# CONTEXT AWARE DIFFUSION IN TEMPORAL EVENT STREAMS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

In the context of continuous time event sequence prediction, diffusion models have achieved state-of-the-art performance. These methods propose to model the entire event sequence by simultaneously diffusing both the time intervals and the event types of the target sequence. However, they employ the default forward noising process, which does not take into consideration the local structural and temporal patterns observed in event sequences. In this work, we propose a locality-based structure-aware noising technique, which customizes the noise scale at each event, considering temporal patterns in the neighborhood. Building on this framework, we further develop specialized training and sampling procedures tailored to the proposed noising mechanism and provide theoretical analysis to support our design choices. We validate our approach in five real-world datasets while comparing several state-of-the-art baselines for both long- and short-horizon forecasting tasks. The proposed method achieves up to $20.11\%$ in time prediction and up to $7.91\%$ in type prediction over the most competitive baseline. Our quantitative and qualitative evaluations establish the effectiveness of data-driven locality-aware noising in diffusion models for continuous-time event sequences.

## 1 INTRODUCTION

Temporal event streams are widely observed across both natural and social systems. Examples include opinion posts on online social networks (De et al., 2016), financial trades on stock markets (Bacry et al., 2015), geo-temporal phenomenon such as earthquakes (Touati et al., 2014) inter alia. Typically, each event in such sequences is characterized by a timestamp as well as a discrete/continuous description of the event.

**Temporal Point Process (TPP) methods.** Marked temporal point processes (Daley & Vere-Jones, 2007) and their neural variants, including recurrent neural networks (Du et al., 2016; Mei & Eisner, 2017; Xiao et al., 2017), self-attention for improved modeling of distant past events (Zhang et al., 2020; Zuo et al., 2020), normalizing flows (Shchur et al., 2019), and diffusion (Lin et al., 2022), have been adopted for modeling temporal event streams. More recent works leverage Neural ODEs (Song et al., 2024) and residual modeling (Yuan & Fang, 2025). While most of these methods demonstrate excellent performance in next-event prediction, they often perform poorly for long-horizon forecasting due to their autoregressive nature, which leads to rapid accumulation of prediction errors over long time horizons.

To address this limitation, diffusion has recently emerged as an effective strategy for long-horizon forecasting. Recent non-autoregressive diffusion-based approaches include (Lüdke et al., 2024), which progressively refines entire event sequences from noise; (Zeng et al., 2023), which employs two distinct diffusion processes for event times and event types that interact through their respective denoising functions; (Zhou et al., 2025), which jointly denoises complete future event sequences; (Yuan et al., 2023), which extends diffusion models to spatio-temporal point processes; and (Zhang et al., 2024), which models intensity evolution as a neural jump-diffusion SDE, providing a continuous-time, intensity-based perspective. See Appendix D for a detailed discussion.

While diffusion-based approaches achieve state-of-the-art performance in long-horizon forecasting, existing methods typically employ standard Gaussian white noise for the forward noising process, following the predominant practice in the diffusion literature. However, temporal event sequences often exhibit diverse inter-arrival time distributions, resulting in a mixture of dense and sparse



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

temporal regions. Consequently, a uniform noising process may not be equally suitable across different regions of an event sequence. In particular, a noise level appropriate for sparse regions may destroy the fine-grained temporal patterns present in dense regions too rapidly, making them difficult to recover during the reverse (denoising) process. We argue that noise should instead be injected dynamically during the forward diffusion process in a locally sensitive manner, adapting the noise level to the temporal density of each region. This, in turn, enables the reverse denoising process to better preserve and recover the varying degrees of sparsity exhibited by real-world temporal event sequences.

**Data-dependent noising.** Recently there are a growing line of work showing that non-uniform, data-dependent noise schedules can outperform fixed ones—e.g., learnable variance schedules (Nichol & Dhariwal, 2021), pixel-wise adaptive noise Sahoo et al. (2024), heat dissipation based diffusion model Rissanen et al. (2022), non-isotropic Gaussian noise Voleti et al. (2022), time-varying correlated noise Huang et al. (2024), etc (refer to Appendix D for details). However, none have addressed adaptive non-white Gaussian noise in the context of temporal event sequences, where irregular and highly variable inter-arrival patterns create unique challenges.

**Our work.** We, therefore, propose a noising strategy – SCALEDIFF that customizes event noise variances in a locality-sensitive manner, downscaling noise variances for dense localities. Building upon this framework, we introduce dedicated training and sampling strategies specifically designed to accommodate the proposed noising mechanism. In addition, we provide theoretical justifications that highlight the soundness of the proposed design choices.

To summarize our key contributions:

(1) We introduce context-aware adaptive forward noising strategies that adjust the noise scale based on temporal structure—an event-specific characteristic—rather than employing a uniform Gaussian noise.

(2) We develop training and sampling strategies specific to the proposed noising approach and theoretically show that it ensures bounded variance and convergence to Gaussian noise while acting as a principled regularization mechanism.

(3) The effectiveness of our proposed method is demonstrated through forecasting experiments across five real-world datasets with improvements up to $20.11\%$ in time prediction, and $7.91\%$ in type prediction over the most competitive baseline.

## 2 PROBLEM STATEMENT

A temporal event stream is characterized by a sequence of events $\boldsymbol{s} = \{e_i = (t_i, m_i)\}_{i=1}^{M}$, where each event $e_i$ is denoted by a tuple $(t_i, m_i)$. Here, $t_i \in \mathbb{R}^+$ corresponds to the timestamp of event $e_i$, and $m_i$ corresponds to the type of the event, assuming that the event type can be among one of $K$ discrete categories. Now, given a history of events in a sequence, the task is to predict future $N$ events in the sequence, where $N$ is fixed apriori.

In this work, we directly model the event sequence distribution conditioned on history. Furthermore, instead of modeling $t_i$, the exact time of occurrence of the event $e_i$, we model $x_i = t_i - t_{i-1}$, the time interval between event $e_i$ and $e_{i-1}$. To address the forecasting problem, we propose to learn the conditional distribution of the sequence $P(\boldsymbol{s}_u|\boldsymbol{s}_c)$, where $\boldsymbol{s}_u = (\boldsymbol{x}_u, \boldsymbol{m}_u)$ denote the $N$ future events, which we define as the target sequence and $\boldsymbol{s}_c$ is the event history. $\boldsymbol{s}_t, t = 0, .., T$ denotes the noisy event sequence at $t-$th diffusion step, where $\boldsymbol{s}_{t,i}$ denotes $i$ th event in that sequence. Specifically, we choose a backbone diffusion framework (Zeng et al., 2023), in which both time intervals $\boldsymbol{x}_u$ and event types $\boldsymbol{m}_u$ are diffused. Subsequently we propose a novel noising technique that improves the learning by allowing finer transitions between the consecutive noisy states. We briefly describe the backbone diffusion framework in the following section followed by our proposed noising technique.

## 3 DIFFUSION FRAMEWORK

Given a temporal sequence, we first apply the Box-Cox transformation Tukey (1957) to $\boldsymbol{x}_u$ which maps it from strictly positive real values $\mathbb{R}^+ = (0, \infty)$ to unrestricted real values $\mathbb{R} = (-\infty, \infty)$.This transformation allows us to make the inter-arrival distribution more like a Gaussian. To encode the

context, we apply a transformer-based context encoder (Xue et al., 2022) that encodes the variable-length context $\boldsymbol{s}_c$ to generate a fixed-dimensional intermediate representation denoted as $h = f(\boldsymbol{s}_c)$. Note that, during sampling we perform a reverse mapping on generated time intervals to obtain positive event time intervals.

**Forward Diffusion.** We define a single diffusion process for both time-intervals and event types. Let the original sequence be $\boldsymbol{s}_0 = (\boldsymbol{x}_0, \boldsymbol{m}_0)$, where $\boldsymbol{x}_0 = \boldsymbol{x}_u$ and $\boldsymbol{m}_0 = \boldsymbol{m}_u$. We define the forward diffusion process as the following joint distribution: $q(\boldsymbol{s}_{1:T}|\boldsymbol{s}_0) = \prod_{t=1}^{T} q(\boldsymbol{s}_t|\boldsymbol{s}_{t-1})$.

Given a linear increasing variance schedule $\{\beta_1, .., \beta_T\}$, the forward noising distribution can be defined as:

$$q(\boldsymbol{s}_t|\boldsymbol{s}_{t-1}) = q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})q(\boldsymbol{m}_t|\boldsymbol{m}_{t-1}) \tag{1}$$

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t\mathcal{I}) \tag{2}$$

$$q(\boldsymbol{m}_t|\boldsymbol{m}_{t-1}) = CAT(\boldsymbol{m}_t; (1-\beta_t)\boldsymbol{m}_{t-1}, \frac{\beta_t}{K}) \tag{3}$$

$$q(\boldsymbol{m}_T) = CAT(\boldsymbol{m}_t; 1/K), \quad q(\boldsymbol{x}_T) = \mathcal{N}(\boldsymbol{x}_t; \boldsymbol{0}, \mathcal{I}) \tag{4}$$

**Denoising.** The denoising processes of the inter-arrival times and the event types are conditional on the historical context $\boldsymbol{s}_c$ and interact with each other. The denoising process $p_\theta(\boldsymbol{s}_{t-1}|\boldsymbol{s}_t)$ is defined as

$$\begin{aligned} p_\theta(\boldsymbol{s}_{t-1}|\boldsymbol{s}_t, \boldsymbol{s}_c) &= p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{s}_t, \boldsymbol{m}_{t-1}, \boldsymbol{s}_c)p_\theta(\boldsymbol{m}_{t-1}|\boldsymbol{s}_t, \boldsymbol{s}_c) \\ p_\theta(\boldsymbol{m}_{t-1}|\boldsymbol{s}_t, \boldsymbol{s}_c) &= CAT(\boldsymbol{m}_{t-1}|\pi_\theta(\boldsymbol{x}_t, \boldsymbol{m}_t, \boldsymbol{s}_c, t)) \\ p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{m}_{t-1}, \boldsymbol{s}_t, \boldsymbol{s}_c) &= \mathcal{N}(\boldsymbol{x}_{t-1}; \mu_\theta(\boldsymbol{m}_{t-1}, \boldsymbol{x}_t, \boldsymbol{s}_c, t), \sigma_t) \end{aligned} \tag{5}$$

$\mu_\theta$ and $\pi_\theta$ are learnable and $\sigma_t$ is set to $\beta_t$. $\mu_\theta$ is further parameterized as

$$\mu_\theta(\boldsymbol{x}_t, \boldsymbol{m}_{t-1}, t, \boldsymbol{s}_c) = \frac{\boldsymbol{x}_t}{\sqrt{\alpha_t}} - \frac{\beta_t \epsilon_\theta(\boldsymbol{x}_t, \boldsymbol{m}_{t-1}, t, \boldsymbol{s}_c)}{\sqrt{\alpha_t}\sqrt{1-\bar{\alpha}_t}} \tag{6}$$

where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$. For event types, we further learn $\phi_\theta$ that learns to predict the target event type $\hat{\boldsymbol{m}}_0$ at each step $t$, while taking as input a noisy sample at step $t$ as $\hat{\boldsymbol{m}}_0 = \phi_\theta(\boldsymbol{m}_t, \boldsymbol{x}_t, t, \boldsymbol{s}_c)$. Finally, this target event type $\hat{\boldsymbol{m}}_0$ is combined with the noisy event type $\boldsymbol{m}_t$, and normalized to generate a target type distribution $\pi_\theta$ as follows, whereas $\epsilon_\theta, \phi_\theta$ and $f_\theta$ are parameterized by transformer-based networks and $\odot$ denotes the Hadamard product. Refer Zeng et al. (2023) for details (see Appendix E).

$$\begin{aligned} \boldsymbol{\theta}(\boldsymbol{m}_t, \hat{\boldsymbol{m}}_0) &= [\alpha_t \boldsymbol{m}_t + \frac{1-\alpha_t}{K}] \odot [\bar{\alpha}_{t-1}\hat{\boldsymbol{m}}_0 + \frac{1-\bar{\alpha}_{t-1}}{K}] \\ \pi_\theta(\boldsymbol{x}_t, \boldsymbol{m}_t, t, \boldsymbol{s}_c) &= \frac{\boldsymbol{\theta}(\boldsymbol{m}_t, \hat{\boldsymbol{m}}_0)}{\sum_{k=1}^{K} \boldsymbol{\theta}(\boldsymbol{m}_t, \hat{\boldsymbol{m}}_0)_k} \end{aligned} \tag{7}$$

## 4 Context-aware Dynamic Noising

Let us take a closer look at the standard Gaussian white noise typically used in diffusion models for temporal event sequence modeling. As we have already stated, forward diffusion of event-time intervals $\boldsymbol{x}_t$ are independent of event-type noising. Precisely, forward diffusion of time intervals consists of a step-wise transformation of a sampled inter-arrival sequence $\boldsymbol{x}_0$ to pure Gaussian noise $\boldsymbol{x}_T \sim \mathcal{N}(\boldsymbol{0}, \mathcal{I})$. We start with $\boldsymbol{x}_0$, and at each time-step, make adjustments so that $\boldsymbol{x}_t$ has a mean that gradually moves towards zero and a variance that gradually increases towards unity as $t$ increases. Similarly, the denoising process starts with pure Gaussian noise and as the time step decreases, the pure noise slowly converges to the original time interval distribution.

