Beyond Point Prediction: Score Matching-based Pseudolikelihood Estimation of Neural Marked Spatio-Temporal Point Process

Zichong Li [1] **Qunzhi Xu** [1] **Zhenghao Xu** [1] **Yajun Mei** [1] **Tuo Zhao** [1] **Hongyuan Zha** [2]

Abstract

Spatio-temporal point processes (STPPs) are potent mathematical tools for modeling and predicting events with both temporal and spatial features. Despite their versatility, most existing methods for learning STPPs either assume a restricted form of the spatio-temporal distribution, or suffer from inaccurate approximations of the intractable integral in the likelihood training objective. These issues typically arise from the normalization term of the probability density function. Moreover, existing works only provide point prediction for events without quantifying their uncertainty, such as confidence intervals for the event's arrival time and confidence regions for the event's location, which is crucial given the considerable randomness of the data. To tackle these challenges, we introduce SMASH: a Score MAtching-based pSeudolikeliHood estimator for learning marked STPPs. Specifically, our framework adopts a normalization-free objective by estimating the pseudolikelihood of marked STPPs through score-matching and predicts confidence intervals/regions for event time and location by generating samples through a score-based sampling algorithm. The superior performance of our proposed framework is demonstrated through extensive experiments on both point and confidence interval/region prediction of events.

1. Introduction

Spatio-temporal point processes (STPPs) are stochastic processes that model event occurrences in time and space, where each event is associated with both temporal and spatial features. STPPs are widely used in various fields, including ecology (González et al., 2016), physiology (Tagliazucchi et al., 2012), and epidemiology (Li et al., 2021), to model complex event sequences such as earthquakes, brain activities, and disease outbreaks. Classical STPPs (Diggle, 2006; González et al., 2016) capture relatively simple spatio-temporal patterns through combining a temporal point process model, such as Poisson process (Kingman, 1992) and self-excitation process (Hawkes, 1971), with a pre-specified spatial distribution estimator. With the advent of neural networks, flexible and expressive neural STPPs have been developed to model much more complicated event dynamics, see Zhou et al. (2022); Dong et al. (2023); Yuan et al. (2023); Okawa et al. (2019); Zhu et al. (2022).

In neural STPPs, event distributions are typically characterized through intensity functions parametrized by the neural network, which model the influence of past events on present occurrences. Current methods predominantly estimate parameters by maximizing the log-likelihood of event time and location. However, computing such a likelihood involves integrating the intensity function over time and space, which is usually intractable due to the intricate form of the neural network. Therefore, they resort to numerical approximation methods such as Monte Carlo integration (González et al., 2016; Zhu et al., 2022; Dong et al., 2023), which inevitably introduce approximation errors that compromise prediction accuracy. Various works have sought solutions for this complication. For instance, Chen et al. (2021) utilize ODE with the continuous-time normalizing flow (CNFs) to model continuous transformation of the distribution, but they need to integrate over the ODE trajectory by inefficient numerical ODE solver. Jia and Benson (2019); Zhou et al. (2022) bypass the approximation of the integral by assuming restricted forms of the distribution such as Gaussian mixture models, but failing to capture complex spatio-temporal dynamics. Yuan et al. (2023) propose a diffusion-based STPP model to avoid integrals and flexibly learn event distributions. However, they suffer from inefficient training and non-trivial hyperparameter configuration.

Another limitation of neural STPPs is the absence of confidence interval prediction. Given the huge inherent randomness of event times and locations, the variance of data points

[1]Georgia Institute of Technology, Atlanta, USA [2]The Chinese University of Hong Kong, Hongkong, China. Correspondence to: Zichong Li <zichongli@gatech.edu>, Tuo Zhao <tourzhao@gatech.edu>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

typically far exceeds the mean, rendering point prediction unreliable and insufficient. Thus, it is crucial to quantify the uncertainty associated with event predictions by providing confidence intervals for event times and confidence regions for locations. Moreover, when dealing with marked STPPs, where discrete marks are associated with each event, we aim to match the predicted posterior distribution to the ground truth data. Since the learned model often exhibits overconfidence or underconfidence, capturing an accurate confidence score for event mark prediction is also necessary to enable users to make more informed decisions. Li et al. (2023) propose SMURF-THP for marked temporal point process and provides confidence interval prediction, which will be further discussed in Section 3.

In this work, we introduce SMASH: a Score MAtching-based pSeudolikeliHood estimator for learning marked STPPs, to address the above issues. Specifically, SMASH bypasses the difficulty in integral calculation by adopting a score-matching objective (Hyvärinen, 2005) for the conditional likelihood of event time and location, which matches the derivative of the log-likelihood (known as score) to the derivative of the log-density of the underlying (unknown) distribution. Furthermore, as a score-based generative model, SMASH can naturally generate samples from the learned score functions using score-based sampling. This facilitates predicting confidence intervals and regions for event time and location.

In summary, we make three primary contributions:

• We propose SMASH, a consistent estimator for marked STPPs that leverages a normalization-free score-based pseudolikelihood objective to bypass the intractable integral computation involved in log-likelihood estimation. SMASH parametrizes the conditional score function of event location and the joint intensity of event time and mark, capturing intricate spatio-temporal dynamics with discrete marks.

• We consider confidence interval/region prediction for events beyond unreliable point prediction, a feature unexplored and unevaluated by existing STPP methods. SMASH supports flexible generation of event time, mark and location via score-based sampling, offering high-quality samples.

• We validate the effectiveness of SMASH using multiple real-world datasets. Our results demonstrate that SMASH offers significant improvements over other baselines in both point and confidence interval/region prediction.

2. Background

We briefly review marked spatio-temporal point process (MSTPP), neural STPP, score matching, Langevin dynamics, and confidence interval prediction in this section.

• **Marked Spatio-Temporal Point Process** is a stochastic process whose realization consists of an ordered sequence of discrete events $S = \{(t_i, k_i, \boldsymbol{x}_i)\}_{i=1}^{L}$ with length L, where $t_i \in [0, T]$ is the time of occurrence, $k_i \in \{1, \cdots, M\}$ is the discrete event mark/type and $\boldsymbol{x}_i \in \mathbb{R}^d$ is the location of the event that has occurred at time t_i. Denote the history events up to time t as $\mathcal{H}_t = \{(t_j, k_j, \boldsymbol{x}_j) : t_j < t\}$, the events' distribution in MSTPPs is usually characterized via the *conditional intensity function* $\lambda(t, k, \boldsymbol{x} \mid \mathcal{H}_{t_i})$. For simplicity, we omit conditional dependency on the history in the following discussions and employ a superscript i to signify the condition on $\mathcal{H}_{t_i}$. The conditional intensity is then defined as

$$\begin{aligned}\lambda^i(t, k, \boldsymbol{x}) &\triangleq \lambda(t, k, \boldsymbol{x} \mid \mathcal{H}_{t_i}) \\ &= \lim_{\Delta t, \Delta \boldsymbol{x} \downarrow 0} \frac{\mathbb{P}^i(t_i \in [t, t+\Delta t], k_i = k, \boldsymbol{x}_i \in B(\boldsymbol{x}, \Delta \boldsymbol{x}))}{|B(\boldsymbol{x}, \Delta \boldsymbol{x})| \Delta t},\end{aligned}$$

where $B(\boldsymbol{x}, \Delta x)$ denotes a ball centered at $\boldsymbol{x}$ and with radius $\Delta \boldsymbol{x}$. The conditional intensity function describes the instantaneous probability that an event of mark k occurs at time t and location $\boldsymbol{x}$ given the events' history $\mathcal{H}_{t_i}$. The generalized conditional probability of the i-th event given history $\mathcal{H}_{t_i}$ can then be expressed as

$$p^i(t, k, \boldsymbol{x}) = \lambda^i(t, k, \boldsymbol{x}) e^{-\int_{t_{i-1}}^{t} \int_{\mathbb{R}^d} \sum_{l=1}^{M} \lambda^i(\tau, l, \boldsymbol{s}) d\tau d\boldsymbol{s}}, \quad (1)$$

where $t \in [t_{i-1}, T]$. The log-likelihood of the event sequence S is:

$$\begin{aligned}\ell(S) = &\sum_{i=1}^{L} \log \lambda^i(t_i, k_i, \boldsymbol{x}_i) \\ &- \int_0^T \int_{\mathbb{R}^d} \sum_{l=1}^{M} \lambda^i(\tau, l, \boldsymbol{s}) d\tau d\boldsymbol{s}. \quad (2)\end{aligned}$$

The second term in the above equation serves as a normalization term for the probability density.

• **Neural STPP** employs deep neural networks to parameterize the conditional intensity function $\lambda^i(t, k, \boldsymbol{x})$. For instance, Zhu et al. (2022) and Dong et al. (2023) leverage Multi-Layer Perceptron (MLP) to construct a non-stationary kernel for modeling the intricate inter-dependency within the intensity. Zhou et al. (2022) utilize the Transformer architecture to encode event history into a latent variable, subsequently deriving the intensity using Radial Basis Function (RBF) kernels. Chen et al. (2021) decompose spatial distribution from time and models the spatial distribution by CNFs, while the time distribution is learned through an ordinary differential equation (ODE) latent process. These methods estimate the model parameters by maximizing the log-likelihood presented in Eq. 2, where the evaluation of the intractable integral is either avoided by assuming restricted forms of event distribution or computed by numerical method.

