporal Event Dynamics. In *Learning for Dynamics and Control Conference.* PMLR, 777–789.

[62] Shixiang Zhu, Ruyi Ding, Minghe Zhang, Pascal Van Hentenryck, and Yao Xie. 2021. Spatio-temporal point processes with attention for traffic congestion event modeling. *IEEE Transactions on Intelligent Transportation Systems* 23, 7 (2021), 7298–7309.

[63] Jiancang Zhuang. 2006. Second-order residual analysis of spatiotemporal point processes and applications in model evaluation. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 68, 4 (2006), 635–653.

[64] Simiao Zuo, Haoming Jiang, Zichong Li, Tuo Zhao, and Hongyuan Zha. 2020. Transformer hawkes process. In *International conference on machine learning.* PMLR, 11692–11702.

APPENDIX

A DATASET

Earthquakes: Earthquakes in Japan from 1990 to 2020 with a magnitude of at least 2.5 are collected from the U.S. Geological Survey[7]. Sequences are generated by sliding windows with a window size of 30 days and a gap of seven days. Therefore, each sequence is within the length of 30 days. The earthquakes from Nov. 2010 to Dec. 2011 are removed because they are outliers compared with data in other periods. We split the dataset into the training set, validation set, and testing set and ensure that there is no overlap between them. Finally, We have 950 sequences for the training set, 50 for the validation set, and 50 for the testing set. The sequence lengths range between 22 to 554.

COVID-19: We construct this dataset by using publicly released COVID19 cases by The New York Times (2020). It records daily infected cases of COVID-19 in New Jersey state[8] from March 2020 to July 2020. We aggregate the data at the county level. Sequences are generated by sliding windows with a window size of 7 days and a gap of three days. Therefore, each sequence is within the length of 7 days. We split the dataset into the training set, validation set, and testing set and ensure that there is no overlap between them. Finally, We have 1450 sequences for the training set, 100 for the validation set, and 100 for the testing set. The sequence lengths range between 5 to 287.

Mobility [57]: This dataset consisting of Check-in behaviors in Tokyo is collected by Foursquare for two months. Each check-in is associated with timestamps and GPS coordinates. Each sequence contains all check-in behaviors of one user in the two months. We randomly split the dataset into the training set, validation set, and testing set. Finally, We have 1200 sequences for the training set, 300 for the validation set, and 300 for the testing set. The sequence lengths range between 5 to 131.

Citibike: This dataset is collected by a bike-sharing service, which records the demand for bike sharing in New York City. We use the records from April 2019 to August 2019. The start of each trip is considered as an event. We split the record sequence of each bike into one-day subsequences starting at 5:00 am in the day. Therefore, each sequence is within the length of 19 hours. We randomly split the dataset into the training set, validation set, and testing set. Finally, We have 2440 sequences for the training set, 300 for the validation set, and 320 for the testing set. The sequence lengths range between 14 to 204.

Crime [9]: It is provided by the Atlanta Police Department, recording robbery crime events from the end of 2015 to 2017. Each robbery report is associated with the time and the neighborhood. Each sequence is within the length of one day. We randomly split the dataset into the training set, validation set, and testing set. Finally, We have 2000 sequences for the training set, 200 for the validation set, and 2000 for the testing set. The sequence lengths range between 26 to 144.

Taxi[10]: This dataset consists of ~173 million pick-up records of taxis in New York City. We use data from 1st, April 2013 to 14th, April 2013, with a period of two weeks. Each pick-up is associated with the time and the neighborhood. This resulted in 2400 sequences in total. We randomly split the dataset into the training set, validation set, and testing set. Finally, We have 2000 sequences for the training set, 200 for the validation set, and 200 for the testing set. The sequence lengths range between 151 to 291.

Synthetic-Independent: The temporal domain is generated by a Hawkes process, and the intensity function is defined as follows:

$$\lambda(t, |H_t) = 0.2 + \sum_{t<t_i} (0.2e^{-0.2(t-t-t_i)} + 4e^{-10(t_i-t)}) \tag{18}$$

the spatial distribution follows a Two-dimensional Gaussian distribution:

$$f(s1, s2) = \tag{19}$$

$$\frac{1}{2\pi\sigma_{s1}\sigma_{s2}\sqrt{(1-\rho^2)}} e^{-\frac{1}{2(1-\rho^2)}[(\frac{s1-\mu_1}{\sigma_{s1}})^2 - 2\rho(\frac{s1-\mu_1}{\sigma_{s1}})(\frac{s2-\mu_2}{\sigma_{s2}}) + (\frac{s2-\mu_2}{\sigma_{s2}})^2]} \tag{20}$$

where $\rho = \frac{\sqrt{2}}{4}, \mu_1 = 4.0, \mu_2 = 7.0, \sigma_{s1} = \sqrt{2}, \sigma_{s2} = 2$.