**Limitations.** The choice of $\beta_t$ is crucial for the correct functioning of the reverse process (Sohl-Dickstein et al., 2015) as both forward and reverse diffusion processes have the same functional form when the values $\beta_t$ are small. In this context, we argue that employing a uniform variance schedule $\beta_t$ across all events may not be optimal. For example, consider events with small time intervals (refer dense regions in Figure 1). These events will be more sensitive to the variance schedule and a higher variance would make it difficult to learn the denoising process for such events. Similarly, for less

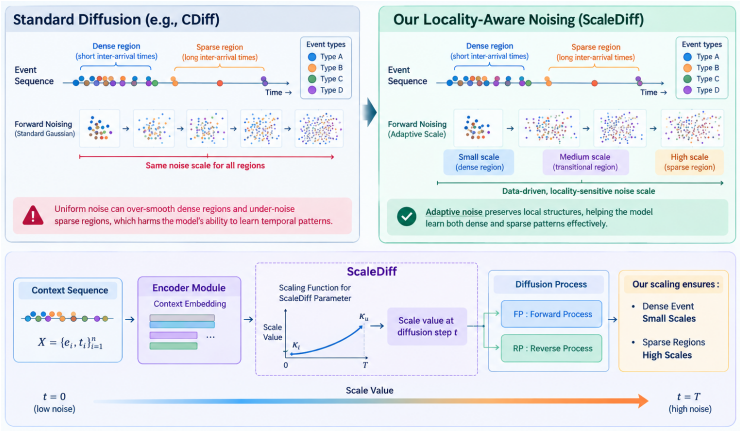



Figure 1: Overview of SCALEDIFF. Events in dense regions receive smaller scales, while events in high region receive higher scales. Adaptive noise allows to preserve local structure better, helping model to learn both sparse and dense regions effectively.

frequent events (i.e., with higher time intervals) (refer sparse regions in Figure 1) a lower variance would not be conducive for the reverse diffusion process.

More precisely, the variance of individual event time intervals ($\mathrm{Var}(x_0^i)$) is often large in temporal event data. In fact, we observe that real-world applications often consist of several event types with varying time intervals (e.g., STACKOVERFLOW consist of 22 event types with average time intervals ranging from $10^{-4}$ to 20.33 time units with an average time interval of 0.95 time units; each time unit is 11 days).

As shown later in Eq. 2, the noisy sequence $\boldsymbol{x}_t$ can be written as $\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{(1-\bar{\alpha}_t)}\epsilon$ where $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathcal{I})$, implying event-wise $\mathrm{SNR}_i(t) = \mathrm{Var}(\sqrt{\bar{\alpha}_t}x_{0,i})/\mathrm{Var}(\sqrt{1-\bar{\alpha}_t}\epsilon_i)$. Within a sequence, sparse events typically exhibit higher variance than dense events. Consequently, for sparse events, signals persist longer under diffusion, whereas for low-variance dense events, signals vanish early. This implies that high-variance regions remain informative longer into the diffusion process, while low-variance regions rapidly degrade into pure noise. Such imbalance may adversely affect the learning of the reverse process: the model tends to reconstruct high-variance regions more accurately while struggling in dense, low-variance zones, leading to suboptimal estimation of $\epsilon_\theta$ (cf. Eq. 6). Consequently, we posit that context-adaptive variance scaling at early diffusion steps is beneficial for stabilizing training and improving the learning of the reverse diffusion dynamics.

That Gaussian white noise may not always be optimal, have been explored in vision diffusion models. Examples include non-isotropic noise (Voleti et al., 2022), time-varying blue noise or correlated noise (Huang et al., 2024). These alternatives have been shown to achieve superior performance.

### 4.1 PROPOSED METHODOLOGY

Motivated by the recent developments in vision, we propose a novel forward noising method, whereby the variance of noise is decided on a per-event basis in a data-dependent manner. Specifically, the variance of noise is decided dynamically based on the relative inter-arrival time of that event. Furthermore, as training progresses, the time interval gaps between the different events slowly converge, and the requirement for different variance also diminishes. Therefore, we employ a time-varying noise, where the Gaussian noise with non-uniform variances slowly converges to Gaussian noise with uniform variances as the diffusion step increases.

We propose to modify the forward diffusion in Eq. 2 as :

$$q(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0, (1-\bar{\alpha}_t)\mathcal{S}_t^2\mathcal{I})) \tag{8}$$

$$\mathcal{S}_t = \sqrt{(1-\gamma_t)}\mathcal{S}_{t-1} + \sqrt{\gamma_t}\mathcal{I} \tag{9}$$

$$\mathcal{S}_t = \sqrt{\bar{\alpha}_t}\mathcal{S}_0 + \sqrt{(1-\bar{\alpha}_t)}\mathcal{I} \tag{10}$$

where $\mathcal{S}_t$ is a diagonal matrix with its diagonal entries containing customized scales for scaling variance for diffusion time step $t$ for the corresponding event and $\gamma_t$ is a scheduler similar to $\beta_t$. $\mathcal{S}_t = \text{diag}(s_1, .., s_N)$, where $s_i$ is the scaling factor for $i$-th event $e_i$ at diffusion time step $t$. The process for obtaining the scale values is explained in Section 4.2.

The diffusion process starts with an initial scale matrix $\mathcal{S}_0$, which contains scales customized according to the temporal context of each event. As the diffusion step $t$ increases, the scale matrix $\mathcal{S}_t$ slowly converges to the identity matrix $\mathcal{I}$, leading to the scaled noise converging to standard Gaussian noise with identical variance $\mathcal{I}$. In intermediate steps, Gaussian noise with identical variance $\mathcal{I}$ and Gaussian noise with scaled variance $\mathcal{S}_0$ are interpolated based on the time step $t$. The smoothness of the interpolation depends on the selection of the $\gamma_t$ function. For $\gamma_t$, we choose one similar to the beta scheduler.

Consequently, $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$ reduces to

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{\alpha_t}\boldsymbol{x}_{t-1}, \Sigma_t) \quad \text{where} \quad \Sigma_t = (1-\bar{\alpha}_t)\mathcal{S}_t^2 - \alpha_t(1-\bar{\alpha}_{t-1})\mathcal{S}_{t-1}^2 \tag{11}$$

Assuming $\mathcal{S}_t$ follows same $\beta$ scheduler as $\boldsymbol{x}_t$, we can show that $\Sigma_t \succeq 0 \quad \forall t$. (Refer Appendix A).

We now derive the corresponding posterior $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$, which, from Bayes Theorem, can be expressed as:

$$q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) = q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_0)\frac{q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_0)}{q(\boldsymbol{x}_t \mid \boldsymbol{x}_0)} \tag{12}$$

$$\propto \exp\left[-\frac{1}{2}\left((\boldsymbol{x}_t - \sqrt{\alpha_t}\boldsymbol{x}_{t-1})^T\Sigma_t^{-1}(\boldsymbol{x}_t - \sqrt{\alpha_t}\boldsymbol{x}_{t-1})\right.\right.$$
$$\left.\left. + \frac{1}{(1-\bar{\alpha}_{t-1})}(\boldsymbol{x}_{t-1} - \sqrt{\bar{\alpha}_{t-1}}\boldsymbol{x}_0)^T\Sigma_{t-1}^{-1}(\boldsymbol{x}_{t-1} - \sqrt{\bar{\alpha}_{t-1}}\boldsymbol{x}_0)^T - C(x_0, x_t)\right)\right] \tag{13}$$

$$= \exp\left[-\frac{1}{2}\left(\boldsymbol{x}_{t-1}^T\left(\alpha_t\Sigma_t^{-1} + \frac{1}{(1-\bar{\alpha}_{t-1})}\mathcal{S}_{t-1}^{-2}\right)\boldsymbol{x}_{t-1}\right.\right.$$
$$\left.\left. -2\boldsymbol{x}_{t-1}^T\left(\sqrt{\alpha}_t\Sigma_t^{-1}\boldsymbol{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}}{(1-\bar{\alpha}_{t-1})}\mathcal{S}_{t-1}^{-2}\boldsymbol{x}_0\right) - C(\boldsymbol{x}_t, \boldsymbol{x}_0)\right)\right] \tag{14}$$

Here $C(x_t, x_0)$ is some function that does not involve $x_{t-1}$, and thereby details are omitted. Assuming $q(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1}; \tilde{\boldsymbol{\mu}}_t(x_t, x_0), \tilde{\boldsymbol{\Sigma}}_t)$. We can show from Eq 40 (Appendix A):

$$\tilde{\boldsymbol{\Sigma}}_t^{-1} = \left(\alpha_t\Sigma_t^{-1} + \frac{1}{1-\bar{\alpha}_{t-1}}\mathcal{S}_{t-1}^{-2}\right) \implies \tilde{\boldsymbol{\Sigma}}_t = \frac{(1-\bar{\alpha}_{t-1})}{(1-\bar{\alpha}_t)}\mathcal{S}_t^{-2}\mathcal{S}_{t-1}^2\Sigma_t \tag{15}$$

Similarly, $\tilde{\boldsymbol{\mu}}_t(x_t, x_0)$ can be written as:

$$\tilde{\boldsymbol{\mu}}_t = \frac{1}{\sqrt{\alpha_t}}(\boldsymbol{x}_t - \frac{1}{\sqrt{1-\bar{\alpha}_t}}\Sigma_t\mathcal{S}_t^{-1}\boldsymbol{\epsilon}) \tag{16}$$

See Appendix A for the complete derivation. The reverse diffusion process is denoted by $p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_c) = \mathcal{N}(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_\theta(x_t, t, \boldsymbol{x}_c), \boldsymbol{\Sigma}_\theta(x_t, t, \boldsymbol{x}_c))$ where $\boldsymbol{\mu}_\theta(x_t, t, \boldsymbol{x}_c)$ mimics $\tilde{\boldsymbol{\mu}}_t$ and $\boldsymbol{\Sigma}_\theta(x_t, t, \boldsymbol{x}_c)$ mimics $\tilde{\boldsymbol{\Sigma}}_t$. Given $(\boldsymbol{x}_t, t, \boldsymbol{x}_c)$, we learn the initial scales $\mathcal{S}_0(\boldsymbol{x}_t, t, \boldsymbol{x}_c)$ for a given context, from which we can derive $\mathcal{S}_t \quad \forall t$. Given $\mathcal{S}_t$, $\tilde{\boldsymbol{\Sigma}}_t$ can be exactly computed. According to standard practices, we employ $\boldsymbol{\epsilon}_\theta(x_t, t, \boldsymbol{x}_c)$ module to learn the Gaussian noise and retrieve $\boldsymbol{\mu}_\theta(x_t, t, \boldsymbol{x}_c)$ as

$$\boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t, \boldsymbol{x}_c) = \frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{x}_t - \frac{1}{\sqrt{1-\bar{\alpha}_t}}\Sigma_t\mathcal{S}_t^{-1}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t, \boldsymbol{x}_c)\right) \tag{17}$$

Refer to Algorithms 1 and 2 for complete training and sampling procedures. In summary, our proposed method modifies the forward diffusion and the inference, while keeping the loss function intact. Following standard norms, one diffusion timestep term per sample is used, which is selected by uniform sampling $t \sim \mathcal{U}(0, T)$. Next, we describe the scale learning process in detail.

### 4.2 Scale Learning

While locality-aware customized scales can be directly computed from the target sequence $\boldsymbol{x}_0$ during training, this approach is not applicable during inference, since the target sequence is unavailable

**Algorithm 1** Training

**repeat**
  $\boldsymbol{x}_0 \sim q(\boldsymbol{x}_0)$
  $t \sim \text{Uniform}(1, .., T)$
  $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathcal{I})$
  $\mathcal{S}_0 = f^{\mathcal{S}}_{\theta_S}(\boldsymbol{x}_c, \boldsymbol{x}_0, 0)$
  $\mathcal{S}_t$ can be obtained from $\mathcal{S}_0$ using Eq 10
  $\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{(1-\bar{\alpha})}\mathcal{S}_t\boldsymbol{\epsilon}$
  Take gradient descent step on $\nabla_\theta \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t, \boldsymbol{x}_c)\|^2$
**until** Convergence

**Algorithm 2** Sampling

$\boldsymbol{x}_T \sim \mathcal{N}(\mathbf{0}, \mathcal{I})$
**for** t = T,..,1 **do**
  $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \mathcal{I})$
  $\mathcal{S}_0 = f^{\mathcal{S}}_{\theta_S}(\boldsymbol{x}_c, \boldsymbol{x}_t, t)$
  $\mathcal{S}_t, \mathcal{S}_{t-1}$ can be obtained from $\mathcal{S}_0$ using Eq 10
  $\Sigma_t, \tilde{\boldsymbol{\Sigma}}_t$ can be found using Eq 11
  $\boldsymbol{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{x}_t - \frac{1}{\sqrt{1-\bar{\alpha}_t}}\mathcal{S}_t^{-1}\Sigma_t\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t, \boldsymbol{x}_c)\right) + \tilde{\boldsymbol{\Sigma}}_t^{\frac{1}{2}}\boldsymbol{z}$
**end for**
**return** $\boldsymbol{x}_0$

at test time. For computing target scale matrix $\mathcal{S}_0$ for a given input $(x_c, x_t)$, we employ a learnable network $f^{\mathcal{S}}_{\theta_s}$, which takes as input (1) event context $x_c$, (2) noisy event target $x_t$, (3) diffusion time step $t$ and generates a corresponding scale matrix $\mathcal{S}_0$. Note that, irrespective of diffusion time step $t$, the module always generates a scale matrix $\mathcal{S}_0$, i.e., scale matrix for time step 0. We can compute $\mathcal{S}_t$ for any timestep $t$ using Eq 10. During both the forward diffusion and reverse sampling processes, we employ $f^{\mathcal{S}}_{\theta_s}$ to estimate the scales $\mathcal{S}_0$. However, in the forward diffusion phase, we can alternatively use the ground-truth ($\mathcal{S}_0^{\text{target}}$), since it is readily available during training. Empirically, we observe that these two choices for $\mathcal{S}_0$ in the forward diffusion step yield comparable test performance (refer Appendix Q). For training the learning network $f^{\mathcal{S}}_{\theta_s}$, we adopt a supervised approach, where we minimize the squared error between predicted scale $f^{\mathcal{S}}_{\theta_s}(x_c, x_t, t)$ and target scales $\mathcal{S}_0^{true}(x_t)$. Refer Appendix B for corresponding error bound.