• **Score Matching** (Hyvärinen, 2005) is a technique for estimating the parameters of unnormalized probability density models. It minimizes the expected squared difference between the model's log-density gradient (also known as the score) and the ground truth log-density gradient, preventing the calculation of the normalization integral. Denoising score matching (Vincent, 2011) enhances the scalability of score matching by adding a noise term to the original data points and matching the model's score to the noisy data's score. Yu et al. (2019) and Yu et al. (2022) further expand score matching to more general classes of domains. Notably, Meng et al. (2022) introduce concrete score matching for discrete data, broadening the application of score-based models in discrete domains.

• **Langevin Dynamics** is a widely-used sampling technique for score-based models, which generates samples from a target distribution using only its score function by simulating a continuous-time stochastic process. Hsieh et al. (2018) introduce Mirror Langevin Dynamics (MLD) as a variant specifically designed for sampling from distributions with constrained domains.

• **Confidence interval prediction** is an essential aspect of data modeling, providing a range that captures the inherent uncertainty in the data (Schruben, 1983). For continuous variables like event time and location in STPPs, we expect models to predict confidence intervals or regions associated with predefined confidence levels. The ideal model should nicely match the confidence level with the actual coverage of the generated confidence interval/regions; the closer the match, the more adeptly the model captures the underlying distribution. However, confidence intervals do not apply to discrete variables like event marks. In this case, we expect the model to learn a posterior distribution matching the actual data distribution. This involves comparing predicted probabilities to empirical accuracies (Guo et al., 2017), also known as calibration.

3. Method

3.1. Score Matching-based Pseudolikelihood

Retaining the notation introduced in Section 2, our goal is to learn $\lambda^i(t, k, \boldsymbol{x})$, the joint intensity given the history $\mathcal{H}_{t_i}$. We aim to employ the score matching technique to avoid the computation of the intractable integral in the log-likelihood (Eq. 2). However, score matching cannot be directly applied to the joint distribution of time, mark and location. This is because: (1) An intractable integral over location $\boldsymbol{x}$ remains after direct application and (2) event mark k is discrete where the gradient does not exist. To tackle these issues, we first decompose the joint intensity by conditioning on the event time and mark:

$$\lambda^i(t, k, \boldsymbol{x}) = \lambda^i(t, k) p^i(\boldsymbol{x} \mid t, k). \tag{3}$$

The first term is essentially the intensity function of a marked temporal point process (MTPP) without spatial features and the second term captures the spatial dynamics. We parametrize both the marginal intensity and the spatial distribution, and derive two score matching-based objectives for estimation, which will be introduced in Section 3.1.1 and 3.1.2, respectively.

3.1.1. Event time and mark

We parametrize the marginal intensity function on event time and mark as $\lambda^i(t, k; \theta)$ using a Transformer model, where θ represents the model parameters. For notational convenience, we omit θ in the subsequent context. Since score matching cannot be directly applied to $p^i(t, k)$ due to the discrete nature of mark, we further decompose $p^i(t, k)$ to $p^i(t \mid k) p^i(k \mid t)$, resulting in a pseudo-likelihood decomposition. We then apply score matching to the conditional distribution of event time $p^i(t \mid k)$ while adopting the conditional likelihood for event mark distribution $p^i(k \mid t)$. Here, the term "conditional" refers to the condition on time/mark besides the history.

The score function is defined as the gradient of log-probability density with respect to the data point. For conditional event time distribution $p^i(t \mid k)$, we have the score function as:

$$\psi_t^i(t \mid k) = \partial_t \log p^i(t \mid k) \tag{4}$$

$$= \partial_t \log \lambda^i(t, k) - \sum_{l=1}^{M} \lambda^i(t, l). \tag{5}$$

We can then formulate the expected score matching objective for the i-th event time:

$$\mathcal{L}_{\text{time}}^{(i)} = \frac{1}{2} \mathbb{E}_{t,k,\boldsymbol{x}} \left[\| \psi_t^i(t \mid k) - \psi_t^{i^*}(t \mid k) \|^2 \right], \tag{6}$$

where $\psi_t^{i^*}$ is the score function of the true event time distribution. However, this objective is unattainable as it depends on the unknown ground truth score. We can resolve this issue by following the general derivation in Hyvärinen (2005) and derive an empirical objective for the event sequence S:

$$\widetilde{\mathcal{L}}_{\text{time}}(S) = \sum_{i=1}^{L} \left[\frac{1}{2} \psi_t^i(t_i \mid k_i)^2 + \partial_t \psi_t^i(t_i \mid k_i) \right]. \tag{7}$$

By minimizing the above score matching objective, the scores of event time will be matched with the ground truth.

For event mark with undefined score, we can derive its conditional probability mass function $p^i(k \mid t)$ in a closed form:

$$p^i(k \mid t) = \frac{\lambda^i(t, k)}{\sum_{l=1}^{M} \lambda^i(t, l)}. \tag{8}$$

Then we learn this distribution by minimizing the negative conditional log-likelihood on the sequence S:

$$\widetilde{\mathcal{L}}_{\text{mark}}(S) = \sum_{i=1}^{L} -\log \frac{\lambda^i(t_i, k_i)}{\sum_{l=1}^{M} \lambda^i(t_i, l)}. \tag{9}$$

3.1.2. Event location

For the spatial distribution $p^i(\boldsymbol{x}|t,k)$, we parametrize the conditional score function $\psi^i_{\boldsymbol{x}}(\boldsymbol{x} \mid t,k) = \nabla_{\boldsymbol{x}} \log p^i(\boldsymbol{x}|t,k)$. This allows us to derive the empirical score matching objective for event location as:

$$\widetilde{\mathcal{L}}_{\text{spatial}}(S) = \frac{1}{L}\sum_{i=1}^{L}\Bigg[\frac{1}{2}\|\psi^i_{\boldsymbol{x}}(\boldsymbol{x}_i \mid t_i, k_i)\|^2 + \sum_{j=1}^{d} \partial_{x^{(j)}} \psi^{i\ (j)}_{\boldsymbol{x}}(\boldsymbol{x}_i \mid t_i, k_i)\Bigg], \tag{10}$$

where $x^{(j)}$ is the j-th dimension of location and $\psi^{i\ (j)}_{\boldsymbol{x}}$ is the j-th dimension of $\psi^i_{\boldsymbol{x}}$.

3.1.3. Score Matching-based Pseudolikelihood for Estimation

Given the score matching-based objectives for event time and location, and the conditional log-likelihood for event mark, we can estimate the model by minimizing their weighted sum:

$$\widetilde{\mathcal{L}}_{\text{SMASH}}(S) = \widetilde{\mathcal{L}}_{\text{time}}(S) + \widetilde{\mathcal{L}}_{\text{spatial}}(S) + \alpha \widetilde{\mathcal{L}}_{\text{mark}}(S), \tag{11}$$

where α is a hyperparameter that regulates the scale of the conditional log-likelihood objective.

Our method is essentially applying score matching to the first two distributions in the pseudo-likelihood decomposition $p^i(\boldsymbol{x}|t,k)p^i(t \mid k)p^i(k \mid t)$. Hence, we term it score matching-based pseudo-likelihood estimation. Additionally, we present the following theorem to demonstrate that the proposed objective in Eq. (11) satisfies local consistency.

Theorem 3.1. *Assume the events in sequence S follow the distribution: $p^\star(t,k,\boldsymbol{x}|\mathcal{H}) = p(t,k,\boldsymbol{x}|\mathcal{H};\theta^\star)$, and that no other parameter gives a pdf that is equal (almost everywhere w.r.t Lebesgue measure) to $p(t,k,\boldsymbol{x}|\mathcal{H};\theta^\star)$. Assume further that the optimization algorithm is able to find the global minimum and $p(t,k,\boldsymbol{x}|\mathcal{H};\theta)$ is positive for all $t, k, \boldsymbol{x}$, and θ. Then the estimator obtained by minimizing Eq. (11) is consistent, i.e., it converges in probability towards $\theta^\star$ when the sample size approaches infinity.*