The lengths of event sequences in each dataset have large variations. Figure 9 illustrates the distribution of sequence lengths for each dataset.

B BASELINE

We provide detailed descriptions of used baselines in our work as follows:

- Conditional KDE: Conditional kernel density estimations. We utilize a history-dependent Gaussian mixture model to model the spatial distribution.

[7] https://earthquake.usgs.gov/earthquakes/search/
[8] https://github.com/nytimes/covid-19-data

[9] http://www.atlantapd.org/i-want-to/crime-data-downloads
[10] http://www.andresmh.com/nyctaxitrips/

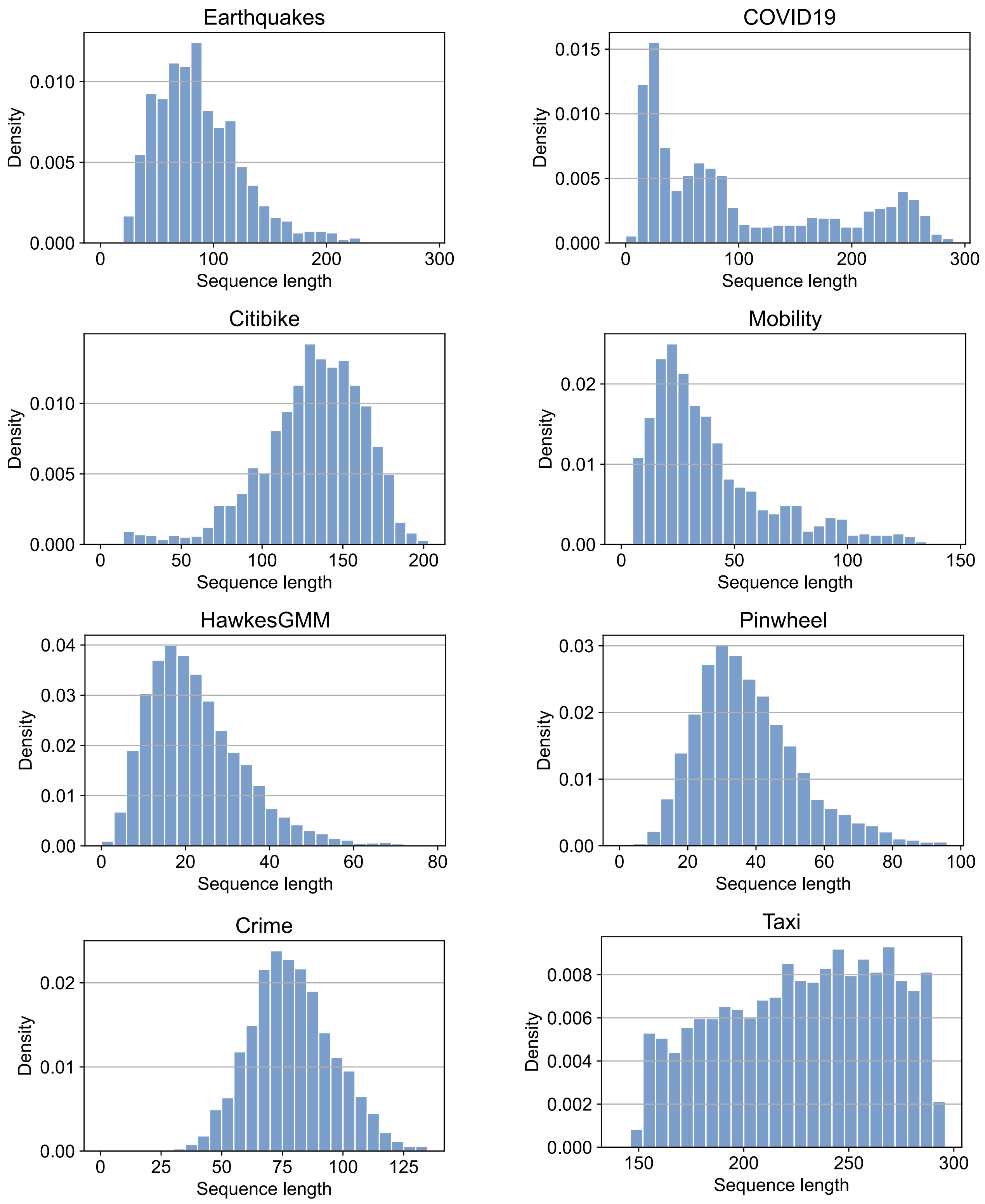


Figure 9: Distribution of the sequence length in each dataset.