$$\theta_s^* = \arg\min_{\theta_s} \sum_{(x_c, x_0, x_t, t) \in D_{\text{train}}} ||f^{\mathcal{S}}_{\theta_s}(x_c, x_t, t) - \mathcal{S}_0^{target}(x_0)||^2 \tag{18}$$

**Computing target scales:** We compute target scale matrix $\mathcal{S}_0$ as follows. Event sequences generally contain bursts of dense and sparse regions, which indicates that target noise scales for events could be constructed in a locality-sensitive manner. To this end, the sequence of original event inter-arrival times $x_0$ is first normalized by the maximum value in the sequence, mapping it to the range $[0, 1]$, and then linearly scaled down to the pre-specified range $[\kappa_l, \kappa_u]$. These scaled inter-arrival times act as the target scale factors for corresponding events, which is eventually employed for training the noise scale predictor. Let $x_l = \min_i \boldsymbol{x}_{0,i}$ and $x_u = \max_j \boldsymbol{x}_{0,j}$ be the minimum and maximum inter-arrival times in a target sequence $x_0$. (Refer Appendix Q for different choices of scale heuristics, namely (i) global min-max instead of sequence-wise min-max, and (ii) quantile-based scaling.) The target scale matrix $\mathcal{S}_0^{target}(x_0)$ can be computed as

$$\mathcal{S}_0^{target}(x_0) = \text{diag}\left(\frac{x_e - x_l}{(\kappa_u - \kappa_l)}(\boldsymbol{x}_0 - x_l \cdot \mathbf{1}) + \kappa_l\right) \tag{19}$$

$\kappa_u$ and $\kappa_l$ are the pre-defined range of scale values, determined through cross-validation. The linear scaling based on the inter-arrival times ensures a small scale factor for dense regions and no additional scaling for sparse regions ($\mathcal{S}_{0,ii}^{target}(x_0) \approx 1$). While applying scales, we directly employ target scales $\mathcal{S}_0^{\text{target}}$ to justify our opted heuristics. Refer Appendix G for detailed architecture.

### 4.3 Theoretical properties

Although we modify the forward noising schedule by introducing a bounded scaling factor, our framework retains the core structure of DDPM and, consequently, its theoretical guarantees. To substantiate this claim, we establish the following properties of the proposed scaled diffusion process.

**Property 1: Bounded variance.** Let (i) $0 \leq \mathcal{S}_0 \leq 1$, (ii) $0 \leq \bar{\alpha}_t \leq 1$ for all $t$, as satisfied by standard $\beta$-schedules, and (iii) $\mathcal{S}_t$ is diagonal, corresponding to independent event-wise noise variances. Then, the forward noising process has bounded event-wise variance.

*Proof sketch:* With $\mathcal{S}_0 \in [0, 1]$ and $\gamma_t \in [0, 1]$ and $\gamma_t$ follows $\beta$-scheduler, $\text{Var}(\boldsymbol{x}_t|\boldsymbol{x}_0) = (1 - \bar{\alpha}_t)\mathcal{S}_t^2 = (1 - \bar{\alpha}_t)(\sqrt{\bar{\alpha}_t}\mathcal{S}_0 + \sqrt{1 - \bar{\alpha}_t})^2 \leq (1 - \bar{\alpha}_t)\bar{\alpha}_t\mathcal{S}_0^2 \leq \mathcal{S}_0^2$. This ensures numerical stability, preventing variance collapse or explosion, and facilitates smooth learning of the noise predictor $\boldsymbol{\epsilon}_\theta$.

**Property 2: Convergence to uniform Gaussian.** The forward marginal $p(\boldsymbol{x}_T)$ remains Gaussian with diagonal covariance. It preserves analytical tractability.

*Proof sketch:* $\gamma$ scheduler follows $\beta$ scheduler. It implies that as $t \to T$, $\bar{\alpha}_t \to 0$. As $\bar{\boldsymbol{\epsilon}}_t \to \mathcal{S}_t\boldsymbol{\epsilon}$ and $\mathcal{S}_t = (\sqrt{\bar{\alpha}_t}\mathcal{S}_0 + \sqrt{1-\bar{\alpha}_t}\mathcal{I})$, $\bar{\boldsymbol{\epsilon}}_t \to \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathcal{I})$. Note that, the convergence from the proposed scaled Gaussian to a standard Gaussian is governed by the proposed $\gamma$-scheduler, which progressively reduces the influence of the scale term throughout the diffusion process. As $t$ increases, $\gamma$ gradually decreases the contribution of the scale, causing the forward process a smooth transition.

**Property 3: Variance regularization.** For a sufficiently low $\gamma_t > 0$ and $\gamma_t << 1$, $\mathcal{S}_{0,ii} < \mathcal{S}_{0,jj} \implies Var(\boldsymbol{x}_{t,i}|\boldsymbol{x}_{0,i}) < Var(\boldsymbol{x}_{t,j}|\boldsymbol{x}_{0,j})$

*Proof sketch:* Trivially follows from Property 1. This implies that in the earlier diffusion steps, noise scaling is based on the frequencey of occurrence. More frequent events are noised slowly and less fequent events are noised at a faster rate, validating our initial claim. This leads to a monotonic frequency regularization.

While Properties 1 and 2 guarantee stability and tractability of our adaptive noising strategy by enforcing bounded variance and convergence to a standard Gaussian, Property 3 shows that the proposed noise-scaling scheme functions purely as a regularizer within the diffusion process.

In summary, SCALEDIFF introduces: (1) a data-dependent scaling mechanism to adapt the noise magnitude to event characteristics, (2) a dedicated scale prediction module, as true scale is unavailable during inference, (3) a modified reverse diffusion to incorporate the estimated scale, (4) $\gamma$-scheduler to transform the scaled Gaussian into standard Gaussian during noising.

# 5 EXPERIMENTS

In this section, we provide a comprehensive evaluation of our proposed diffusion methods on a diverse set of real datasets. We measure the utility of our methods in terms of their impact on forecasting performance. The results presented in this section include (1) detailed performance comparison with competitive baselines on long-horizon forecasting, (2) performance on shorter horizons and, (3) qualitative analysis to demonstrate the superiority of the proposed noising method.

## 5.1 EXPERIMENTAL SETUP

Our experimental setup closely follows (Zeng et al., 2023) in terms of datasets, baselines, evaluation metrics, and prediction setup to allow fair comparison.

**Datasets.** We employ the five real-world datasets for evaluation, which include: 1) **Taxi** (Whong, 2014) (traffic), 2) **StackOverflow** (Leskovec & Krevl, 2014) (online platform), 3) **Retweet** (Zhou et al., 2013) (social media), 4) AMAZON (Ni et al., 2019) (online transaction), and 5) **Taobao** (Zhu et al., 2018) (clickbait). Refer Appendix I for details. The train/test split follows (Zeng et al., 2023).

**Baselines.** The baselines include Neural Hawkes Process (NHP) (Mei & Eisner, 2017), Attentive Neural Hawkes Process (AttNHP) (Yang et al., 2022), Log-Normal Mixture Model (LNM) (Shchur et al., 2020), Temporal Conditional Diffusion Denoising Model (TCDDM) (Lin et al., 2022), DualTPP (Deshpande et al., 2021), HYPRO (Xue et al., 2022),NJDTPP (Zhang et al., 2024), ResidualTPP (Yuan & Fang, 2025) and CDIFF (Zeng et al., 2023). Refer Appendix J for details. We adopt all baselines from CDIFF, as same train-test split is used. In addition, we reproduce ResidualTPP and NJDTPP from their codebase using the same train-test split of CDiff.

**Evaluation metrics.** To evaluate the quality of generated event sequences, we adopt the four primary metrics used in the existing literature: 1) Optimal Transport Distance (OTD) to measure the edit distance between predicted sequence and ground truth, 2) $\text{RMSE}_{Type}$ that measures Root Mean Squared Error between predicted and true event type distribution, and 3) standard $\text{RMSE}_{Time}$ and 4) Symmetric Mean Absolute Percentage Error (SMAPE) for measuring event time prediction error. Additional details on the implementation and metrics are provided in the Appendix F and Appendix H.

**Prediction setup.** For each sequence, the last $N$ events are set as $\boldsymbol{s}_u$ and the earlier events are considered as context $\boldsymbol{s}_c$. Test results averaged across 5 seeds are reported. Diffusion steps $T$ is set to 200. Box-Cox transformation and hyperparameter configuration follow (Zeng et al., 2023).

Table 1: Long-horizon performance comparison across for models across all datasets on which the model is trained. **bold:** best, underlined: second best.

| Model | TAXI | | | | TAOBAO | | | |
|---|---|---|---|---|---|---|---|---|
| | OTD | $\text{RMSE}_{Type}$ | $\text{RMSE}_{Time}$ | sMAPE | OTD | $\text{RMSE}_{Type}$ | $\text{RMSE}_{Time}$ | sMAPE |
| HYPRO | 21.653±0.163 | 1.231±0.015 | 0.372±0.004 | 93.803±0.454 | 44.336±0.127 | 2.710±0.021 | 0.594±0.030 | 134.922±0.473 |
| Dual-TPP | 24.483±0.383 | 1.353±0.037 | 0.402±0.006 | 95.211±0.187 | 47.324±0.541 | 3.237±0.049 | 0.871±0.005 | 141.687±0.431 |
| Attnhp | 24.762±0.217 | 1.276±0.015 | 0.430±0.003 | 97.388±0.381 | 45.555±0.345 | 2.737±0.021 | 0.708±0.010 | 134.582±0.920 |
| NHP | 25.114±0.268 | 1.297±0.019 | 0.399±0.040 | 96.459±0.521 | 48.131±0.297 | 3.355±0.030 | 0.837±0.009 | 137.644±0.764 |
| LNM | 24.053±0.609 | 1.364±0.032 | 0.384±0.005 | 95.719±0.779 | 45.757±0.287 | 3.193±0.043 | 0.575±0.012 | 127.436±0.606 |
| TCDDM | 22.148±0.529 | 1.309±0.030 | 0.382±0.019 | 90.596±0.574 | 45.563±0.889 | 2.850±0.058 | 0.569±0.015 | 126.512±0.491 |
| NJDTPP | 20.975±0.065 | **0.788±0.007** | 0.278±0.001 | 77.183±3.944 | 46.966±9.046 | 3.710 ± 0.936 | 0.507± 0.001 | 199.588±0.467 |
| ResidualTPP | 30.850±3.090 | 1.228±0.234 | 0.381±0.025 | 95.302±4.016 | 46.810 ± 0.372 | 2.871±0.046 | 0.641±0.041 | 134.543±2.593 |
| CDIFF | 21.013±0.158 | 1.131±0.017 | 0.351±0.004 | 87.993±0.178 | 44.621±0.139 | 2.653±0.022 | 0.551±0.002 | **125.685±0.151** |
| SCALEDIFF | **18.767 ± 0.374 (+10.52%)** | 1.013 ± 0.056 | **0.275 ± 0.003 (+1.07%)** | **76.631 ± 0.814 (+0.07%)** | **42.081 ± 0.349 (+5.69%)** | **2.443 ± 0.060 (+7.91%)** | **0.405 ± 0.002 (+20.11%)** | 139.708 ± 1.712 |

| Model | AMAZON | | | | STACKOVERFLOW | | | |
|---|---|---|---|---|---|---|---|---|
| | OTD | $\text{RMSE}_{Type}$ | $\text{RMSE}_{Time}$ | sMAPE | OTD | $\text{RMSE}_{Type}$ | $\text{RMSE}_{Time}$ | sMAPE |
| HYPRO | 38.613±0.536 | **2.007±0.054** | 0.477±0.010 | 82.506 ± 0.84 | 42.359±0.170 | 1.140±0.014 | 1.554±0.010 | 110.988±0.559 |
| DualTPP | 42.646 ± 0.752 | 2.562 ± 0.202 | 0.482 ± 0.012 | 86.453 ± 2.044 | 41.752 ± 0.200 | **1.134 ± 0.019** | 1.514 ± 0.017 | 117.582 ± 0.420 |
| AttNHP | 39.480 ± 0.326 | 2.166 ± 0.026 | 0.476 ± 0.033 | 84.323 ± 1.815 | 42.591 ± 0.408 | 1.142 ± 0.011 | 1.340 ± 0.006 | 108.542 ± 0.531 |
| NHP | 42.571 ± 0.293 | 2.561 ± 0.060 | 0.519 ± 0.023 | 92.053 ± 1.553 | 43.791 ± 0.147 | 1.244 ± 0.030 | 1.487 ± 0.004 | 116.952 ± 0.404 |
| LNM | 43.820 ± 0.232 | 3.050 ± 0.286 | 0.481 ± 0.145 | 90.910 ± 1.611 | 46.280 ± 0.892 | 1.447 ± 0.057 | 1.669 ± 0.005 | 115.122 ± 0.627 |
| TCDDM | 42.245 ± 0.174 | 2.998 ± 0.115 | 0.476 ± 0.111 | 83.826 ± 1.508 | 42.128 ± 0.591 | 1.467 ± 0.014 | 1.315 ± 0.004 | 107.659 ± 0.934 |
| NJDTPP | 48.272±1.406 | 3.638±0.047 | 0.349±0.019 | 89.847±7.681 | 53.604 ± 1.023 | 2.813±0.045 | 1.081±0.018 | 102.477±3.619 |
| ResidualTPP | 44.548±0.396 | 3.389±0.284 | 0.473±0.001 | 87.926±3.107 | 51.361±3.039 | 2.619±0.200 | 1.497±0.914 | 116.275±9.562 |
| CDIFF | 37.728 ± 0.199 | 2.091 ± 0.163 | 0.464 ± 0.086 | 81.987 ± 1.905 | 41.245 ± 1.400 | 1.141 ± 0.007 | 1.199 ± 0.006 | 106.175± 0.340 |
| SCALEDIFF | **37.345 ± 0.228 (+1.01%)** | 2.155 ± 0.020 | **0.344 ± 0.002 (+1.43%)** | **78.203 ± 0.672 (+4.61%)** | **38.918 ± 0.556 (+5.64%)** | 1.501 ± 0.057 | **0.983 ± 0.012 (+9.06%)** | **86.492 ± 0.185 (+15.59%)** |