3.2. Denoising Score Matching

In practice, the direct score-based objective given by Eq. 11 has two limitations: 1) the score functions of event location are inaccurately estimated in regions of low data density (Song and Ermon, 2019) and 2) the objective $\widetilde{\mathcal{L}}_{\text{time}}(S)$ contains second-order derivatives, leading to numerical instability. We resolve these issues by applying denoising score matching (Vincent, 2011) to the conditional distribution of event time and location, which reduces the area of the low-density region by perturbing data and improves stability by simplifying the objective. Specifically, we add Gaussian noise to events' time and location. Then we match the model to the perturbed data distribution $\widetilde{p}^i(\widetilde{t}, k, \widetilde{\boldsymbol{x}}) = \int_t \int_{\boldsymbol{x}} p^i(t,k,\boldsymbol{x}) q(\widetilde{t} \mid t) q(\widetilde{\boldsymbol{x}} \mid \boldsymbol{x}) dt d\boldsymbol{x}$, where $q(\widetilde{t} \mid t) \sim \mathcal{N}(t, \sigma_1)$ and $q(\widetilde{\boldsymbol{x}} \mid \boldsymbol{x}) \sim \mathcal{N}(\boldsymbol{x}, \boldsymbol{\sigma}_2)$. Denote the perturbed i-th event as $\{(\widetilde{t}^j_i, k, \widetilde{\boldsymbol{x}}^j_i)\}_{j=1}^{Q}$, where Q is the number of perturbed samples. The denoising variant of the score matching objective for event time and location can be expressed as

$$\widetilde{\mathcal{L}}^{\text{D}}_{\text{time}}(S) = \frac{1}{2}\sum_{i,j=1}^{L,Q}\left[\psi^i_{\widetilde{t}}(\widetilde{t}^j_i \mid k) - \partial_{\widetilde{t}} \log q(\widetilde{t}^j_i \mid t_i)\right]^2,$$

$$\widetilde{\mathcal{L}}^{\text{D}}_{\text{spatial}}(S) = \frac{1}{2}\sum_{i,j=1}^{L,Q}\left[\psi^i_{\widetilde{\boldsymbol{x}}}(\widetilde{\boldsymbol{x}}^j_i \mid k) - \partial_{\widetilde{\boldsymbol{x}}} \log q(\widetilde{\boldsymbol{x}}^j_i \mid t_i)\right]^2.$$

We then train the model by minimizing the following objective:

$$\widetilde{\mathcal{L}}^{\text{D}}_{\text{SMASH}}(S) = \widetilde{\mathcal{L}}^{\text{D}}_{\text{time}}(S) + \widetilde{\mathcal{L}}^{\text{D}}_{\text{spatial}}(S) + \alpha \widetilde{\mathcal{L}}_{\text{mark}}(S). \tag{12}$$

3.3. Sampling

With the learned intensities and score functions, we can generate new samples using Langevin Dynamics (LD) to provide point and confidence interval prediction for events. Suppose we want to generate the i-th event conditioning on the history $\mathcal{H}_{t_i}$, we first generate events' time and mark by sampling the initial time gap $t^{(0)}$ and event mark $k^{(0)}$ from a pre-specified distribution π. We then use LD to recursively update the continuous time gap with step size $\epsilon > 0$ following the equation:

$$t^{(n)} = t^{(n-1)} + \frac{\epsilon}{2}\psi^i_t(t_{i-1} + t^{(n-1)}, k^{(n-1)}) + \sqrt{\epsilon} w_n, \tag{13}$$

for $n = 1, \cdots, N$. Here, w_n is standard Gaussian noise. The event mark is updated by sampling $k^{(n)}$ from a categorical distribution defined by

$$p^i(k \mid t_{i-1} + t^{(n-1)}) = \frac{\lambda^i(t_{i-1} + t^{(n-1)}, k)}{\sum_l \lambda^i(t_{i-1} + t^{(n-1)}, l)}.$$

After the Langevin sampling, we utilize an additional denoising by Tweedie's formula (Efron, 2011), following Li et al. (2023):

$$\widehat{t} = t^{(N)} + \sigma_1 \psi^i_t(t_{i-1} + t^{(N)}, k^{(N)}). \tag{14}$$

We then sample $\widehat{k}$ from the updated categorical distribution to obtain the i-th event's time and mark as $(t_{i-1} + \widehat{t}, \widehat{k})$. We

proceed similarly to generate the event's location through N-step Langevin Dynamics given the generated time and mark:

$$\boldsymbol{x}^{(n)} = \boldsymbol{x}^{(n-1)} + \frac{\epsilon}{2}\psi^i_{\boldsymbol{x}}(\boldsymbol{x}^{(n-1)} \mid t_{i-1} + \widehat{t}, \widehat{k}) + \sqrt{\epsilon}\boldsymbol{z}_n, \quad (15)$$

$$\widehat{\boldsymbol{x}} = \boldsymbol{x}^{(N)} + \boldsymbol{\sigma}_2\psi^i_{\boldsymbol{x}}(\boldsymbol{x}^{(N)} \mid t_{i-1} + \widehat{t}, \widehat{k}), \quad (16)$$

where the initial $\boldsymbol{x}^{(0)}$ is sampled from an uniform distribution and $\boldsymbol{z}_n$ is standard multivariate Gaussian noise. We note that the sampling process involves two cascaded denoising processes, which is potentially less efficient than the parallel sampling used in (Yuan et al., 2023). To accelerate the process, we can sample event time and location in parallel, which we find does not significantly affect the results. Algorithm 1 in the appendix summarizes the procedure to generate the i-th event conditioning on the history $\mathcal{H}_{t_i}$.

With the generated samples, we can predict the next event and also compute confidence intervals. We predict the time and location of the next event by taking the mean of the time and location samples, respectively. For event mark, we assign the mode of the mark samples. We compute confidence intervals for event time by taking quantiles of the time samples, and compute the confidence regions for event location by thresholding the location samples' density. Additionally, we use the proportion of samples that contain the predicted mark as the confidence score to measure the model's calibration performance on event mark prediction.

3.4. Learning Marked Temporal Point Process

Our proposed method is general and can be easily applied to marked TPP data. By simply removing the objective for spatial features, we can learn the conditional intensity function $\lambda^i(t, k)$ for marked TPP model through minimizing

$$\widetilde{\mathcal{L}}^{\text{TPP}}_{\text{SMASH}}(S) = \widetilde{\mathcal{L}}^{\text{D}}_{\text{time}}(S) + \alpha\widetilde{\mathcal{L}}_{\text{mark}}(S). \quad (17)$$

From the learned intensities, event time and mark samples can be jointly generated by the former part of Algorithm 1.

Li et al. (2023) focus on marked TPPs rather than STPPs and propose SMURF-THP that applies score matching technique to provide uncertainty quantification. Compared to their work, we jointly model event time and mark by parameterizing an intensity function for each mark, whereas SMURF-THP employs a single intensity function for all event marks, hindering its ability to differentiate event time patterns for different marks. Furthermore, we capture event mark distribution by the joint intensity through a unified model. In contrast, SMURF-THP relies on an independent decoder, separated from the intensity function, to predict the mark of the next event, resulting in less accurate modeling of the time-mark dependency.

4. Experiments

We present an empirical evaluation of the proposed method by comparing its performance against multiple classical and neural STPP baselines on real-world datasets. We examine the models' performance on both point and confidence interval prediction of the next event given history. As the proposed framework is general and can be directly applied to marked TPPs, we also include a comparison with neural marked TPP models on marked TPP datasets. Further experimental details and results can be found in the Appendix. Our code is publicly available at https://github.com/zichongli5/SMASH.git.

Metrics: We employ four metrics to evaluate the model's performance:

• *Calibration Score (CS)* measures the calibration performance of the generated confidence intervals/regions across different confidence levels. It first calculates the calibration error for each level, which is determined by the difference between the actual coverage of the region and the desired confidence level. Then it takes the average calibration error across all different confidence levels. A lower CS denotes superior performance in capturing data distribution. In our experiment, we compute the average error at confidence levels from 0.5 to 1 in increments of 0.1 for STPPs, and from 0.8 to 1 in increments of 0.05 for TPPs. This choice is motivated by that confidence intervals/regions with higher levels are typically more useful.

• *Mean Absolute Error (MAE)* measures the mean difference between the point prediction and the ground truth of the events' times and locations. We make point predictions by taking the average time and location of the generated samples.

• *Mark Prediction Accuracy (Acc)* calculates the accuracy of the event mark predicted by the samples. We designate the mode of the generated samples' mark as the prediction.

• *Expected Calibration Error (ECE)* evaluates the model's calibration performance on event mark prediction. It measures the discrepancy between the predicted probabilities and the true observed frequencies of outcomes.

4.1. Marked Spatio-Temporal Point Process.

We first compare SMASH against classical and neural STPP baselines on Spatio-Temporal data, where we evaluate the calibration score and MAE on both event time and location, with accuracy and ECE for event mark prediction.

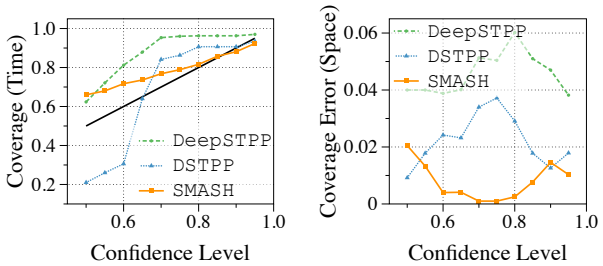
Datasets: We utilize the following three marked STPP datasets: (1) *Earthquake* dataset contains the location and time of all earthquakes in Japan from 1990 to 2020 with a

magnitude of at least 2.0 from the U.S. Geological Survey [1]. We partition all earthquakes into three categories: "small", "medium" and "large" based on their magnitude. (2) *Crime* dataset comprises reported crime from 2015 to 2020 provided by Atlanta Police Department [2]. Events are classified into four types according to the crime type. (3) *Football* (Yeung et al., 2023) dataset records football event data retrieved from the WyScout Open Access Dataset. Each event signifies an action made by the player, associated with the type of the action.