- CNF and Time-varying CNF [4]: We use Continuous normalizing flow for modeling spatial distribution. Time-varying CNF denotes the dependence on the timestamps.
- Possion [41]: The homogeneous Poisson process is the simplest point process, where the number of events occurring during time intervals are independent, and the probability of a single event occurrence is proportional to the length of the interval.
- Hawkes [18]: Its essential property is that the occurrence of any event increases the probability of further events occurring by a certain amount. The triggering kernel which captures temporal dependencies can be chosen in advance or directly learned from data.
- Self-correcting [21]: In contrast to the Hawkes process, this point process follows the pattern that the occurrence of past events inhibits the occurrence of future events. Every time when a new event appears, the intensity is decreased by multiplying a constant less than 1.
- Recurrent Marked Temporal Point Process (RMTPP) [9]: This neural temporal point process model applies a nonlinear function of past events to model the intensity function and leverages RNNs to learn a representation of the influence from event history, where time intervals act as explicit inputs.
- Neural Hawkes Process (NHP) [30]: With the goal of capturing the temporal evolution of event sequences, it uses continuous-time LSTMs to model a marked TPP. The modeling of future event intensities is conditioned on the RNN's hidden state.
- Transformer Hawkes Process (THP) [64]: It is an extension to the transformer by modeling the conditional intensity. The self-attention mechanism is leveraged to capture long-term dependencies.
- Self-attentive Hawkes Process (SAHP) [59]: It learns the temporal dynamics by leveraging a self-attention mechanism to aggregate historical events. In order to take the time intervals between events into consideration, it modifies the conventional positional encoding by converting time intervals into phase shifts of sinusoidal functions.
- Log Normal Mixture model (LogNormMix) [45]: It adopts intensity-free learning of TPPs, which models the PDF by a log-normal mixture model. Additionally, a simple mixture model is proposed to match the flexibility of flow-based models. The loglikelihood for training and density for sampling are in closed form.
- Wasserstein GAN (WGAN) [55]: This intensity-free approach transforms nuisance processes to a target one. And the Wasserstein distance is used to train the model, which is a likelihood-free method. Loglikelihood cannot be obtained for this approach.
- Neural Jump Stochastic Differential Equations (NJSDE) [45]: It models TPPs with a piecewise-continuous latent representation, where the discontinuities are brought by stochastic events. The spatial distribution is modeled with a Gaussian mixture model.
- Neural Spatio-temporal Point Process (NSTPP) [3]: It applies Neural ODEs as the backbone, which parameterizes the temporal intensity with Neural Jump SDEs and the spatial PDF with continuous-time normalizing flows.
- Deep Spatiotemporal Point Process (DeepSTPP) [61]: It is the state-of-the-art STPP model, which suggests using a nonparametric space-time intensity function governed by a latent process. Amortized variational inference is leveraged to deduce the latent process.

C IMPLEMENTATION DETAILS

C.1 Evaluation Metrics

Suppose $\boldsymbol{y} = y_1, ..., y_M$ represents the ground truth for real values, $\hat{\boldsymbol{y}} = \hat{y}_1, ..., \hat{y}_N$ represents the predicted real values, $\boldsymbol{k} = k_1, ..., k_M$ represents the ground truth for real values, $\hat{\boldsymbol{k}} = \hat{k}_1, ..., \hat{k}_N$ represents the predicted discrete labels, and N denotes the number of test samples, we can formulate these metrics as follows:

$$\begin{aligned} \text{RMSE}(\boldsymbol{y}, \hat{\boldsymbol{y}}) &= \sqrt{\frac{1}{N} \sum_{i}^{N} (y_i - \hat{y}_i)^2}, \\ \text{d}_{\text{Euclid}}(\boldsymbol{y}, \hat{\boldsymbol{y}}) &= \frac{1}{N} \sum_{i=1}^{N} \|y_i - \hat{y}_i\| \\ \text{Accuracy}(\boldsymbol{k}, \hat{\boldsymbol{k}}) &= \frac{TP + TN}{TP + TN + FP + FN} \end{aligned} \tag{21}$$

where d_{Euclid} denotes the Euclidean distance between two real-valued vectors. TN, TP, FN, and FP are the number of true negative samples, the number of true positive samples, the number of false negative samples, and the number of false positive samples.