| Model | RETWEET | | | |
|---|---|---|---|---|
| | OTD | $\text{RMSE}_{Type}$ | $\text{RMSE}_{Time}$ | sMAPE |
| HYPRO | 61.031 ± 0.092 | 2.623 ± 0.036 | 30.100 ± 0.413 | 106.110 ± 1.505 |
| DualTPP | 61.095 ± 0.101 | 2.679 ± 0.026 | 28.914 ± 0.300 | 106.900 ± 1.293 |
| AttNHP | 60.634 ± 0.097 | 2.561 ± 0.054 | 28.812 ± 0.272 | 107.234 ± 1.293 |
| NHP | 60.953 ± 0.079 | 2.651 ± 0.045 | 27.130 ± 0.224 | 107.075 ± 1.398 |
| LNM | 61.715 ± 0.152 | 2.776 ± 0.043 | 27.582 ± 0.191 | 106.711 ± 1.615 |
| TCDDM | 60.501 ± 0.087 | 2.387 ± 0.050 | 27.303 ± 0.152 | 106.048 ± 0.610 |
| NJDTPP | 60.173±0.452 | 6.612±0.095 | 24.063±1.707 | 101.895±4.764 |
| ResidualTPP | 61.496±0.582 | 6.458±0.199 | 32.723±3.881 | 99.954±0.963 |
| CDIFF | 60.661 ± 0.101 | **2.293 ± 0.034** | 27.101 ± 0.113 | 106.184 ± 1.121 |
| SCALEDIFF | **58.832 ± 0.278 (+2.76%)** | 2.901 ± 0.060 | **21.074 ± 0.054 (+12.42%)** | **85.252 ± 0.349 (+14.70%)** |

Appendix N justifies use of box-cox transformation. We train for 500 epochs and report the model that performs best on the validation set. The scale parameters $\kappa_l^t, \kappa_l^m$ are selected from $[0.4, 0.6, 0.8]$, and linear $\gamma$-scheduler is used.

### 5.2 LONG-HORIZON FORECASTING

We compare the $N$ event forecasting performance of SCALEDIFF with the baselines for the five real-world datasets on four standard metrics, measuring both time and type prediction performance. Table 1 reports the performance averaged across five seeds, where the metrics for the baselines are reported from Zeng et al. (Zeng et al., 2023) (refer Appendix O for comparison with ADD-and-THIN Lüdke et al. (2024)). SCALEDIFF demonstrate consistent improvements in most of the metrics, indicating more accurate modeling of inter-event time-interval distributions, by aligning noise injection with local structural regularity. The margin of improvement (up to 7.91% in $\text{RMSE}_{Type}$, and most significantly 20.11% in $\text{RMSE}_{Time}$) validates the utility of structure-aware noise customization.

Regarding learning the type distribution, SCALEDIFF improve over the baselines for TAXI and TAOBAO, whereas provide comparable or slightly inferior performance for RETWEET, STACKOVERFLOW, and AMAZON. It further indicates that our proposed noise schemes facilitate better event type distribution learning, specifically for skewed or low entropy event type distribution. Refer Appendix V for detail error analysis on STACKOVERFLOW.

Among the baselines, we observe the classic neural methods, namely NHP, or ATTNHP, which were primarily deployed for next-event prediction, perform poorly in current long-horizon forecasting, potentially due to error accumulation in autoregressive inference. Diffusion-based models, namely HYPRO, TCDDM, ResidualTPP and NJDTPP achieve prediction performance close to CDIFF, with NJDTPP and CDIFF often surpassing the rest. However, their relatively poor performance for STACKOVERFLOW shows their inability to capture complex temporal dynamics for long-horizon scenario. CDIFF improves over rest of the baselines, by jointly modeling time and type as well as noising the entire sequence, clearly giving it advantage in long-horizon prediction. To summarize, injecting semantic or structural awareness into the forward diffusion process offers tangible benefits

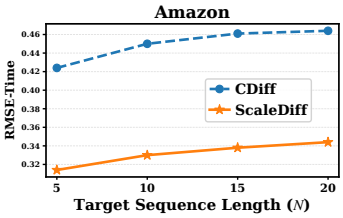



Figure 2: $\text{RMSE}_{Time}$ for shorter horizons.

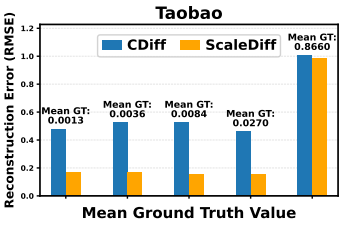



Figure 3: Reconstruction Loss in sparse and dense regions

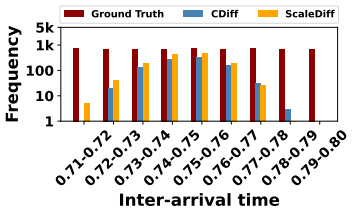



Figure 4: Histogram of inter-arrival time at AMAZON.

in temporal modeling. Our results suggest that temporal proximity-aware scaling strikes the most favorable balance — enhancing temporal fidelity without significantly compromising event-type accuracy. A ranking of the methods based on performance across all the datasets show the superiority of SCALEDIFF for all the time related metrics over the existing methods (further details in Appendix K).

### 5.3 PREDICTING SHORTER HORIZONS

Figure 2 compare SCALEDIFF with CDIFF for predicting shorter horizon for window size $[5, 10, 15, 20]$. While performance degrades with increase in forecasting window, SCALEDIFF maintain superior performance over CDIFF across the entire range of forecasting window with the performance gap being particularly significant for larger window size (window size 20), indicating that the advantage of structure-aware temporal noising is more prominent while generating longer sequences. Refer to Appendix L for similar results on the other datasets.

**Additional experiments.** Figure 3 analyzes **reconstruction error** across inter-arrival-time bins, from dense to sparse regions, which explicitly indicates lower reconstruction error in dense regions in SCALEDIFF over CDIFF (refer Appendix T). Appendix P analyze the behavior of ScaleDiff under controlled temporal dynamics – **synthetic Hawkes and Poisson** event streams, with SCALEDIFF having additional benefits on Hawkes dataset having explicit clusters. Appendix O compares SCALEDIFF with ADD-AND-THIN Lüdke et al. (2024) (which only predicts time but not type), showing it to underperform consistently compared to SCALEDIFF. Appendix S reports **inference and training cost** of SCALEDIFF. Further, we explore **sensitivity** of our models to choices of scaling bounds $\kappa_l$ in Appendix U advocating use of hyper-parameter-tuning and impact of $\gamma$-scheduler in Appendix Q. Appendix Q reports for detailed **ablations** for various choice of scales. Appendix R analyzes **robustness** of SCALEDIFF in two setting – (i) by removing partial history and (ii) perturbing the predicted scales during inference, and we find SCALEDIFF as reasonably robust in both settings.

### 5.4 QUALITATIVE ANALYSIS

Figure 4 compares the histogram of predicted inter-arrival time distribution of SCALEDIFF with CDIFF at selected dense regions of the distribution, allowing us to investigate model alignment with ground truth at a finer granularity. Evidently, SCALEDIFF achieves closer alignment with ground truth inter-arrival time distribution than CDIFF across a wide range of values. Moreover, in Appendix M we compare the predicted distributions employing Kernel Density Estimation (KDE), showing SCALEDIFF learns lower entropy/more informative inter-arrival time distributions, indicating more fine-grained learning.

## 6 CONCLUSION

In this work, we introduced a structure-aware noising strategy tailored for diffusion models for temporal event streams, with a specific emphasis on perturbing entire event sequences. Subsequently, we design customized training and sampling strategies that align with the proposed strategy and provide theoretical justification for the underlying design principles. Through comprehensive evaluations on five SOTA real-world datasets and several strong SOTA baselines, we demonstrate the effectiveness of our approach. Our findings underline the promise of using non-white Gaussian noise in enhancing diffusion processes within temporal modeling tasks.

### AI USE STATEMENT

The authors used freely available versions of ChatGPT and Claude solely for manuscript restructuring and prose review. The authors take full responsibility for the methods, experiments, ablation studies, citations, and all other scientific content presented in the manuscript. No AI tools were used to conduct or design the experiments, generate experimental results, or select citations.

### ETHICS STATEMENT

This work addresses the temporal pattern issue by proposing a locality-based, structure-aware noising technique that adaptively determines the noise scale for each event based on temporal patterns in its local neighborhood. The proposed method does not introduce additional ethical concerns beyond those inherent to the underlying data and application domain.

### REPRODUCIBILITY STATEMENT

To facilitate reproducibility, the authors provide the codebase in the supplementary material along with a working example based on one dataset. The implementation includes the components necessary to reproduce the reported methodology and experiments. Upon acceptance, the authors will release the complete codebase and associated resources to further support independent verification and replication of the results.

## REFERENCES

Emmanuel Bacry, Iacopo Mastromatteo, and Jean-François Muzy. Hawkes processes in finance. *Market Microstructure and Liquidity*, 1(01):1550005, 2015.

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

Abir De, Isabel Valera, Niloy Ganguly, Sourangshu Bhattacharya, and Manuel Gomez Rodriguez. Learning and forecasting opinion dynamics in social networks. *Advances in neural information processing systems*, 29, 2016.

Prathamesh Deshpande, Kamlesh Marathe, Abir De, and Sunita Sarawagi. Long horizon forecasting with temporal point processes. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pp. 463–471, 2021.

Nan Du, Hanjun Dai, Rakshit Trivedi, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and Le Song. Recurrent marked temporal point processes: Embedding event history to vector. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1555–1564, 2016.

Xingchang Huang, Corentin Salaün, Cristina Vasconcelos, Christian Theobalt, Cengiz Öztireli, and Gurprit Singh. Blue noise for diffusion models, 2024. URL https://arxiv.org/abs/2402.04930.

Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, 2014. Accessed: 2025-04-19.

Haitao Lin, Lirong Wu, Guojiang Zhao, Pai Liu, and Stan Z Li. Exploring generative neural temporal point process. *arXiv preprint arXiv:2208.01874*, 2022.

David Lüdke, Marin Biloš, Oleksandr Shchur, Marten Lienen, and Stephan Günnemann. Add and thin: Diffusion for temporal point processes, 2024. URL https://arxiv.org/abs/2311.01139.

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

Hongyuan Mei, Guanghui Qin, and Jason Eisner. Imputing missing events in continuous-time event streams, 2019. URL https://arxiv.org/abs/1905.05570.

Jianmo Ni, Jiacheng Li, and Julian McAuley. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 188–197, Hong Kong, China, 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1018. URL https://aclanthology.org/D19-1018/.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pp. 8162–8171. PMLR, 2021.

Severi Rissanen, Markus Heinonen, and Arno Solin. Generative modelling with inverse heat dissipation. *arXiv preprint arXiv:2206.13397*, 2022.

Subham Sekhar Sahoo, Aaron Gokaslan, Christopher De Sa, and Volodymyr Kuleshov. Diffusion models with learned adaptive noise. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=loMa99A4p8.

Oleksandr Shchur, Marin Biloš, and Stephan Günnemann. Intensity-free learning of temporal point processes. *arXiv preprint arXiv:1909.12127*, 2019.

Oleksandr Shchur, Marin Biloš, and Stephan Günnemann. Intensity-free learning of temporal point processes. In *International Conference on Learning Representations*, 2020.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. pmlr, 2015.

Yujee Song, Donghyun Lee, Rui Meng, and Won Hwa Kim. Decoupled marked temporal point process using neural ordinary differential equations. In *International Conference on Learning Representations*, volume 2024, pp. 28158–28175, 2024.

Sarah Touati, Mark Naylor, and Ian G Main. Statistical modeling of the 1997–1998 colfiorito earthquake sequence: Locating a stationary solution within parameter uncertainty. *Bulletin of the Seismological Society of America*, 104(2):885–897, 2014.

John W Tukey. On the comparative anatomy of transformations. *The Annals of Mathematical Statistics*, pp. 602–632, 1957.

Vikram Voleti, Christopher Pal, and Adam Oberman. Score-based denoising diffusion with non-isotropic gaussian noise models. *arXiv preprint arXiv:2210.12254*, 2022.