Baselines: We compare our model against two classical STPP methods and five neural STPP methods: (1) *Poisson Process* (Kingman, 1992) with mixtures of Gaussian for spatial distribution; (2) *Hawkes Process* (Hawkes, 1971) with mixtures of Gaussian; (3) *NJSDE* (Jia and Benson, 2019) adopts an SDE latent process to model temporal dynamics and utilize mixtures of Gaussian for spatial distribution; (4) *NSTPP* (Chen et al., 2021) incorporates an ODE latent process for temporal dynamics and CNFs for spatial distribution. (5) *NSMPP* (Zhu et al., 2022) utilizes neural networks to construct a non-stationary kernel for modeling the joint conditional intensity. (6) *DeepSTPP* (Zhou et al., 2022) introduces an RBF kernel-based parametrization for the joint intensity function that supports exact likelihood computation. (7) *DSTPP* (Yuan et al., 2023) leverages the diffusion model to capture the complex spatio-temporal dynamics. Note that we utilize multiple sampling methods to generate samples from baseline models for performance evaluation, with a detailed description in Appendix B.

Overall performance: Table 1 summarizes the results. We can observe that SMASH yields the best performance in terms of CS and MAE for event time, while achieving remarkable improvement in event location modeling. For event mark prediction, SMASH obtains comparable accuracy with DSTPP and the lowest ECE. The improvements of SMASH can be attributed to two factors: 1) SMASH learns the score function without restricted assumption on the event distribution, whereas NJSDE and DeepSTPP examine higher CS and MAE due to the inaccurate pre-specified parametric form. 2) SMASH optimizes through a normalization-free objective, while NJSDE, NSTPP and NSMPP suffer from inaccurate numerical approximation.

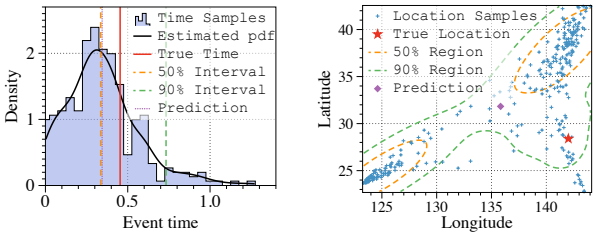
Different Confidence Level: We further compare the coverage of the confidence regions at different levels on the Crime dataset. Figure 1(a) displays the predicted confidence intervals' coverage for event time. The black diagonal represents the ideal coverage. Compared with DeepSTPP and DSTPP, SMASH is much closer to the black line, indicating that the generated confidence regions are more accurate. We can see that DeepSTPP and SMASH experience overconfidence on low confidence levels, while DSTPP tends to be underconfidence. As the level increases, SMASH approaches the correct coverage, whereas DSTPP and DeepSTPP present severe overconfidence. For event location, we highlight the coverage error of the confidence regions in Figure 1(b) to elucidate the distinctions between the models. As shown, SMASH presents the lowest coverage error for most confidence levels, with DeepSTPP being the worst.



(a) Event Time (b) Event Location

Figure 1. Comparison of coverage of different confidence levels on the Crime dataset.

Sample distribution visualization: We visualize the distribution of the generated samples for a randomly selected event from the Earthquake dataset. As shown in Figure 2, both the samples' times and locations exhibit a broad spread, which indicates the huge randomness within the event dynamics. Similar distributions is also observed among other baselines. This proves the perspective that single point prediction is not sufficient where confidence interval is needed for the evaluation of the predicted distribution. Furthermore, the samples' locations on the right present a multi-modal distribution, where the point prediction calculated from the average can easily diverge from the actual observation.



(a) Event Time (b) Event Location

Figure 2. Distribution of samples' times and locations generated by SMASH and the ground truth on the Earthquake dataset.

4.2. Marked Temporal Point Process.

As SMASH provides a general framework, we can also model marked TPP data using score-based objectives by simply removing the loss term of the spatial location. In this

[1] https://earthquake.usgs.gov/earthquakes/search/
[2] https://www.atlantapd.org

Table 1. Comparison of different methods' performance on three marked STPP datasets in terms of Calibration Score (CS) and Mean Absolute Error (MAE) for event time and location, Accuracy (Acc) and Expected Calibration Error (ECE) for event mark.

Datasets	Methods	$\text{CS}_{\text{time}}(\downarrow)(\%)$	$\text{MAE}_{\text{time}}(\downarrow)(\%)$	$\text{CS}_{\text{space}}(\downarrow)(\%)$	$\text{MAE}_{\text{space}}(\downarrow)(\%)$	Acc(↑)(%)	ECE(↓)(%)
Earthquake	Poisson	15.13±0.12	0.75±0.04	7.14±0.21	8.12±0.02	–	–
	Hawkes	14.62±0.10	0.71±0.05	7.15±0.19	8.15±0.05	–	–
	NJSDE	16.23±0.55	0.41±0.08	8.35±0.35	8.05±0.04	90.03±0.05	11.23±0.22
	NSTPP	14.66±0.89	0.45±0.12	7.33±0.43	7.45±0.05	90.13±0.03	12.21±0.12
	NSMPP	25.30±0.25	0.69±0.03	8.66±0.14	7.99±0.03	88.20±0.03	10.47±0.03
	DeepSTPP	22.77±1.32	0.38±0.01	7.31±0.25	7.55±0.10	90.11±0.09	21.20±1.50
	DSTPP	14.80±0.67	0.35±0.01	**5.81**±0.56	7.42±0.08	90.30±0.01	7.30±1.45
	SMASH	**3.53**±0.55	**0.29**±0.01	6.95±0.21	**7.37**±0.06	90.30±0.01	**4.85**±0.09
Crime	Poisson	18.06±0.25	1.82±0.11	8.06±0.18	0.062±0.01	–	–
	Hawkes	13.61±0.15	1.53±0.10	8.23±0.19	0.060±0.01	–	–
	NJSDE	17.33±0.25	1.72±0.21	7.89±0.70	0.083±0.20	36.26±0.04	13.42±0.65
	NSTPP	17.26±0.69	1.41±0.14	7.03±0.98	0.065±0.35	37.03±0.04	14.20±0.28
	NSMPP	16.09±0.25	1.63±0.12	9.15±0.08	0.058±0.017	36.30±0.03	15.47±0.23
	DeepSTPP	16.64±0.31	1.16±0.02	5.04±0.79	0.058±0.010	36.25±0.03	15.14±0.62
	DSTPP	14.50±1.81	1.43±0.13	1.92±1.48	0.057±0.012	39.33±0.93	14.65±1.53
	SMASH	**6.93**±0.36	1.16±0.06	**1.08**±0.12	0.057±0.008	39.40±1.56	**11.36**±0.78
Football	Poisson	23.13±0.77	6.50±0.13	3.78±0.10	36.71±0.03	–	–
	Hawkes	16.91±0.35	5.01±0.08	3.59±0.06	36.66±0.03	–	–
	NJSDE	19.21±0.55	4.85±0.25	11.32±0.75	36.66±0.20	65.58±0.63	9.53±0.21
	NSTPP	18.36±0.73	4.66±0.30	7.23±0.98	27.57±0.35	66.33±0.84	10.68±0.43
	NSMPP	19.02±1.19	4.85±0.41	8.62±0.57	30.65±0.65	65.80±0.21	9.37±0.22
	DeepSTPP	19.54±1.25	4.24±0.36	9.24±0.18	32.54±0.79	65.92±0.05	12.10±0.08
	DSTPP	14.53±4.22	3.76±0.24	2.79±0.81	**19.41**±0.07	68.62±0.86	4.64±0.83
	SMASH	**6.38**±0.54	**3.40**±0.08	**2.52**±0.34	20.35±0.13	68.38±1.03	**3.23**±1.68

subsection, we compare SMASH with neural TPP models on four real-world marked TPP data.

Datasets: We utilize the following four real-world datasets: (1) *StackOverflow* (Leskovec and Krevl, 2014). This dataset contains sequences of $6,633$ users' receipt of awards from a question answering website over a two-year period. Events are marked according to the type of the award, with a total of 21 distinct types. (2) *Retweet* (Zhao et al., 2015) dataset comprises $24,000$ sequences of tweets, where each sequence begins with the original tweet at time 0. Subsequent events represent retweets by other users, which are classified into three categories based on their follower counts. (3) *MIMIC-II* (Du et al., 2016) dataset is a comprehensive collection of clinical data from patients admitted to intensive care units (ICUs) over a seven-year period. We select a subset of 650 patients and construct sequences from their visit time and diagnosis codes. (4) *Financial Transactions* (Du et al., 2016) dataset records a total of 0.7 million transaction actions for a stock from the New York Stock Exchange. We partition the long single sequence of transactions into $2,000$ subsequences for evaluation. Each event is labeled with the transaction time and the action that was taken: buy or sell.