C.2 Parameter Settings

In our experiments, we all use three-layer MLPs with ReLU activations and hidden size of 64. The training process is performed in batch. After training the model for ten epochs (all training instance), we examine the model's performance on validation set. The model that delivers the best performance on the validation set will be used to validate the performance on the test set. We set the learning rate as 3e-4 via searching in a set of $\{1e-4, 3e-4, 1e-3\}$. The proposed framework is implemented with Pytorch. We train it on a Linux server with eight GPUs (NVIDIA RTX 2080 Ti * 8). In practice, our framework can be effectively trained within 6 hours on a single GPU. As for some baselines that use CPUs, the models are trained on a Linux server with two CPUs (Intel Xeon E5-2650 * 2).

D ADDITIONAL RESULTS AND FIGURES

Table 6 and 7 show the overall performance of our framework and baseline methods for the remaining datasets. Figure 10 visualizes the denoising process for the remaining datasets.

Table 4: Performance evaluation for negative log-likelihood per event on test data. ↓ means lower is better. Bold denotes the best results and underline denotes the second-best results.

	Mobility		Pinwheel	
Model	Spatial	Temporal	Spatial	Temporal
Conditional KDE	0.638±0.021	-	2.74±0.005	-
CNF	1.96±0.058	-	2.19±0.002	-
TVCNF	2.45±0.061	-	2.24±0.009	-
Possion	-	1.63±0.002	-	0.778±0.000
Hawkes	-	0.854±0.000	-	0.265±0.000
Self-correcting	-	15.4±0.299	-	2.46±0.320
RMTPP	-	1.63±0.001	-	0.974±0.002
NHP	-	0.121±0.006	-	-0.108±0.001
THP	-	-0.500±0.016	-	-1.01±0.001
SAHP	-	-0.111±0.023	-	-1.60±0.083
LogNormMix	-	-0.0566±0.018	-	-0.0680±0.004
NJSDE	1.54±0.016	1.52±0.078	2.08±0.026	1.16±0.020
NSTPP	1.28±0.035	-0.326±0.004	1.64±0.012	-0.0171±0.003
DeepSTPP	**-5.17±0.010**	0.573±0.003	3.36±0.005	0.0590±0.002
DSTPP (ours)	-0.226±0.017	**-2.69±0.060**	**-2.10±0.045**	**-3.42±0.003**

Table 5: Performance evaluation for predicting both time and space of the next event. We use Euclidean distance to measure the prediction error of the spatial domain and use RMSE between real intervals and predicted intervals for the temporal domain.

	Mobility		Pinwheel	
Model	Spatial	Temporal	Spatial	Temporal
Conditional KDE	0.0720±0.001	-	0.718±0.001	-
CNF	8.67±0.195	-	2.74±0.020	-
TVCNF	5.04±0.005	-	1.97±0.000	-
Possion	-	4.08±0.021	-	3.23±1.565
Hawkes	-	2.07±1.215	-	7.59±0.000
Self-correcting	-	36.3±0.311	-	395±5.292
RMTPP	-	5.17±0.001	-	2.07±0.004
NHP	-	3.36±0.040	-	2.23±0.017
THP	-	7.27±0.034	-	4.27±0.003
SAHP	-	3.36±0.064	-	1.90±0.034
LogNormMix	-	3.78±0.115	-	1.92±0.023
WGAN	-	3.78±0.058	-	1.87±0.025
NJSDE	0.0980±0.001	2.60±0.072	1.88±0.002	1.50±0.002
NSTPP	0.0413±0.000	4.16±0.134	1.97±0.000	1.97±0.048
DeepSTPP	0.0487±0.000	2.16±0.001	1.77±0.020	1.26±0.000
DSTPP (ours)	**0.0342±0.000**	**1.22±0.041**	**0.275±0.002**	**0.0444±0.002**

Table 6: Performance evaluation for negative log-likelihood per event on test data. ↓ means lower is better. Bold denotes the best results and underline denotes the second-best results.