Chris Whong. Foiling nyc's taxi trip data. https://chriswhong.com/open-data/foil_nyc_taxi/, 2014. Accessed: 2025-04-19.

Shuai Xiao, Junchi Yan, Xiaokang Yang, Hongyuan Zha, and Stephen Chu. Modeling the intensity function of point process via recurrent neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

Siqiao Xue, Xiaoming Shi, James Zhang, and Hongyuan Mei. Hypro: A hybridly normalized probabilistic model for long-horizon prediction of event sequences. *Advances in Neural Information Processing Systems*, 35:34641–34650, 2022.

Chenghao Yang, Hongyuan Mei, and Jason Eisner. Transformer embeddings of irregularly spaced events and their participants. *arXiv preprint arXiv:2201.00044*, 2021.

Chenghao Yang, Hongyuan Mei, and Jason Eisner. Transformer embeddings of irregularly spaced events and their participants. In *International Conference on Learning Representations*, 2022.

Ruoxin Yuan and Guanhua Fang. Residual tpp: A unified lightweight approach for event stream data analysis. In *Forty-second International Conference on Machine Learning*, 2025.

Yuan Yuan, Jingtao Ding, Chenyang Shao, Depeng Jin, and Yong Li. Spatio-temporal diffusion point processes. In *Proceedings of the 29th ACM SIGKDD conference on knowledge discovery and data mining*, pp. 3173–3184, 2023.

Mai Zeng, Florence Regol, and Mark Coates. Interacting diffusion processes for event sequence forecasting. *arXiv preprint arXiv:2310.17800*, 2023.

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

Shuai Zhang, Chuan Zhou, Yang Aron Liu, Peng Zhang, Xixun Lin, and Zhi-Ming Ma. Neural jump-diffusion temporal point processes. In *Forty-first International Conference on Machine Learning*, 2024.

Ke Zhou, Hongyuan Zha, and Le Song. Learning triggering kernels for multi-dimensional hawkes processes. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, pp. 1301–1309, 2013.

Wang-Tao Zhou, Zhao Kang, Ling Tian, Jinchuan Zhang, and Yumeng Liu. Non-autoregressive diffusion-based temporal point processes for continuous-time long-term event prediction. *Expert Systems with Applications*, 267:126210, 2025.

Han Zhu, Xiang Li, Pengye Zhang, Guozheng Li, Jie He, Han Li, and Kun Gai. Learning tree-based deep model for recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery &; Data Mining*, KDD '18, pp. 1079–1088. ACM, July 2018. doi: 10.1145/3219819.3219826. URL http://dx.doi.org/10.1145/3219819.3219826.

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

## A Forward and Reverse Diffusion

Our modified marginal in forward diffusion can be written as:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0, (1-\bar{\alpha})\mathcal{S}_{t-1}^2)) \tag{20}$$

It can be shown that corresponding $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$ reduces to

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{\alpha_t}\boldsymbol{x}_{t-1}, \Sigma_t) \tag{21}$$

where $\Sigma_t$ is a diagonal matrix with $\Sigma_t(i,i) = \sigma_{i,t}^2$, where $\sigma_{i,t}^2 = (1-\bar{\alpha}_t)s_{i,t}^2 - \alpha_t(1-\bar{\alpha}_{t-1})s_{i,t-1}^2$.

We can show it for single event $e_i$ for which noisy variables are $x_0$ to $x_t$ and scales are $s_0$ to $s_t$. The results generalizes to entire sequence $\boldsymbol{x}_0$ as noise variance is inherently diagonal.

$$\begin{aligned}
x_t &= \sqrt{\alpha_t}x_{t-1} + \sigma_t\epsilon && (22)\\
&\implies \text{Var}(x_t|x_0) = \sqrt{\alpha_t}\text{Var}(x_{t-1}|x_0) + \sigma_t^2 && (23)\\
&\implies (1-\bar{\alpha}_t)s_t^2 = \alpha_t(1-\bar{\alpha}_{t-1})s_{t-1}^2 + \sigma_t^2 && (24)\\
&\implies \sigma_t^2 = (1-\bar{\alpha}_t)s_t^2 - \alpha_t(1-\bar{\alpha}_{t-1})s_{t-1}^2 && (25)
\end{aligned}$$

Now, to ensure that it remains Gaussian, we need to show that $\sigma_t^2 \geq 0 \forall t$. For that, we assume $\mathcal{S}_t$ follows the same $\beta$ scheduler as $\boldsymbol{x}_t$.

We need to show that

$$\begin{aligned}
&\sigma_t^2 \geq 0 && (26)\\
&\implies (1-\bar{\alpha}_t)s_t^2 - \alpha_t(1-\bar{\alpha}_{t-1})s_{t-1}^2 \geq 0 && (27)\\
&\implies \frac{s_t^2}{s_{t-1}^2} \geq \frac{\alpha_t(1-\bar{\alpha}_{t-1})}{(1-\bar{\alpha}_t)} && (28)
\end{aligned}$$

Now, we know that

$$s_t \geq \sqrt{1-\gamma_t} s_{t-1} \tag{29}$$

$$\implies \frac{s_t^2}{s_{t-1}^2} \geq (1-\gamma_t) \tag{30}$$

$$\implies \frac{s_t^2}{s_{t-1}^2} \geq (\alpha_t) \quad \text{as we employ same scheduler for } \mathcal{S}_t \text{ and } \boldsymbol{x}_t \tag{31}$$

So, we just need to show that $\alpha_t \geq \frac{\alpha_t(1-\bar{\alpha}_{t-1})}{(1-\bar{\alpha}_t)}$, .

We will prove it by contradiction. Let

$$\alpha_t < \frac{\alpha_t(1-\bar{\alpha}_{t-1})}{(1-\bar{\alpha}_t)} \tag{32}$$

$$\implies 1 < \frac{(1-\bar{\alpha}_{t-1})}{(1-\bar{\alpha}_t)} \tag{33}$$

$$\implies (1-\bar{\alpha}_t) < (1-\bar{\alpha}_{t-1}) \tag{34}$$

$$\implies (\bar{\alpha}_t) > (\bar{\alpha}_{t-1}) \tag{35}$$

$$\implies (\alpha_t) > 1 \quad \text{this can not be true, which concludes the proof.} \tag{36}$$

Now we need to compute the posterior $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$. Now, the covariance matrix of the posterior $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$ is diagonal, we will derive mean and variance of posterior in the upcoming discussion for i-th event $e_i$, which generalizes to entire matrix. Let us assume the event time is denoted by $x$, which is gradually noised from $x_0$ to $x_T$. Its corresponding scales are denoted as $s_t$ and $\sigma_{i,t}^2$ will be refered as $\sigma_t^2$ in subsequent discussions.

Now the posterior $q(x_{t-1}|x_t, x_0)$ can be expressed as follows, using Bayes Theorem:

$$q(x_{t-1}|x_t, x_0) = q(x_t \mid x_{t-1}, x_0)\frac{q(x_{t-1} \mid x_0)}{q(x_t \mid x_0)} \tag{37}$$

$$\propto \exp\left[-\frac{1}{2}\left(\frac{(x_t - \sqrt{\alpha_t}x_{t-1})^2}{\sigma_t^2} + \frac{(x_{t-1} - \sqrt{\bar{\alpha}_{t-1}}x_0)^2}{(1-\bar{\alpha}_{t-1})s_{t-1}^2} - C(x_0, x_t)\right)\right] \tag{38}$$

$$= \exp\left[-\frac{1}{2}\left(\frac{\alpha_t x_{t-1}^2 - 2\sqrt{\alpha_t}x_{t-1}x_t}{\sigma_t^2} + \frac{x_{t-1}^2 - 2\sqrt{\bar{\alpha}_{t-1}}x_{t-1}x_0}{(1-\bar{\alpha}_{t-1})s_{t-1}^2} - C(x_t, x_0)\right)\right] \tag{39}$$

$$= \exp\left[-\frac{1}{2}\left(x_{t-1}^2\left(\frac{\alpha_t}{\sigma_t^2} + \frac{1}{(1-\bar{\alpha}_{t-1})s_{t-1}^2}\right)\right.\right.$$
$$\left.\left. -2x_{t-1}\left(\frac{\sqrt{\alpha}_t x_t}{\sigma_t^2} + \frac{\sqrt{\bar{\alpha}_{t-1}}x_0}{(1-\bar{\alpha}_{t-1})s_{t-1}^2}\right) - C(x_t, x_0)\right)\right] \tag{40}$$

Here $C(x_t, x_0)$ is some function that does not involve $x_{t-1}$, and thereby details are omitted.

Let $q(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1}; \tilde{\boldsymbol{\mu}}_t(x_t, x_0), \tilde{\boldsymbol{\Sigma}}_t)$.

Moreover $(\tilde{\mu}_t, \tilde{\beta}_t)$ are mean and variance of $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_0, \boldsymbol{x}_t)$ for a single $i$-th event.

$$\tilde{\mu}_t = \tilde{\boldsymbol{\mu}}_{t,i}, \quad \tilde{\beta}_t = \tilde{\boldsymbol{\Sigma}}_{t,ii} \tag{41}$$

Now we derive the mean and variance for individual events.

Let suppose $q(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1}; \tilde{\mu}_t(x_t, x_0), \tilde{\beta}_t \mathcal{I})$. Then following the standard Gaussian density function, we can show from Eq 40:

$$\tilde{\beta}_t = \frac{1}{\left(\frac{\alpha_t}{\sigma_t^2} + \frac{1}{(1-\bar{\alpha}_{t-1})s_{t-1}^2}\right)} \tag{42}$$

$$= \frac{\sigma_t^2(1-\bar{\alpha}_{t-1})s_{t-1}^2}{\alpha_t(1-\bar{\alpha}_{t-1})s_{t-1}^2 + \sigma_t^2} \tag{43}$$

$$= \frac{\sigma_t^2(1-\bar{\alpha}_{t-1})s_{t-1}^2}{\alpha_t(1-\bar{\alpha}_{t-1})s_{t-1}^2 + (1-\bar{\alpha}_t)s_t^2 - \alpha_t(1-\bar{\alpha}_{t-1})s_{t-1}^2)} \tag{44}$$

$$= \frac{s_{t-1}^2\sigma_t^2(1-\bar{\alpha}_{t-1})}{(1-\bar{\alpha}_t)s_t^2} \tag{45}$$

Similarly, $\tilde{\mu}_t(x_t, x_0)$ can be written as:

$$\tilde{\mu}_t = \tilde{\beta}_t \left[\frac{\sqrt{\alpha}_t x_t}{\sigma_t^2} + \frac{\sqrt{\bar{\alpha}_{t-1}} x_0}{(1-\bar{\alpha}_{t-1})s_{t-1}^2}\right] \tag{46}$$

We know that, $x_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t - \sqrt{1-\bar{\alpha}_t}s_t\epsilon)$. Using that, we replace $x_0$ in Eq 46 to obtain:

$$\tilde{\mu}_t = \frac{\tilde{\beta}_t\sqrt{\alpha}_t x_t}{\sigma_t^2} + \frac{\tilde{\beta}_t\sqrt{\bar{\alpha}_{t-1}}}{(1-\bar{\alpha}_{t-1})s_{t-1}^2}\frac{1}{\sqrt{\bar{\alpha}_t}}(x_t - \sqrt{1-\bar{\alpha}_t}s_t\epsilon) \tag{47}$$

$$= \left(\frac{\tilde{\beta}_t\sqrt{\alpha}_t}{\sigma_t^2} + \frac{\tilde{\beta}_t\sqrt{\bar{\alpha}_{t-1}}}{(1-\bar{\alpha}_{t-1})s_{t-1}^2}\frac{1}{\sqrt{\bar{\alpha}_t}}\right)x_t - \frac{\tilde{\beta}_t\sqrt{\bar{\alpha}_{t-1}}}{(1-\bar{\alpha}_{t-1})s_{t-1}^2}\frac{\sqrt{1-\bar{\alpha}_t}s_t}{\sqrt{\bar{\alpha}_t}}\epsilon \tag{48}$$

Now, terms associated with $\epsilon$ reduces as follows:

$$\frac{\tilde{\beta}_t\sqrt{\bar{\alpha}_{t-1}}}{(1-\bar{\alpha}_{t-1})s_{t-1}^2}\frac{\sqrt{1-\bar{\alpha}_t}s_t}{\sqrt{\bar{\alpha}_t}}\epsilon \tag{49}$$

$$= \frac{s_{t-1}^2\sigma_t^2(1-\bar{\alpha}_{t-1})}{(1-\bar{\alpha}_t)s_t^2}\frac{\sqrt{\bar{\alpha}_{t-1}}}{(1-\bar{\alpha}_{t-1})s_{t-1}^2}\frac{\sqrt{1-\bar{\alpha}_t}s_t}{\sqrt{\bar{\alpha}_t}}\epsilon \tag{50}$$

$$= \frac{\sigma_t^2}{s_t\sqrt{\alpha_t}\sqrt{1-\bar{\alpha}_t}}\epsilon \tag{51}$$

Similarly,

$$\left(\frac{\tilde{\beta}_t\sqrt{\alpha}_t}{\sigma_t^2} + \frac{\tilde{\beta}_t}{(1-\bar{\alpha}_{t-1})s_{t-1}^2\sqrt{\bar{\alpha}_t}}\right)x_t \tag{52}$$

$$= \left(\frac{\sqrt{\alpha_t}\sigma_t^2(1-\bar{\alpha}_{t-1})s_{t-1}^2}{\sigma_t^2(1-\bar{\alpha}_t)s_t^2} + \frac{\sigma_t^2(1-\bar{\alpha}_{t-1})s_{t-1}^2}{(1-\bar{\alpha}_t)s_t^2(1-\bar{\alpha}_{t-1})s_{t-1}^2\sqrt{\alpha_t}}\right)x_t \tag{53}$$

$$= \left(\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})s_{t-1}^2}{(1-\bar{\alpha}_t)s_t^2} + \frac{\sigma_t^2}{(1-\bar{\alpha}_t)s_t^2\sqrt{\alpha_t}}\right)x_t \tag{54}$$

$$= \left(\frac{\alpha_t(1-\bar{\alpha}_{t-1})s_{t-1}^2 + \sigma_t^2}{(1-\bar{\alpha}_t)s_t^2\sqrt{\alpha_t}}\right)x_t \tag{55}$$

$$= \left(\frac{\alpha_t(1-\bar{\alpha}_{t-1})s_{t-1}^2 + (1-\bar{\alpha}_t)s_t^2 - \alpha_t(1-\bar{\alpha}_{t-1})s_{t-1}^2}{(1-\bar{\alpha}_t)s_t^2\sqrt{\alpha_t}}\right)x_t \tag{56}$$

$$= \left(\frac{1}{\sqrt{\alpha_t}}\right)x_t \tag{57}$$

$$\tag{58}$$

Combining both results, it implies:

$$\tilde{\mu}_t = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\sigma_t^2}{s_t\sqrt{1-\bar{\alpha}_t}}\epsilon) \tag{59}$$

**Proof of Property 1: Bounded variance.** The proof relies on the following assumptions: (i) The learned initial scale is constrained element-wise as $0 \leq S_0 \leq 1$. (ii) The diffusion schedule satisfies $0 \leq \bar{\alpha}t \leq 1$, which holds for standard $\beta$-schedules. (iii) $S_t$ is diagonal (equivalently, the event-wise noise variances are independent).