Baselines: We compare our model against the following five existing methods: (1) *NHP* (Mei and Eisner, 2017) designs a continuous-time LSTM to model the evolution of the intensity function by updating the latent state; (2) *NCE-TPP* (Mei et al., 2020) utilizes noise-contrastive estimation on MTPPs to bypass likelihood computation. (3) *SAHP* (Zhang et al., 2020) introduces a time-shifted positional encoding and employs self-attention to model the intensity function. (4) *THP* (Zuo et al., 2020) leverages the Transformer architecture to capture long-term dependencies in history and parameterizes the intensity function through a tailored continuous formulation. (5) *SMURF-THP* (Li et al., 2023) develops a score matching-based objective to train the THP model and predict confidence intervals for predicted arrival time.

Overall performance: Table 2 summarizes the results. We can see that SMASH outperforms other baselines by noticeable margins in terms of CS, and it achieves the lowest MAE on three datasets. Although SMURF-THP also leverages score matching for modeling marked TPPs, SMASH better captures the time-mark dependencies by modeling the joint intensity function and constructing a unified model. Additionally, SMASH also improves the ECE while exhibiting

Table 2. Comparison of different methods' performance on four marked TPP datasets in terms of Calibration Score (CS), Mean Absolute Error (MAE) for event time, Accuracy (Acc) and Expected Calibration Error (ECE) for event mark.

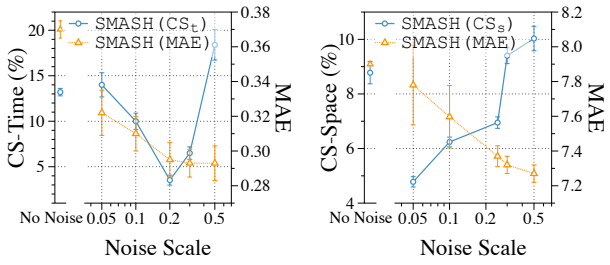
	StackOverflow				Retweet			
Methods	CS(%)(↓)	MAE(↓)	Acc(%)(↑)	ECE(%)(↓)	CS(%)(↓)	MAE(↓)	Acc(%)(↑)	ECE(%)(↓)
NHP	1.18±0.21	0.72±0.01	46.26±0.02	5.22±0.06	3.78±0.28	1.63±0.02	60.69±0.11	2.63±0.22
NCE-TPP	1.35±0.28	0.67±0.01	46.02±0.04	5.56±0.13	2.73±0.35	1.46±0.04	60.01±0.23	2.42±0.24
SAHP	0.85±0.21	0.67±0.01	46.15±0.03	5.13±0.08	1.71±0.15	1.10±0.02	60.65±0.13	2.57±0.18
THP	1.36±0.40	0.63±0.01	46.50±0.02	5.42±0.10	3.43±0.51	1.59±0.04	60.82±0.06	2.96±0.33
SMURF-THP	0.34±0.04	0.64±0.01	46.26±0.05	5.41±0.04	0.35±0.04	0.99±0.01	60.80±0.08	2.45±0.11
SMASH	**0.29**±0.12	0.63±0.01	46.26±0.14	**3.89**±0.03	**0.27**±0.09	0.96±0.01	60.80±0.06	**2.23**±0.05
	Financial				MIMIC			
Methods	CS(%)(↓)	MAE(↓)	Acc(%)(↑)	ECE(%)(↓)	CS(%)(↓)	MAE(↓)	Acc(%)(↑)	ECE(%)(↓)
NHP	1.66±0.21	1.96±0.05	60.39±0.25	4.15±0.13	1.43±0.10	0.99±0.01	83.10±0.91	15.80±0.63
NCE-TPP	1.64±0.27	2.30±0.16	60.12±0.08	4.38±0.35	1.36±0.68	1.13±0.01	83.17±0.67	14.62±1.09
SAHP	1.38±0.30	1.61±0.05	60.83±0.12	3.85±0.86	1.36±0.46	0.87±0.01	82.10±0.87	21.56±0.99
THP	1.54±0.01	1.89±0.01	60.84±0.30	3.48±0.22	1.20±0.37	1.09±0.01	83.73±0.05	13.35±0.81
SMURF-THP	1.28±0.06	1.40±0.01	60.85±0.38	3.71±0.15	1.14±0.23	0.87±0.01	83.72±0.48	15.65±0.85
SMASH	**0.81**±0.21	1.42±0.01	60.95±0.37	**2.30**±0.71	**0.85**±0.38	0.87±0.02	83.72±0.13	**12.23**±0.70

comparable accuracy with other baselines. We attribute it to the fact that we perturb event time with noise to use denoising score matching, which implicitly induces regularization for event mark prediction.

4.3. Ablation Study

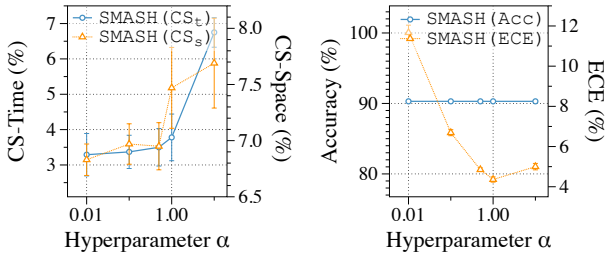
Denoising: SMASH perturbs data points with Gaussian noise and employs denoising score matching to achieve better stability and computational efficiency. We investigate the effects of noise scale σ by adding different amounts of noise to the data, including training without noise added. Results tested on the Earthquake dataset are presented in Figure 3. The figures suggest that adding perturbations effectively improves performance on both event time and location when a suitable noise scale is chosen. The calibration score of event time first decreases and then increases as the noise grows, while the calibration score of event location manifests a continuous increase. This suggests that a small noise is sufficient to cover the low-density regions in spatial distribution, while the event time requires a larger noise magnitude. Notably, the MAE of both event time and location decrease and tend to converge as the noise increase, implying that larger noise can bring smaller mean bias.

Hyperparameter α. We investigate our model's sensitivity to the hyperparameter α in the training objectives on the Earthquake dataset. As depicted in Figure 4, the calibration scores of both event time and location follow an increasing trend as α increases, while the ECE for event mark shows improvement. This observation aligns with our understanding of α as a scaler of the event mark modeling objective, and increasing it places more emphasis on fitting event mark distribution. The slight increase in ECE at α being 10 may hint at overfitting within the mark distribution.



(a) Event Time (b) Event Location

Figure 3. Performance of SMASH with different noise scales on the Earthquake dataset.



(a) Calibration Score (b) Event Mark

Figure 4. Sensitivity of the hyperparameter α in the training objective to model's performance on the Earthquake dataset.

5. Discussions

We remark that our work bears similarity to that of Yuan et al. (2023), which applies a denoising diffusion probabilistic model (DDPM) to STPPs. Their approach introduces multiple noise scales to perturb the event distribution and learns all perturbed distributions. In contrast, we only add the noise once and learn the corresponding distribution, greatly simplifying the framework. Moreover, they apply DDPM as a model-free method without parametric assumptions, while we utilize the self-modulating intensity formula in the Hawkes process to better capture the pattern. Notably, our experimental results indicate that complex diffusion processes are unnecessary for STPP modeling, perhaps due to the lower data dimension compared to the original images use case. It is also worth noting that while one could generate samples from some existing STPP methods, they do not consider or discuss confidence interval prediction for spatio-temporal events. SMASH naturally supports flexible sampling via score-based algorithms, achieving superior prediction performance.

In parallel, several studies on temporal point process (TPP) explore non-likelihood-based objectives to circumvent the computationally challenging integral calculation. For instance, Xiao et al. (2017) apply a discriminative learning method to estimate the model's parameters. However, it does not support flexible sampling due to the absence of an explicit intensity function. Sahani et al. (2016) also employ score matching for TPPs. However, they assume intensity functions are independent of historical events, oversimplifying the modeling of intricate event dependencies in modern data. TPPRL (Li et al., 2018) employ reinforcement learning (RL) for learning a policy to generate events in the setting of temporal point process (TPP), which can not be trivially extended to marked TPP/STPP. RLPP (Upadhyay et al., 2018) apply RL to marked TPP under an overly restrictive assumption of exponential intensity functions, which limits the ability to capture complex point processes. INITIATOR (Guo et al., 2018) and NCE-TPP (Mei et al., 2020) adopt noise-contrastive estimations to model marked TPP, while these two methods still need to deal with the intractable integral due to the likelihood-based approach for the training of noise generation network. As NCE-TPP has been shown to outperform INITIATOR by the authors, we include NCE-TPP in our set of baseline comparisons.

6. Conclusion

In this work, we present SMASH, a Score-MAtching based pSeudolikeliHood estimator for learning marked STPPs. We begin by decomposing the intensity function to separate the spatial features, thereby circumventing the intractable spatial integral that arises when applying the score-matching technique. Then we derive the score-matching objectives for the conditional likelihood of event time and location, and integrate the conditional likelihood of event mark as part of the objective. Confidence intervals for event time and location are obtained through flexible sampling using Langevin dynamics and the learned score function. We conduct experiments on various real-world datasets to illustrate that SMASH achieves superior performance under both marked STPPs and TPPs settings.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Chen, R. T. Q., Amos, B., and Nickel, M. (2021). Neural spatio-temporal point processes. In *ICLR*. OpenReview.net.

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

Dong, Z., Cheng, X., and Xie, Y. (2023). Spatio-temporal point processes with deep non-stationary kernels. In *ICLR*. OpenReview.net.