	Mobility		Citibike	
Model	Spatial NLL	Temporal NLL	Spatial NLL	Temporal NLL
Conditional KDE	0.638±0.021	-	2.74±0.005	-
CNF	1.96±0.058	-	2.19±0.002	-
TVCNF	2.45±0.061	-	2.24±0.009	-
Possion	-	1.63±0.002	-	0.778±0.000
Hawkes	-	0.854±0.000	-	0.265±0.000
Self-correcting	-	15.4±0.299	-	2.46±0.320
RMTPP	-	1.63±0.001	-	0.974±0.002
NHP	-	0.121±0.006	-	-0.108±0.001
THP	-	<u>-0.500±0.016</u>	-	-1.01±0.001
SAHP	-	-0.111±0.023	-	<u>-1.60±0.083</u>
LogNormMix	-	-0.0566±0.018	-	-0.0680±0.004
NJSDE	1.54±0.016	1.52±0.078	2.08±0.026	1.16±0.020
NSTPP	1.28±0.035	-0.326±0.004	<u>1.64±0.012</u>	-0.0171±0.003
DeepSTPP	**-5.17±0.010**	0.573±0.003	3.36±0.005	0.0590±0.002
DSTPP (ours)	<u>-0.226±0.017</u>	**-2.69±0.060**	**-2.10±0.045**	**-3.42±0.003**

Table 7: Performance evaluation for predicting both time and space of the next event. We use Euclidean distance to measure the prediction error of the spatial domain and use RMSE between real intervals and predicted intervals for the temporal domain.

	Mobility		Citibike	
Model	Distance	RMSE	Distance	RMSE
Conditional KDE	0.0720±0.001	-	<u>0.718±0.001</u>	-
CNF	8.67±0.195	-	2.74±0.020	-
TVCNF	5.04±0.005	-	1.97±0.000	-
Possion	-	4.08±0.021	-	3.23±1.565
Hawkes	-	<u>2.07±1.215</u>	-	7.59±0.000
Self-correcting	-	36.3±0.311	-	395±5.292
RMTPP	-	5.17±0.001	-	2.07±0.004
NHP	-	3.36±0.040	-	2.23±0.017
THP	-	7.27±0.034	-	4.27±0.003
SAHP	-	3.36±0.064	-	1.90±0.034
LogNormMix	-	3.78±0.115	-	1.92±0.023
NJSDE	0.0980±0.001	2.60±0.072	1.88±0.002	1.50±0.002
NSTPP	<u>0.0413±0.000</u>	4.16±0.134	1.97±0.000	1.97±0.048
DeepSTPP	0.0487±0.000	2.16±0.001	1.77±0.020	<u>1.26±0.000</u>
DSTPP (ours)	**0.0342±0.000**	**1.22±0.041**	**0.275±0.002**	**0.0444±0.002**

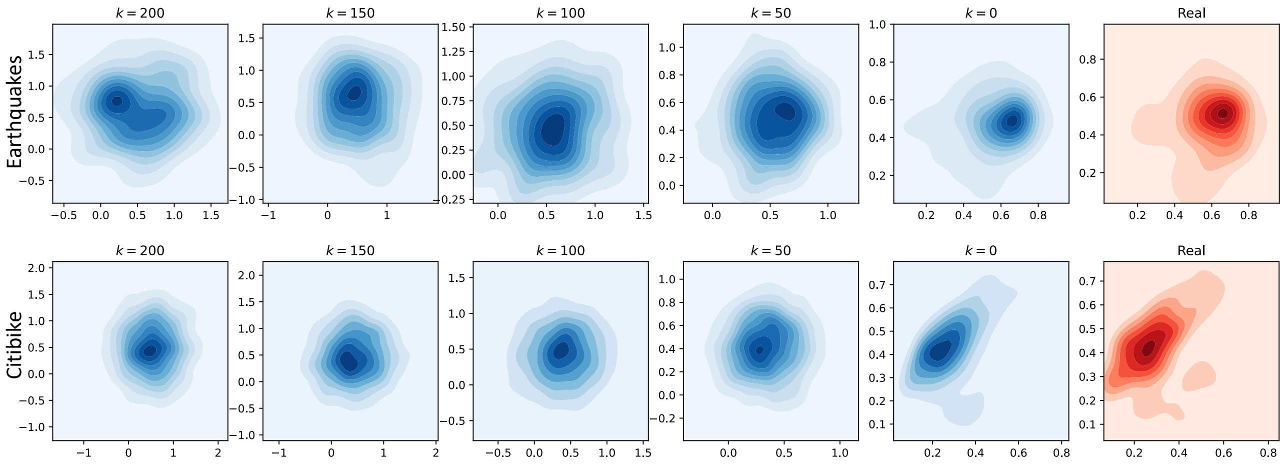


Figure 10: Visualization of the spatial distribution at different stages in the denoising process (the first five columns in blue color). The last column in red color presents the real distribution. Starting from Gaussian noise, our DSTPP model gradually fits the spatial distribution of ground truth. Best viewed in color.