Proof: Let $i$ th diagonal entry of $S_0$ is $s_i$. Now, Var$(x_{t,i}|x_0) = (1-\bar{\alpha}_t)(\sqrt{\bar{\alpha}_t}s_i + \sqrt{1-\bar{\alpha}_t})^2$. Now $0 \leq s_i \leq 1$. Therefore, $\sqrt{\bar{\alpha}_t}s_i + \sqrt{1-\bar{\alpha}_t} \leq \sqrt{\bar{\alpha}_t} + \sqrt{1-\bar{\alpha}_t}$. Now, $(\sqrt{\bar{\alpha}_t} + \sqrt{1-\bar{\alpha}_t})^2 = 1+2\sqrt{\bar{\alpha}_t(1-\bar{\alpha}_t)}$. As $\sqrt{\bar{\alpha}_t(1-\bar{\alpha}_t)} \leq \frac{1}{2}$, $(\sqrt{\bar{\alpha}_t}+\sqrt{1-\bar{\alpha}_t})^2 = 1+2\sqrt{\bar{\alpha}_t(1-\bar{\alpha}_t)} \leq 1+2.\frac{1}{2} = 2$. Therefore, Var$(x_{t,i}|x_0) \leq (1-\bar{\alpha}_t)(\sqrt{\bar{\alpha}_t} + \sqrt{1-\bar{\alpha}_t})^2 \leq 2(1-\bar{\alpha}_t) \leq 2$. Thus the forward variance remains uniformly bounded, preventing variance explosion.

## B Error due to scale prediction

ScaleDiff introduces an additional learned component through the scale predictor $f_S$. To analysis corresponding error, let us make the following assumption.

Assumption I (Scale prediction consistency). There exists $\delta > 0$ such that for every diffusion step, $|\hat{S}_t - S_t| \leq \delta$.
Assumption II (Diagonality and boundedness of scales). $S_t$ is diagonal with bounded entries away from zero, $0 < m \leq S_{ii} \leq M$.

Using $\hat{S}_t$ gives updated mean and variance of the posterior as follows:

$$\hat{\tilde{\Sigma}}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\hat{S}_t^{-2}\hat{S}_{t-1}^2\Sigma_t.$$

$$\hat{\mu}_\theta = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1}{\sqrt{1-\bar{\alpha}_t}}\Sigma_t\hat{S}_t^{-1}\epsilon_\theta\right).$$

Due to our assumption on diagonality and boundedness of scales $S_t$, it can be shown that $||\hat{\mu}_\theta - \mu_\theta|| \leq C_1\delta + C_2\delta_\epsilon$ and $||\hat{\tilde{\Sigma}}_t - \tilde{\Sigma}_t|| \leq C_3\delta$.

For mean, it follows from the fact that if $0 < m \leq S_{ii} \leq M$, $|\frac{1}{s_i} - \frac{1}{\hat{s}_i}| \leq C\frac{\delta}{m^2}$.

For variance, we proceed as follows.
$\hat{\tilde{\Sigma}}_t - \tilde{\Sigma}_t = c_t\left(\hat{S}_t^{-2}\hat{S}_{t-1}^2 - S_t^{-2}S_{t-1}^2\right)\Sigma_t$.

Now $\hat{S}_t^{-2}\hat{S}_{t-1}^2 - S_t^{-2}S_{t-1}^2 = (\hat{S}_t^{-2} - S_t^{-2})\hat{S}_{t-1}^2 + S_t^{-2}(\hat{S}_{t-1}^2 - S_{t-1}^2)$.

Since the inversion is Lipschitz on $[m, M]$, $||\hat{S}_t^{-2} - S_t^{-2}|| = O(\delta)$ because $\frac{d}{ds}(s^{-2}) = -\frac{2}{s^3}$, which is bounded whenever $s \geq m > 0$. Similarly, $\hat{S}_{t-1}^2 - S_{t-1}^2 = (\hat{S}_{t-1} - S_{t-1})(\hat{S}_{t-1} + S_{t-1}) = O(\delta)$. So, $||\hat{\tilde{\Sigma}}_t - \tilde{\Sigma}_t|| = O(\delta)$.

As as both noise prediction error and scale prediction errors vanish ($\delta_\epsilon \to 0, \quad \delta_S \to 0$), the reverse chain converges to the same data distribution as in DDPM.

## C Challenges in scaling type diffusion

ScaleDiff introduces data-dependent (locality-aware) noise scaling only for the continuous-time diffusion process, while leaving the discrete event-type diffusion unchanged. This design is intentional and is motivated by both theoretical and empirical considerations. Since event timestamps lie in a continuous domain, the forward diffusion naturally admits an event-dependent covariance, preserves the Gaussian formulation while allocating different noise levels to dense and sparse temporal regions.

In contrast, event types are discrete variables and are diffused using the discrete transition process inherited from our backbone. Extending locality-aware variance scaling to discrete diffusion is non-trivial, as there is no continuous covariance parameter analogous to the Gaussian variance that can be adjusted while preserving the underlying categorical transition process. Consequently, ScaleDiff retains the original discrete diffusion formulation for event types.

Empirically, we observe that the dominant source of reconstruction error improvement comes from timestamp prediction, particularly in densely occurring events where fixed-variance Gaussian noise excessively perturbs neighboring timestamps. Event-type prediction, however, exhibits substantially lower variability and is considerably less sensitive to the forward noise schedule.

Finally, the proposed locality-aware scaling still influences event-type prediction through the cross-diffusion reverse process employed by our backbone. Specifically, the reverse model first reconstructs the denoised timestamps, that subsequently influence the type prediction, allowing the benefits of locality-aware noise scaling to propagate to the discrete generation process.

## D Related Work

**Long-horizon forecasting.** In the context of long-horizon forecasting, HYPRO (Xue et al., 2022) is a hybrid normalized probabilistic model with two components: an autoregressive base model that generates prediction proposals, and an energy function that reweights these proposals, assigning higher probabilities to more realistic outcomes. Similarly, DualTPP (Deshpande et al., 2021) consists of an intensity-free MTPP model to capture microscopic event dynamics by modeling the time of future events. Alongside, it models aggregated counts of events in a given time window, thus encapsulating macroscopic event dynamics. Among more recent works, Song et al. (2024) leveraged Neural ODEs to disentangle event influences over continuous time, while Yuan & Fang (2025) combines classical point processes with lightweight neural residual modeling to improve efficiency and prediction accuracy.

Among diffusion-based approaches, (Lüdke et al., 2024) propose a novel diffusion-inspired approach that progressively refines entire event sequences at once from noise. (Zeng et al., 2023) uses a diffusion-based framework where it directly learns the joint probability distribution of types and inter-arrival times for the events. The model is composed of two distinct diffusion processes for the time intervals and the event types that interact through their respective denoising functions, allowing the model to learn complex interactions.

Recent advances in TPP-based diffusion models include DLTPP Zhou et al. (2025), which introduces non-autoregressive long-horizon forecasting by jointly denoising complete future event sequences, and DSTPP Yuan et al. (2023), that extends diffusion models to spatio-temporal point process through a spatio-temporal co-attention mechanism. More recently, Neural Jump-Diffusion TPP (NJDTPP) Zhang et al. (2024) formulates TPP intensity evolution as a neural jump-diffusion stochastic differential equation, providing a fundamentally different continuous-time intensity-based perspective.

Importantly, while the existing diffusion-based TPP models (e.g., Zhou et al. (2025) and Zeng et al. (2023)) employ a standard forward diffusion process with a globally fixed noise schedule, treating all events identically during corruption, ScaleDiff introduces a locality-aware adaptive forward noising process, where the diffusion scale is customized for each event according to its local temporal structure. This preserves informative neighborhood patterns during the forward process, leading to improved reverse denoising and more accurate event sequence generation. Unlike Zhang et al. (2024) and Song et al. (2024), which propose alternative generative formulations based on stochastic differential equations and Neural ODEs respectively, our contribution lies in redesigning the diffusion mechanism itself while retaining the tractability and training framework of DDPM-based diffusion models. We further provide theoretical analysis showing that the proposed adaptive noising preserves the core tractability guarantees while acting as a principled regularizer.

**Diffusion with other variants of noise.** While most existing diffusion models use Gaussian noise for training and sampling across all time steps, recent works in vision-based diffusion models have explored other variants of Gaussian noise in order to achieve better performance. For example, (Rissanen et al., 2022) explore a diffusion process inspired by heat dissipation. Similarly, (Voleti et al., 2022) employ non-isotropic Gaussian noise in score-based diffusion models. (Huang et al., 2024) employ time-varying correlated noise within and across images in their deterministic diffusion-based

framework, which gradually evolves into Gaussian white noise as the forward diffusion progresses. Among the more recent works, Nichol & Dhariwal (2021) introduced learned reverse-process variances whereas Sahoo et al. (2024) proposed pixel-wise adaptive noise injection, where the noise level is dynamically adjusted for individual pixels based on image content.

However, these methods are developed for vision-based diffusion models, where the noise design is motivated by image-specific properties such as spatial structure, frequency content, or pixel correlations. In contrast, our work addresses diffusion for temporal point processes, where the data consist of continuous-time event sequences rather than spatial signals. Therefore, none of the above approaches can trivially be extended to a temporal event data-based diffusion framework. To the best of our knowledge, existing diffusion-based TPP models employ isotropic Gaussian noise, and no prior work has explored data-dependent non-isotropic noise tailored to temporal event sequences. We bridge this gap by proposing a novel noising technique, tailored for multi-event sequence prediction for temporal point processes, whereby the variance of the Gaussian noise is customized in a per-event basis in a data-dependent manner.

Among existing works, Sahoo et al. (2024) is conceptually the closest to our approach, as both assign data-dependent noise levels at the level of individual data elements. However, their method is designed for image generation, where the adaptation is performed per pixel based on image content. In contrast, our method introduces per-event, data-dependent variance scaling for diffusion over temporal point processes, where the adaptive noise is motivated by the temporal characteristics of event sequences rather than spatial image structure.

However, these methods are developed for vision-based diffusion models, where the noise design is motivated by image-specific properties such as spatial structure, frequency content, or pixel correlations. In contrast, our work addresses diffusion for temporal point processes, where the data consist of continuous-time event sequences rather than spatial signals. Therefore, none of the above approaches can trivially be extended to a temporal event data-based diffusion framework.

## E Diffusion Framework

Given a temporal sequence, we first apply the Box-Cox transformation to $\boldsymbol{x}_u$ which maps it from strictly positive real values $\mathbb{R}^+ = (0, \infty)$ to unrestricted real values $\mathbb{R} = (-\infty, \infty)$.This transformation allows us to make the inter-arrival distribution more like a Gaussian. To encode the context, we apply a transformer-based context encoder (Xue et al., 2022) that encodes the variable-length context $\boldsymbol{s}_c$ to generate a fixed-dimensional intermediate representation denoted as $h = f(\boldsymbol{s}_c)$. Note that, during sampling we perform a reverse mapping on generated time intervals to obtain positive event time intervals.