Du, N., Dai, H., Trivedi, R., Upadhyay, U., Gomez-Rodriguez, M., and Song, L. (2016). Recurrent marked temporal point processes: Embedding event history to vector. In *KDD*, pages 1555–1564. ACM.

Efron, B. (2011). Tweedie's formula and selection bias. *Journal of the American Statistical Association*, 106(496):1602–1614. PMID: 22505788.

González, J. A., Rodríguez-Cortés, F. J., Cronie, O., and Mateu, J. (2016). Spatio-temporal point process statistics: a review. *Spatial Statistics*, 18:505–544.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017). On calibration of modern neural networks. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR.

Guo, R., Li, J., and Liu, H. (2018). INITIATOR: noise-contrastive estimation for marked temporal point process. In *IJCAI*, pages 2191–2197. ijcai.org.

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

Hsieh, Y., Kavis, A., Rolland, P., and Cevher, V. (2018). Mirrored langevin dynamics. In *NeurIPS*, pages 2883–2892.

Hyvärinen, A. (2005). Estimation of non-normalized statistical models by score matching. *J. Mach. Learn. Res.*, 6:695–709.

Jia, J. and Benson, A. R. (2019). Neural jump stochastic differential equations. In *NeurIPS*, pages 9843–9854.

Kingman, J. F. C. (1992). *Poisson processes*, volume 3. Clarendon Press.

Leskovec, J. and Krevl, A. (2014). SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data.

Li, S., Wang, L., Chen, X., Fang, Y., and Song, Y. (2021). Understanding the spread of COVID-19 epidemic: A spatio-temporal point process view. *CoRR*, abs/2106.13097.

Li, S., Xiao, S., Zhu, S., Du, N., Xie, Y., and Song, L. (2018). Learning temporal point processes via reinforcement learning. In *NeurIPS*, pages 10804–10814.

Li, Z., Xu, Y., Zuo, S., Jiang, H., Zhang, C., Zhao, T., and Zha, H. (2023). SMURF-THP: score matching-based uncertainty quantification for transformer hawkes process. In *ICML*, volume 202 of *Proceedings of Machine Learning Research*, pages 20210–20220. PMLR.

Mei, H. and Eisner, J. (2017). The neural hawkes process: A neurally self-modulating multivariate point process. In *NIPS*, pages 6754–6764.

Mei, H., Wan, T., and Eisner, J. (2020). Noise-contrastive estimation for multivariate point processes. In *NeurIPS*.

Meng, C., Choi, K., Song, J., and Ermon, S. (2022). Concrete score matching: Generalized score matching for discrete data. *arXiv preprint arXiv:2211.00802*.

Okawa, M., Iwata, T., Kurashima, T., Tanaka, Y., Toda, H., and Ueda, N. (2019). Deep mixture point processes: Spatio-temporal event prediction with rich contextual information. In *KDD*, pages 373–383. ACM.

Sahani, M., Bohner, G., and Meyer, A. (2016). Score-matching estimators for continuous-time point-process regression models. In *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–5. IEEE.

Schruben, L. (1983). Confidence interval estimation using standardized time series. *Oper. Res.*, 31(6):1090–1108.

Song, Y. and Ermon, S. (2019). Generative modeling by estimating gradients of the data distribution. In *NeurIPS*, pages 11895–11907.

Tagliazucchi, E., Balenzuela, P., Fraiman, D., and Chialvo, D. R. (2012). Criticality in large-scale brain fmri dynamics unveiled by a novel point process analysis. *Frontiers in physiology*, 3:15.

Upadhyay, U., De, A., and Rodriguez, M. G. (2018). Deep reinforcement learning of marked temporal point processes. In *NeurIPS*, pages 3172–3182.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In *NIPS*, pages 5998–6008.

Vincent, P. (2011). A connection between score matching and denoising autoencoders. *Neural Comput.*, 23(7):1661–1674.

Xiao, S., Yan, J., Yang, X., Zha, H., and Chu, S. M. (2017). Modeling the intensity function of point process via recurrent neural networks. In *AAAI*, pages 1597–1603. AAAI Press.

Yeung, C. C., Sit, T., and Fujii, K. (2023). Transformer-based neural marked spatio temporal point process model for football match events analysis. *arXiv preprint arXiv:2302.09276*.

Yu, S., Drton, M., and Shojaie, A. (2019). Generalized score matching for non-negative data. *J. Mach. Learn. Res.*, 20:76:1–76:70.

Yu, S., Drton, M., and Shojaie, A. (2022). Generalized score matching for general domains. *Information and Inference: A Journal of the IMA*, 11(2):739–780.

Yuan, Y., Ding, J., Shao, C., Jin, D., and Li, Y. (2023). Spatio-temporal diffusion point processes. In *KDD*, pages 3173–3184. ACM.

Zhang, Q., Lipani, A., Kirnap, Ö., and Yilmaz, E. (2020). Self-attentive hawkes process. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, pages 11183–11193. PMLR.

Zhao, Q., Erdogdu, M. A., He, H. Y., Rajaraman, A., and Leskovec, J. (2015). SEISMIC: A self-exciting point process model for predicting tweet popularity. In *KDD*, pages 1513–1522. ACM.

Zhou, Z., Yang, X., Rossi, R. A., Zhao, H., and Yu, R. (2022). Neural point process for learning spatiotemporal event dynamics. In *L4DC*, volume 168 of *Proceedings of Machine Learning Research*, pages 777–789. PMLR.

Zhu, S., Wang, H., Dong, Z., Cheng, X., and Xie, Y. (2022). Neural spectral marked point processes. In *ICLR*. OpenReview.net.

Zuo, S., Jiang, H., Li, Z., Zhao, T., and Zha, H. (2020). Transformer hawkes process. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, pages 11692–11702. PMLR.

A. Model Architecture

We utilize the self-attention mechanism to encode both marked STPP and TPP data, as validated by previous research (Zuo et al., 2020; Zhang et al., 2020; Zhou et al., 2022; Yuan et al., 2023). This mechanism captures long-term dependencies by assigning an attention weight between any two events, with higher weights signifying stronger dependencies between those events. We adopt the approach of (Yuan et al., 2023) and utilize the Transformer Hawkes Process model (Zuo et al., 2020) to parameterize our intensity functions. Here we detail the model for marked STPP data.

Initially, each event in the sequence is encoded by summing the temporal encoding $\mathbf{C}_t$, spatial encoding $\mathbf{C}_{\boldsymbol{x}}$ and event mark embedding $\mathbf{C}_k$ (Yuan et al., 2023; Zuo et al., 2020), where $\mathbf{C}_t, \mathbf{C}_{\boldsymbol{x}}, \mathbf{C}_k \in \mathbb{R}^{L\times d}$ with d being the dimension of embedding. Through this process, the sequence S is encoded as $\mathbf{C} = \mathbf{C}_t + \mathbf{C}_{\boldsymbol{x}} + \mathbf{C}_k$. We then pass $\mathbf{C}$ along with each embedding $\mathbf{C}_t, \mathbf{C}_{\boldsymbol{x}}, \mathbf{C}_k$ through the self-attention module. Taking $\mathbf{C}$ as the example, the attention output $\mathbf{A}$ is computed as:

$$\mathbf{A} = \mathrm{Softmax}\left(\frac{\mathbf{Q}^\top \mathbf{K}}{\sqrt{d_k}}\right)\mathbf{V},$$
$$\mathbf{Q} = \mathbf{C}\mathbf{W}^Q, \quad \mathbf{K} = \mathbf{C}\mathbf{W}^K, \quad \mathbf{V} = \mathbf{C}\mathbf{W}^V.$$

The matrices $\mathbf{W}^Q, \mathbf{W}^K \in \mathbb{R}^{d\times d_k}$ and $\mathbf{W}^V \in \mathbb{R}^{d\times d_v}$ serve as weights for linear transformations that transform $\mathbf{C}$ into query, key, and value, respectively. To enhance model capacity, (Vaswani et al., 2017) suggest using multi-head self-attention. This involves inducing multiple sets of weights $\{\mathbf{W}_h^Q, \mathbf{W}_h^K, \mathbf{W}_h^V\}_{h=1}^H$ and computing different attention outputs $\{A_h\}_{h=1}^H$. The final attention output for the event sequence is obtained by concatenating $\{A_h\}_{h=1}^H$ and aggregating them with $\mathbf{W}^O \in \mathbb{R}^{(d_v * H)\times d}$:

$$\mathbf{A} = \mathrm{Concat}(\mathbf{A}_1, ..., \mathbf{A}_H)\mathbf{W}^O.$$

The attention output $\mathbf{A}$ is then processed through a position-wise feed-forward neural network (FFN) to obtain the hidden representations:

$$\mathbf{H} = \mathrm{ReLU}(\mathbf{A}\mathbf{W}_1^{\mathrm{FFN}} + \mathbf{b_1})\mathbf{W}_2^{\mathrm{FFN}} + \mathbf{b_2},$$
$$\mathbf{h}(i) = \mathbf{H}(i, :).$$

In this context, $\mathbf{h}(i)$ encodes the i-th event and all past events up to time t_i. We incorporate future masks during the computation of attention to prevent learning from future events. We stack multiple self-attention modules and FFNs to construct a larger model that can capture high-level dependencies.

Following the above computation, we obtain $\mathbf{h}(i), \mathbf{h}_t(i), \mathbf{h}_{\boldsymbol{x}}(i), \mathbf{h}_k(i)$ as the encoding of different aspects of the event history. We then parametrize the intensity functions $\lambda^i(t, k)$ given history $\mathcal{H}_{t_i}$ using multiple layers of network, where each layer follows:

$$\mathbf{h}_{tk}^i = \sigma(\mathbf{W}_t t + \mathbf{b}_t + \mathbf{W}_h \mathbf{h}(i) + \mathbf{b}_h + \mathbf{W}_{tk}(\mathbf{h}_t(i) + \mathbf{h}_k(i)) + \mathbf{b}_{tk}).$$

Here, $\mathbf{W}_t \in \mathbb{R}^{d\times 1}, \mathbf{W}_h, \mathbf{W}_{tk} \in \mathbb{R}^{d\times d}, \mathbf{b}_t, \mathbf{b}_h, \mathbf{b}_{tk} \in \mathbb{R}^d$ are trainable parameters. σ denotes the RELU activation function. We stack three such layers and pass the output to another FFN, with the softplus activation in the last layer:

$$\lambda^i(t, k) = \mathrm{Softplus}(\mathrm{FFN}(\mathbf{h}_{tk}^i)).$$

For score function of event location, we employ the Co-attention Denoising Network (CDN) proposed by Yuan et al. (2023):

$$\psi^i(\boldsymbol{x} \mid t, k) = \mathrm{CDN}(\mathbf{h}(i), \mathbf{h}_t(i), \mathbf{h}_{\boldsymbol{x}}(i), \mathbf{h}_k(i)).$$

B. Experiment Detail

B.1. Training Detail

Dataset Preprocessing: (1) Earthquake: We adopt the same preprocessing procedure as in Chen et al. (2021), with the exception that we exclude earthquakes with magnitudes below 2.0. (2) Crime: We select crime events from 2015 to 2020 at Atlanta with the following crime types: "Burglary", "Agg Assault", "Robbery" and "Homicide". The occurrence time of the crime serves as the event time, with longitude and latitude representing spatial features. (3) Football: We follow

the same preprocessing as Yeung et al. (2023). (4) Four TPP datasets: We adopt the same data preprocessing as those used by (Du et al., 2016) and (Mei and Eisner, 2017). For additional details and downloadable links, please refer to the aforementioned papers. Table 5 summarizes the statistics of the seven datasets used in the experiments. We randomly split all datasets to train/test/valid by the proportion of 0.8/0.1/0.1. As the datasets' scales differ, we employ normalization and log-normalization techniques. Specifically, we log-normalize event time by $\frac{\log(t)-\text{Mean}(\log(t))}{\text{Var}(\log(t))}$ and apply standard normalization for event location. We rescale back the generated samples before evaluation.

Hyperparameters: In STPP experiments, we employ the same backbone architecture as DSTPP to ensure a fair comparison and maintain default hyperparameters for other baselines. In TPP experiments, we employ the same backbone architecture as SMURF-THP. We note that while the original NCE-TPP uses LSTM as the backbone, we implement it using the same Transformer backbone as ours. A detailed hyperparameter breakdown for the adopted backbone architecture can be found in Table 3. During training, we use the Adam optimizer and train all models for 150 epochs on an NVIDIA Tesla V100 GPU. Hyperparameters specific to our method were fine-tuned via grid search and are detailed in Table 4. The number of perturbations and samples are fixed at 300 for STPPs and 100 for TPPs; increasing these may improve the models' performance. The two numbers of noise scale on marked STPP datasets signify the noise scale on event time and location.

Table 3. Summary of backbone architecture hyperparameters.

Dataset	#head	#layer	d_{model}	$d_k = d_v$	d_{hidden}	dropout	batch	learning rate
Earthquake	4	4	16	16	64	0.1	32	1e-3
Crime	4	4	64	16	256	0.1	32	1e-3
Football	4	4	32	16	128	0.1	4	1e-3
StackOverflow	4	4	64	16	256	0.1	4	1e-3
Retweet	3	3	64	16	256	0.1	16	5e-3
Financial	6	6	128	64	2048	0.1	1	1e-4
MIMIC-II	3	3	64	16	256	0.1	1	1e-4

Table 4. Summary of hyperparameters for the method.

Dataset	Loss weight α	noise scale σ	Langevin step size ϵ	#step
Earthquake	0.5	0.2/0.25	0.005	2000
Crime	0.5	0.3/0.03	0.005	1000
Football	0.5	0.2/0.1	0.01	2000
StackOverflow	0.2	0.1	0.005	2000
Retweet	0.5	0.1	0.0003	2000
Financial	0.5	0.01	0.005	2000
MIMIC-II	0.5	0.005	0.002	200

B.2. Computing Confidence Interval/Region

Let's denote the Q generated samples as $\{(t_i^j, \boldsymbol{x}_i^j, k_i^j)\}_{j=1}^Q$. For event time, which often follows long-tail distribution, we calculate the q-confidence level interval as $[0, t_i^q]$, with t_i^q being the q-th quantile of sample times. For event location, we first obtain the estimated pdf through gaussian kernel density estimation. A threshold is then determined such that the region with a density exceeding this threshold satisfies the desired confidence level. This delineated region becomes the final confidence region.

B.3. Baselines

Most of the neural STPP baselines we consider do not inherently support marked STPP modeling. We augment them for marked STPP by parametrizing an intensity function for each mark based on their original approach. We utilize multiple sampling methods to adapt to different baselines. For all baselines except DSTPP, we sample event time by Langevin Dynamics from the learned intensities as in our method. We sample event location for NJSDE by directly sampling from the learned Gaussian distribution. For NSTPP, we sample through their CNF procedure, that is, sampling starts from the initial distribution followed by the ODE progression. Both NSMPP and DeepSTPP, which model joint intensity, utilize Langevin

Dynamics for event location sampling. DSTPP sampling adheres to its native diffusion sampling method.

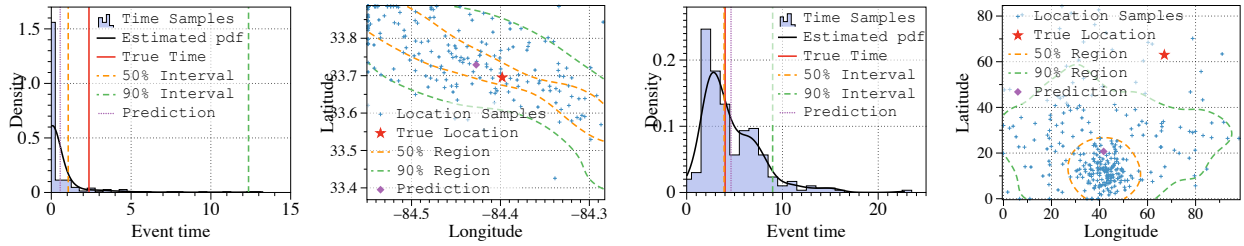
Table 5. Datasets statistics containing the name of the dataset, the number of event types, the number of events, and the average length per sequence

Dataset	#Type	#Event	Average Length
Earthquake	3	88064	73
Crime	4	35381	141
Football	7	67408	688
StackOverflow	22	480413	64
Retweet	3	2173533	109
Financial	2	414800	2074
MIMIC-II	75	2419	4

C. Additional Results

C.1. Sample Visualization

We visualize the sample distribution for two randomly selected events from the Crime and Football datasets in Figure 5. We can observe a long-tail distribution of event time and a wide-spread distribution of event location.

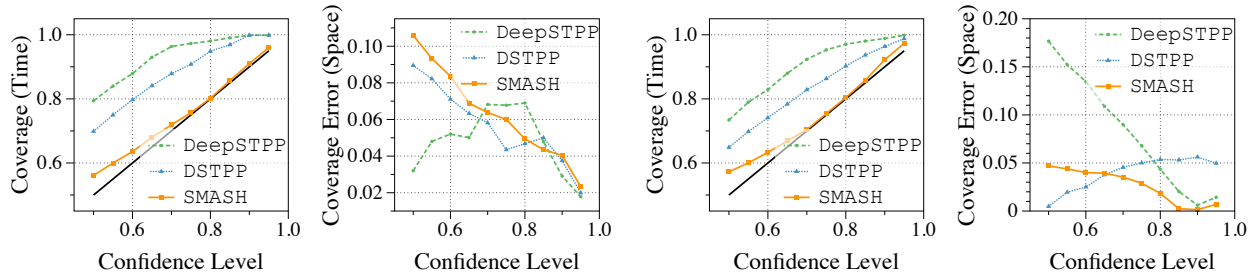


(a) Crime (b) Football

Figure 5. Distribution of samples' times and locations generated by SMASH and the ground truth on the Crime and Football dataset.

C.2. Different Confidence Levels

We display coverage and coverage error for different confidence levels on the Earthquake and Football datasets in Figure 6. For event time, SMASH consistently outperforms DeepSTPP and DSTPP across all confidence levels. For event location, SMASH achieves comparable coverage error with DSTPP on the Earthquake dataset and slightly better performance on the Football dataset.