**Forward Diffusion.** We define a single diffusion process for both time-intervals and event types. Let the original sequence be $\boldsymbol{s}_0 = (\boldsymbol{x}_0, \boldsymbol{m}_0)$, where $\boldsymbol{x}_0 = \boldsymbol{x}_u$ and $\boldsymbol{m}_0 = \boldsymbol{m}_u$. We define the forward diffusion process as the following joint distribution:

$$q(\boldsymbol{s}_{1:T}|\boldsymbol{s}_0) = \prod_{t=1}^{T} q(\boldsymbol{s}_t|\boldsymbol{s}_{t-1}) \tag{60}$$

During training, we learn the denoising process, for which we learn the intermediate distributions $p_\theta(\boldsymbol{s}_{t-1}|\boldsymbol{s}_t, \boldsymbol{s}_c)$. The training objective is to maximize the log-likelihood of the target sequence $\log q(\boldsymbol{s}_0|\boldsymbol{s}_c)$, which has been demonstrated by (Zeng et al., 2023) to be equivalent to minimizing the KL-divergence between the denoising distribution $p_\theta(\boldsymbol{s}_{t-1}|\boldsymbol{s}_t, \boldsymbol{s}_c)$ and the noisy distribution $q(\boldsymbol{s}_{t-1}|\boldsymbol{s}_t, \boldsymbol{s}_0, \boldsymbol{s}_c)$. The forward diffusion process, $q(\boldsymbol{s}_t|\boldsymbol{s}_{t-1})$ is further decomposed as

$$q(\boldsymbol{s}_t|\boldsymbol{s}_{t-1}) = q(\boldsymbol{x}_t, \boldsymbol{m}_t|\boldsymbol{x}_{t-1}, \boldsymbol{m}_{t-1}) \tag{61}$$

$$= q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) q(\boldsymbol{m}_t|\boldsymbol{m}_{t-1}) \tag{62}$$

Given a linear increasing variance schedule $\{\beta_1, .., \beta_T\}$, the forward noising distribution can be defined as -

$$q(\boldsymbol{s}_t|\boldsymbol{s}_{t-1}) = q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})q(\boldsymbol{m}_t|\boldsymbol{m}_{t-1}) \tag{63}$$

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t\mathcal{I}) \tag{64}$$

$$q(\boldsymbol{x}_T) = \mathcal{N}(\boldsymbol{x}_t; \boldsymbol{0}, \mathcal{I}) \tag{65}$$

$$q(\boldsymbol{m}_t|\boldsymbol{m}_{t-1}) = CAT(\boldsymbol{m}_t; (1-\beta_t)\boldsymbol{m}_{t-1}, \frac{\beta_t}{K}) \tag{66}$$

$$q(\boldsymbol{m}_T) = CAT(\boldsymbol{m}_t; 1/K) \tag{67}$$

**Denoising.** The denoising processes of the inter-arrival times and the event types are conditional on the historical context $\boldsymbol{s}_c$ and interact with each other. The denoising process $p_\theta(\boldsymbol{s}_{t-1}|\boldsymbol{s}_t)$ is defined as

$$\begin{aligned} p_\theta(\boldsymbol{s}_{t-1}|\boldsymbol{s}_t, \boldsymbol{s}_c) &= p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{s}_t, \boldsymbol{m}_{t-1}, \boldsymbol{s}_c)p_\theta(\boldsymbol{m}_{t-1}|\boldsymbol{s}_t, \boldsymbol{s}_c) \\ p_\theta(\boldsymbol{m}_{t-1}|\boldsymbol{s}_t, \boldsymbol{s}_c) &= CAT(\boldsymbol{m}_{t-1}|\pi_\theta(\boldsymbol{x}_t, \boldsymbol{m}_t, \boldsymbol{s}_c, t)) \\ p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{m}_{t-1}, \boldsymbol{s}_t, \boldsymbol{s}_c) &= \mathcal{N}(\boldsymbol{x}_{t-1}; \mu_\theta(\boldsymbol{m}_{t-1}, \boldsymbol{x}_t, \boldsymbol{s}_c, t), \sigma_t) \end{aligned} \tag{68}$$

$\mu_\theta$ and $\pi_\theta$ are learnable and $\sigma_t$ is set to $\beta_t$. $\mu_\theta$ is further parameterized as

$$\mu_\theta(\boldsymbol{x}_t, \boldsymbol{m}_{t-1}, t, \boldsymbol{s}_c) = \frac{\boldsymbol{x}_t}{\sqrt{\alpha_t}} - \frac{\beta_t\epsilon_\theta(\boldsymbol{x}_t, \boldsymbol{m}_{t-1}, t, \boldsymbol{s}_c)}{\sqrt{\alpha_t}\sqrt{1-\bar{\alpha}_t}} \tag{69}$$

where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$. For event types, we further learn $\phi_\theta$ that learns to predict the target event type $\hat{\boldsymbol{m}}_0$ at each step $t$, while taking as input a noisy sample at step $t$ as $\hat{\boldsymbol{m}}_0 = \phi_\theta(\boldsymbol{m}_t, \boldsymbol{x}_t, t, \boldsymbol{s}_c)$. Finally, this target event type $\hat{\boldsymbol{m}}_0$ is combined with the noisy event type $\boldsymbol{m}_t$, and normalized to generate a target type distribution $\pi_\theta$ as follows, whereas $\epsilon_\theta, \phi_\theta$ and $f_\theta$ are parameterized by transformer-based networks and $\odot$ denotes the Hadamard product.

$$\begin{aligned} \boldsymbol{\theta}(\boldsymbol{m}_t, \hat{\boldsymbol{m}}_0) &= [\alpha_t\boldsymbol{m}_t + \frac{1-\alpha_t}{K}] \odot [\bar{\alpha}_{t-1}\hat{\boldsymbol{m}}_0 + \frac{1-\bar{\alpha}_{t-1}}{K}] \\ \pi_\theta(\boldsymbol{x}_t, \boldsymbol{m}_t, t, \boldsymbol{s}_c) &= \frac{\boldsymbol{\theta}(\boldsymbol{m}_t, \hat{\boldsymbol{m}}_0)}{\sum_{k=1}^{K} \boldsymbol{\theta}(\boldsymbol{m}_t, \hat{\boldsymbol{m}}_0)_k} \end{aligned} \tag{70}$$

Model parameters are trained by optimizing the following loss function where the second and third term corresponds to the diffusion process for event-inter-arrival time and event types.

$$\begin{aligned} \mathcal{L} = \sum_{j=1}^{M} (\log p_\theta(\boldsymbol{x}_0^j|\boldsymbol{x}_1^j, \boldsymbol{m}_0^j, \boldsymbol{s}_c^j) \log p_\theta(\boldsymbol{m}_0^j|\boldsymbol{x}_1^j, \boldsymbol{m}_1^j, \boldsymbol{s}_c^j) - \sum_{t=2}^{T} (||\bar{\epsilon} - \epsilon_\theta(\sqrt{\bar{\alpha}_t}\boldsymbol{x}_0^j + \sqrt{1-\bar{\alpha}_t}\epsilon, t, \hat{\boldsymbol{m}}_{t-1}^j, \boldsymbol{s}_c^j)||^2 \\ + \sum_{k=1}^{K} \bar{\boldsymbol{\theta}}_{post}(\boldsymbol{m}_t^j, \boldsymbol{m}_0^j)_k \cdot \log \frac{\bar{\boldsymbol{\theta}}_{post}(\boldsymbol{m}_t^j, \boldsymbol{m}_0^j)_k}{\pi_\theta(\boldsymbol{x}_t^j, \boldsymbol{m}_t^j, t, \boldsymbol{s}_c^j)_k} \end{aligned} \tag{71}$$

Given a sequence context $\boldsymbol{s}_c$, we can generate $\boldsymbol{s}_0$ by sampling from $p_\theta(\boldsymbol{s}_0|\boldsymbol{s}_c)$. Final time intervals are decided by taking average of multiple samples, and final event types are decided by taking majority votes over the generated samples.

## F Implementational Details

Following Zeng et al. (2023), for each method, 7 samples are generated for each prediction. For time prediction, the generated samples are averaged. For type prediction, majority voting is used over generated samples. THP Zuo et al. (2020) is followed for positional encoding of temporal sequnence, AttNHP Yang et al. (2021) is followed for the history encoder, and Pytorch built-in transformer blocks are used for denoising decoders for time and type ($\epsilon_\theta(\cdot), \phi_\theta(\cdot)$). Cosine scheduler Nichol & Dhariwal (2021) is used for $\beta$ and $\gamma$, unless otherwise specified. All experiments were conducted on a single NVIDIA Tesla P100 PCIe 16 GB GPU. The total runtime was approximately 2 hours, including scale training for one seed, diffusion model training, and inference across five seeds combined per dataset, with the batch size maximized to fully utilize the available hardware. Despite this aggressive configuration, the overall memory consumption is confined within 16GB GPU..

## G SCALE LEARNING MODULE DETAILS

The scale learning module of SCALEDIFF, discussed in Section 4.2, conditions the diffusion process on both the historical context and the current noisy target state. Specifically, the scheduler input is constructed by concatenating the event context $x_c$, noisy event target $x_t$, and diffusion timestep $t$. The resulting representation is passed through a learnable linear projection followed by a layer normalization operation. Finally, the normalized outputs are mapped to the predefined interval $[\kappa_l, \kappa_u]$, yielding adaptive diffusion scales that dynamically regulate the amount of injected noise based on the temporal context and current diffusion state. Loss is calculated as defined in 4.2.

## H EVALUATION METRICS

In the following, we describe each metric in detail.
1) **Optimal Transport Distance (OTD)**: OTD measures the cost of transforming the generated sequence $\hat{\mathcal{S}}_u$ into the ground-truth sequence $\mathcal{S}_u$ using a dynamic programming algorithm that jointly considers event times and types. It penalizes insertions, deletions, and mismatches, with a tunable deletion cost hyperparameter $C_{\text{del}}$. The final OTD value is obtained by averaging over multiple $C_{\text{del}}$ values (e.g., 0.05 to 4.0), as recommended in prior work. A lower OTD indicates a closer match in both the event timings and the categorical structure of the sequences.

2) **RMSE**$_{Type}$: This metric employed for measuring type prediction performance computes how well the predicted event type distribution matches the ground truth. This metric evaluates the L2 norm distance between the predicted event type distribution $\hat{\mathbf{m}}_i$ and the one-hot true label $\mathbf{m}_i$, averaged across all positions, as given in the following equation, where $M$ is the number of sequences.

3) **RMSE**$_{Time}$: This metric measures the root mean squared error between the predicted and ground-truth inter-event times. This captures absolute timing fidelity; lower values indicate a more accurate interval prediction.

4) **Symmetric Mean Absolute Percentage Error (SMAPE)**: SMAPE is a scale-invariant relative error metric for measuring time-forecasting performance. It is defined as:

$$\text{RMSE}_{Time} = \sqrt{\frac{1}{M}\sum_{j=1}^{M}\left\|\mathbf{x}_u^{+,j} - \hat{\mathbf{x}}_u^{+,j}\right\|_2^2}$$

$$\text{RMSE}_{Type} = \sqrt{\frac{1}{M}\sum_{j=1}^{M}\left\|\mathbf{m}_u^{j} - \hat{\mathbf{m}}_u^{j}\right\|_2^2}$$

$$\text{SMAPE} = \frac{1}{M}\sum_{j=1}^{M}\frac{100}{N}\sum_{i=1}^{N}\frac{2\left|x_{u,i}^{+,j} - \hat{x}_{u,i}^{+,j}\right|}{\left|x_{u,i}^{+,j}\right| + \left|\hat{x}_{u,i}^{+,j}\right|}$$

## I DATASET DETAILS

We employ the five real-world datasets, following Zeng et al. (2023) to evaluate our method. The datasets include:

1) **Taxi** Whong (2014): This dataset contains taxi pickup and drop-off events in New York City in 2013. Following Mei et al. (2019), each event is a tuple of $(\text{location}, \text{action})$, where the location is one of the five NYC boroughs and the action is either pickup or drop-off. This gives $K = 5 \times 2 = 10$ event types, with types 0–4 corresponding to pickups and 5–9 to drop-offs. A total of 2000 sequences were retained, with an average sequence length of 39 and an average inter-arrival time of 0.22 hours (time unit = 1 hour). The training, validation, and test sets consist of 1400, 200, and 400 sequences, respectively.
2) **StackOverflow** Leskovec & Krevl (2014): This dataset tracks user badge-award events on the StackOverflow Q&A platform over a two-year period. Each badge corresponds to a unique event type, with $K = 22$ distinct badge types in total. The training, validation, and test sets comprise 1400, 400, and 400 user sequences, respectively. The time unit is 11 days, and the average inter-arrival time

between events is 0.95.
3) **Retweet** Zhou et al. (2013): It contains retweet events annotated with timestamps and categorized into three user-type classes: small, medium, and large users. Users with fewer than 120 followers are considered "small," those with fewer than 1363 as "medium," and all others as "large," yielding $K = 3$ event types. We use a subset of 9000 retweet sequences. The train, validation, and test splits contain 6000, 1500, and 1500 sequences, respectively. This dataset features short inter-event times and bursty behavior. There are 2 versions of this dataset available. We have used the one that C-Diff linked to from their official repo.
4) **Amazon** Ni et al. (2019): This dataset tracks user product reviews over a ten-year period from January 2008 to October 2018. Each event is associated with the timestamp and product category reviewed. The dataset was filtered to include the 5,200 most active users, each with an average sequence length of 70. Product categories were collapsed into $K = 16$ event types. This dataset features long sequences and highly sparse temporal dynamics across users.
5) **Taobao** Zhu et al. (2018): This dataset captures user click events on Taobao's shopping website between November 25 and December 03, 2017. Each user's activity is represented as a sequence of item clicks with timestamps and associated item categories. To reduce complexity, the item categories were ranked by frequency, and only the top 16 were retained; all remaining categories were grouped into a single "other" category, resulting in $K = 17$ event types. The dataset includes 2,000 of the most active users, each with an average sequence length of 58. The disjoint training, validation, and test sets consist of 1300, 200, and 500 sequences respectively. The time unit is 3 hours, with an average inter-arrival time of 0.06 (i.e., approximately 0.18 hours).
Results on the test split is reported where the train-test split given by Zeng et al. (2023) is followed.