(a) Earthquake (b) Football

Figure 6. Comparison of coverage of different confidence levels on the Earthquake and Football dataset.

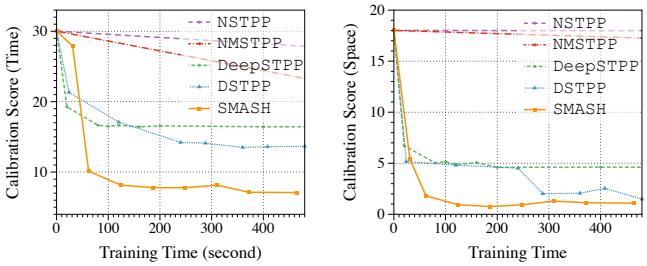
C.3. Computational cost

We compare the computational costs of different baselines by measuring the training runtime on the Crime dataset, which contains over 30,000 events. Table 6 presents the one epoch training time. We can observe that SMASH yields comparable training time per epoch compared to DeepSTPP and DSTPP, which also avoid the computation of intractable integrals. Conversely, NSTPP and NSMTPP require significantly more time; NSTPP involves numerical integration over ODE, and NSMTPP necessitates MC approximation for the integral calculation.

Furthermore, the overall training cost is influenced by the training convergence rate. Therefore, we also assessed the performance of each baseline under different runtime budgets, as illustrated in Figure 7. Our method, SMASH, demonstrates rapid convergence to optimal performance. DSTPP and DeepSTPP also achieve convergence within 200 seconds, whereas NSMTPP and NSTPP are considerably more time-consuming.

Table 6. Training time of 1 epoch of all baselines.

Methods	Training Time of 1 epoch (s)
NSTPP	486.0
NSMTPP	53.2
DeepSTPP	2.0
DSTPP	2.4
SMASH	3.1



(a) Calibration Score on Time (b) Calibration Score on Space

Figure 7. Performance of baselines under different training time.

D. Proof of Theorem 3.1

Proof. Let $\psi^\star_{\mathbf{x}}(\mathbf{x}|t,k,\mathcal{H})$ and $\psi^\star_t(t|k,\mathcal{H})$ be the associated score functions of the true distributions $p^\star(\mathbf{x}|t,k,\mathcal{H})$ and $p^\star(t|k,\mathcal{H})$, respectively. Denote $\psi_{\mathbf{x}}(\mathbf{x}|t,k,\mathcal{H};\theta)$ and $\psi_t(t|k,\mathcal{H};\theta)$ as the score functions of the model with parameter θ. The score matching objective we use in Eq. (11) is an empirical estimator of the following objective:

$$\mathcal{L}(\theta) = \mathbb{E}_{p^\star}\Bigg\{\sum_{i=1}^{L}\Big\{\frac{1}{2}\big[\psi_{\mathbf{x}}(\mathbf{x}_i|t_i,k_i,\mathcal{H}_{t_i};\theta) - \psi^\star_{\mathbf{x}}(\mathbf{x}_i|t_i,k_i,\mathcal{H}_{t_i})\big]^2 + \frac{1}{2}\big[\psi_t(t_i|k_i,\mathcal{H}_{t_i};\theta) - \psi^\star_t(t_i|k_i,\mathcal{H}_{t_i})\big]^2 - \log p(k_i|t_i,\mathcal{H}_{t_i};\theta)\Big\}\Bigg\},$$

where the expectation is over each event in the sequence following the true distribution $p^\star$. We first prove that $\mathcal{L}(\theta) \geq \mathcal{L}(\theta^\star)$ for all θ and the equality holds when $\theta = \theta^\star$.

The first two score matching terms of $\mathcal{L}(\theta)$ must be larger than or equal to those of $\mathcal{L}(\theta^\star)$ as the latter equals zero. The

inequality of the last log-likelihood term can be derived as

$$\mathbb{E}_{(t_i,k_i)\sim p^*}[-\log p(k_i|t_i,\mathcal{H}_{t_i};\theta^*)] - \mathbb{E}_{(t_i,k_i)\sim p^\star}[-\log p(k_i|t_i,\mathcal{H}_{t_i};\theta)] = \mathbb{E}_{(t_i,k_i)\sim p^*}[\log \frac{p(k_i|t_i,\mathcal{H}_{t_i};\theta)}{p^*(k_i|t_i,\mathcal{H}_{t_i})}]$$
$$\leq \sum_{k_i=1}^{M}\int_{t_i}(\frac{p(k_i|t_i,\mathcal{H}_{t_i};\theta)}{p^*(k_i|t_i,\mathcal{H}_{t_i})}-1)p^\star(t_i,k_i|\mathcal{H}_{t_i})dt_i$$
$$= \sum_{k_i=1}^{M}\int_{t_i}(p(k_i|t_i,\mathcal{H}_{t_i};\theta)-p^\star(k_i|t_i,\mathcal{H}_{t_i}))p^\star(t_i|\mathcal{H}_{t_i})dt_i = \int_{t_i}(1-1)p^\star(t_i|\mathcal{H}_{t_i})dt_i = 0.$$

By summing up all events, we have that the last term of $\mathcal{L}(\theta)$ is also greater than or equal to that of $\mathcal{L}(\theta^\star)$. Then we get $\mathcal{L}(\theta) \geq \mathcal{L}(\theta^\star)$. To prove the second statement, we look at the three terms step by step. For simplicity, we omit the condition on history in the following.

If $\mathcal{L}(\theta) = \mathcal{L}(\theta^\star)$, the first term in $\mathcal{L}(\theta)$ must be zero. Since $p^\star(\cdot)$ is positive, we can infer that $\psi_{\mathbf{x}}(\cdot;\theta)$ and $\psi^*_{\mathbf{x}}(\cdot)$ are equal. This implies $\log p^\star(\mathbf{x}|t,k) = \log p(\mathbf{x}|t,k;\theta) + c$ for some constant c. Because both $p^\star$ and p are pdfs, the constant c must be 0, and hence we have $p^\star(\mathbf{x}|t,k) = p(\mathbf{x}|t,k;\theta)$. Similarly, for the second term, we can get $p^\star(t|k) = p(t|k;\theta)$ and hence $\frac{p^\star(t,k)}{p^\star(k)} = \frac{p(t,k;\theta)}{p(k;\theta)}$. From the third term, we can obtain $p^\star(k|t) = p(k|t;\theta)$ given the above derivation. So we have $\frac{p^\star(t,k)}{p^\star(t)} = \frac{p(t,k;\theta)}{p(t;\theta)}$. By dividing the two equations, we get $\frac{p^\star(t)}{p^\star(k)} = \frac{p(t;\theta)}{p(k;\theta)}$. Taking integration over t on both sides, we have $p^\star(k) = p(k;\theta)$ and therefore we get $p^\star(t,k) = p(t,k;\theta)$. Adding that $p^\star(\mathbf{x}|t,k) = p(\mathbf{x}|t,k;\theta)$, we have the joint distributions to be equal: $p^\star(\mathbf{x},t,k) = p(\mathbf{x},t,k;\theta)$. By assumption, $\theta^\star$ is the only parameter that fulfills this equation, so necessarily $\theta = \theta^\star$.

Then, according to the law of large numbers, the empirical version of the loss converges to $\mathcal{L}(\theta)$ as the sample size approaches infinity. Thus, the estimator converges to a point where $\mathcal{L}(\theta)$ is globally minimized. Considering $\mathcal{L}(\theta^\star) = \mathcal{L}(\theta) \Rightarrow \theta = \theta^\star$, the minimum is unique and must be found at the true parameter $\theta^\star$. □

E. Sampling Algorithm

Algorithm 1 Sampling i-th event given history $\mathcal{H}_{t_i}$

Target: generate the i-th event $(\widehat{t}_i, \widehat{\boldsymbol{x}}_i, \widehat{k}_i)$.
$(t^{(0)}, \boldsymbol{x}^{(0)}, k^{(0)}) \sim \pi$
for $n = 1, 2, \cdots, N$ **do**
  $w_n \sim \mathcal{N}(0,1)$
  $\boldsymbol{z}_n \sim \mathcal{N}(\mathbf{0},\mathbf{1})$
  Update $t^{(n)}$ according to Eq. 13
  $k^{(n)} \sim$ Categorical$(\frac{\lambda^i(t_{i-1}+t^{(n-1)},k)}{\sum_{l=1}^{M}\lambda^i(t_{i-1}+t^{(n-1)},l)})$
  Update $\boldsymbol{x}^{(n)}$ according to Eq. 15 using $t^{(n)}$ and $k^{(n)}$
end for
Calculate $\widehat{t}$ based on Eq. 14
$\widehat{k} \sim$ Categorical$(\frac{\lambda^i(t_{i-1}+\widehat{t},k)}{\sum_{l=1}^{M}\lambda^i(t_{i-1}+\widehat{t},l)})$
Calculate $\widehat{\boldsymbol{x}}$ based on Eq. 16
return $\widehat{t}_i = \widehat{t} + t_{i-1}, \widehat{\boldsymbol{x}}_i = \widehat{\boldsymbol{x}}, \widehat{k}_i = \widehat{k}$.