## J BASELINES

We compare our method with standard state-of-the-art baselines, which were followed in CDIFF. For CDIFF and other baselines, the results reported in CDIFF are used, as we employ the same train-test split as used in CDIFF. The baselines include:
**Neural Hawkes Process (NHP)** Mei & Eisner (2017): A Hawkes-based model utilizing a continuous-time LSTM to capture temporal dependencies.
**Attentive Neural Hawkes Process (AttNHP)** Yang et al. (2022): Enhances the Hawkes process with attention mechanisms, achieving state-of-the-art performance in single event forecasting.
**Log-Normal Mixture Model (LNM)** Shchur et al. (2020): An intensity-free temporal point process model that employs a log-normal mixture distribution, offering fast sampling capabilities.
**Temporal Conditional Diffusion Denoising Model (TCDDM)** Lin et al. (2022): A diffusion-based generative model assuming conditional independence between inter-arrival times and event types.
**Dual-TPP** Deshpande et al. (2021): Targets long-horizon forecasting by jointly learning the distribution of event counts in segmented time intervals.
**HYPRO** Xue et al. (2022): A hybridly normalized probabilistic model for long-horizon prediction, combining an autoregressive base with an energy-based reweighting mechanism.
**Residual tpp** Yuan & Fang (2025) A simple base model for capturing overall patterns with a neural network that models residual events - which captures patterns at first and then that pattern is distilled to a base model, providing a flexible, lightweight, and computationally efficient approach for event stream analysis. We use the AttNHP variant.
**NJDTPP** Zhang et al. (2024) A generative diffusion model which captures the intensity dynamics of temporal point processes using a flexible neural jump-diffusion SDE, without relying on a predefined functional form.
**CDIFF** Zeng et al. (2023) A generative diffusion model allowing multi-step predictions based on history, along with learning the joint probability distribution of types and inter-arrival times

## K RANKING OF ALGORITHMS

Figure 5 ranks the algorithms based on their performances across the datasets for all the metrics. We observe the proposed noise scaling approach (SCALEDIFF) to rank superior for all time-related metrics (OTD, $\text{RMSE}_{Time}$, and SMAPE). For event type prediction metric $\text{RMSE}_{Type}$ however, it shows comparable performance with HYPRO, DUAL-TPP, and CDIFF.

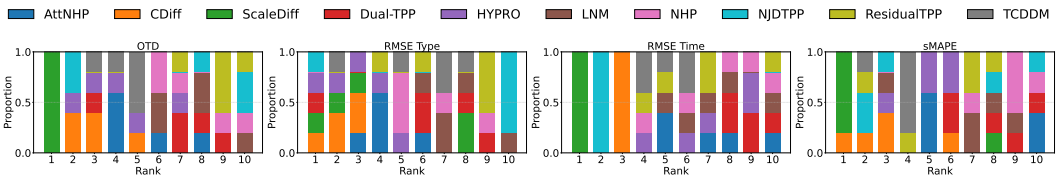



Figure 5: Ranking of algorithms for diverse metrics across all the datasets. SCALEDIFF consistently rank amongs the top for time prediction across the datasets.

## L SHORTER HORIZON FORECASTING

Refer Figure 6 for forecasting results on different horizons for AMAZON, TAXI, and STACKOVERFLOW.

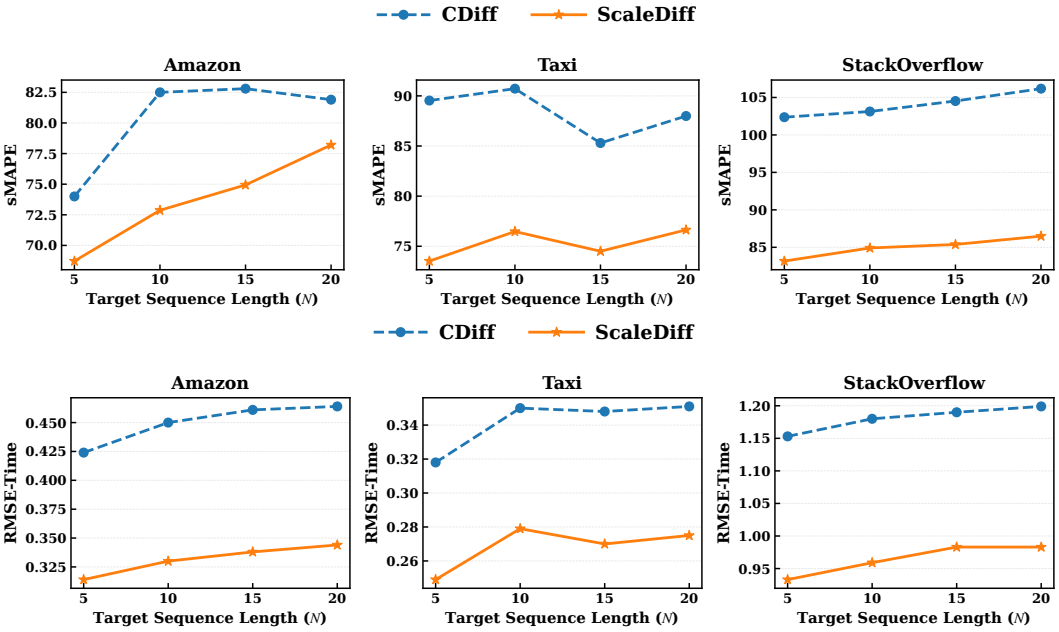



Figure 6: sMAPE and $\text{RMSE}_{Time}$ for different horizons for SCALEDIFF, and CDIFF for AMAZON, STACKOVERFLOW, and TAXI datasets. For $\text{RMSE}_{Time}$, performance improvement is more prominent for longer horizons.

## M QUALITATIVE ANALYSIS

We assess whether the generated inter-arrival times $\hat{x}_i$ match the empirical distribution of ground-truth data $x_i$ in Figure 7 for AMAZON dataset. To this end, we employ Kernel Density Estimation (KDE), which evaluates how well the model captures the global shape of the temporal dynamics. Though both the predicted distributions ignore the minor peak of the ground truth around 0, SCALEDIFF captures the next peak (around 0.8) more faithfully over the baseline. On the contrary, CDIFF distributes the density more evenly across a wider region. It also shows SCALEDIFF learns lower entropy/more informative inter-arrival time distribution, indicating more fine-grained learning.

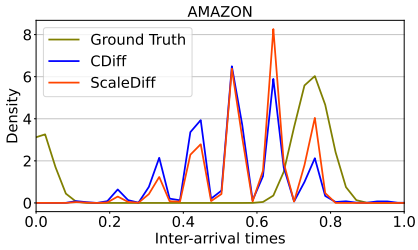



Figure 7: KDE estimation of inter-arrival time distribution for ground truth and predicted sequences for SCALEDIFF and CDIFF for AMAZON dataset. SCALEDIFF learns lower entropy/more informative inter-arrival time distribution, indicating more fine-grained learning.

## N ANALYSIS OF BOX-COX TRANSFORMATION

**Box-cox preserves sparse and dense regions** We further validate that Box cox transformation ensures sparse and dense regions in data. Box-cox is monotonic in nature, i.e. for $x_i < x_j$, boxcox($x_i$)<boxcox($x_j$) for $i \neq j$. The box-cox transformation function is represented as BoxCox($x$)=$\frac{x^\lambda - 1}{\lambda}$ for $\lambda > 0$. For all datasets we tested, the transformation preserves monotonicity and actually amplifies differences in inter-arrival times, especially in regions where the raw values are very small. This ensures that sparse and dense regions remain distinguishable after transformation rather than being smoothed out.

We illustrate with an example sequence (in sorted order - not the actual order) from TAXI where we compare time intervals before and after box-cox transformation as shown in Table 2.

Table 2: Sorted inter-event time intervals before and after Box–Cox Transformation. We observe that sparse and dense regions remain distinguishable after transformation rather than being smoothed out.

| **Original sequence** | **After Box–Cox** |
|---:|---:|
| 0.01083 | -1.75237 |
| 0.02305 | -1.40184 |
| 0.02527 | -1.35305 |
| ⋮ | ⋮ |
| 0.15388 | -0.03704 |
| 0.15555 | -0.02669 |
| 0.18694 | 0.15557 |
| 0.21833 | 0.31806 |
| ⋮ | ⋮ |
| 0.61277 | 1.63035 |
| 0.82305 | 2.09235 |
| 0.90805 | 2.25621 |

The transformed values are mapped into a numerically stable range that still reflects the differences between small and large intervals. Importantly, without the Box–Cox transformation, additive Gaussian noise in the forward diffusion process would easily destroy the relative ordering of raw inter-arrival times, especially in datasets where the scale varies by several orders of magnitude. Furthermore, the entropy for inter-event times with and without Box-Cox are respectively $2.1436$ and $1.7461$ respectively, implying that box cox events are more evenly spread.

## O COMPARISON WITH ADD-AND-THIN

We perform a comparative study of SCALEDIFF against ADD-AND-THIN Lüdke et al. (2024) on TAXI, STACKOVERFLOW, AMAZON, and TAOBAO datasets. Table 3 reports the results for $\text{RMSE}_{Time}$ and **sMAPE**.

For this study, we run ADD-AND-THIN in a conditional setting, where the model is trained to predict the next 20 future events given the historical sequence. Since ADD-AND-THIN models only inter-arrival times and does not incorporate event types, we repeat the experiments of SCALEDIFF by initializing all event types to be identical across the datasets to ensure a fair comparison.

The results show that ADD-AND-THIN consistently underperforms compared to SCALEDIFF. This performance gap arises because ADD-AND-THIN predicts, on average, only 16–17 future events, even when the prediction horizon is extended to a large time range. As a result, the model fails to generate the full set of required future events, leading to degraded forecasting accuracy.

Table 3: Time prediction comparison of SCALEDIFF versus ADD-AND-THIN Lüdke et al. (2024) on TAXI, STACKOVERFLOW, AMAZON, and TAOBAO datasets (Lower the better). Best results are in **bold**.

| | TAXI | | STACKOVERFLOW | | AMAZON | | TAOBAO | |
|---|---|---|---|---|---|---|---|---|
| **Method** | **$\text{RMSE}_{Time}$** | **sMAPE** | **$\text{RMSE}_{Time}$** | **sMAPE** | **$\text{RMSE}_{Time}$** | **sMAPE** | **$\text{RMSE}_{Time}$** | **sMAPE** |
| ADD-AND-THIN | 1.503±0.006 | 92.887±2.194 | 29.310±3.170 | 92.926±12.760 | 12.978±4.139 | 114.752±33.273 | 2.097±0.825 | 135.035±8.898 |
| SCALEDIFF | **0.291±0.013** | **76.199±1.4545** | **0.973±0.017** | **84.594±0.260** | **0.348±0.001** | **70.938±1.240** | **0.462±0.001** | **138.246±2.747** |

## P EXPERIMENTAL RESULTS ON SYNTHETIC DATASETS

To further analyze the behavior of ScaleDiff under controlled temporal dynamics, we construct two synthetic 3-dimensional event-stream datasets: (i) a Hawkes process exhibiting alternating sparse and dense regimes, and (ii) a Poisson process with comparatively homogeneous arrivals.

As shown in Table 4, SCALEDIFF consistently outperforms CDiff across both datasets. The improvements are particularly pronounced in the Hawkes setting, where temporal clustering and inter-arrival variability are substantially higher. Notably, SCALEDIFF achieves a significant reduction in RMSE Time, indicating improved modeling of highly non-uniform temporal dynamics. In contrast, gains on the Poisson dataset are comparatively modest, which is expected due to its more stationary arrival behavior and lower variance structure. Overall, these results support our central claim that the proposed context-aware noising mechanism is especially effective in heterogeneous temporal environments.

## Q DETAILED ABLATIONS

Appendix Q reports detailed ablation study, which includes (i) learned scale vs ground-truth scale in both forward diffusion and inference, which shows SCALEDIFF remains largely consistent under both settings.

**Learned scales vs Ground-Truth scales.** Since the ground-truth scale is unavailable during reverse diffusion, our method learns $S_0$. To quantify the effect of using a learned scale, we replace the learned scales with the ground-truth scales during the forward process in training. The results are reported in Table 5, where we observe that the performance of SCALEDIFF remains largely consistent when using the predicted scale instead of the true scale. Across all five datasets, the two variants achieve nearly identical results on all evaluation metrics, indicating that the proposed scale prediction network accurately estimates the latent scale required during reverse diffusion.

We further compare the predicted scale with the ground-truth scale during inference and report results in Table 6. The results show that the learned scale achieves comparable or better performance in several cases (e.g., Taxi and StackOverflow), while both variants perform similarly on Amazon and Taobao. This suggests that the scale predictor learns a representation better suited for the reverse denoising process rather than merely approximating the ground-truth scale.

Table 4: Performance comparison on synthetic Hawkes and Poisson datasets.

| Dataset | Method | OTD ↓ | RMSE Type ↓ | RMSE Time ↓ | SMAPE ↓ |
|---|---|---|---|---|---|
| Hawkes | CDiff | $41.452 \pm 0.660$ | $3.885 \pm 0.123$ | $1.763 \pm 0.103$ | $90.018 \pm 1.740$ |
| | SCALEDIFF | $\mathbf{40.724 \pm 0.277}$ | $\mathbf{3.852 \pm 0.187}$ | $\mathbf{1.699 \pm 0.052}$ | $\mathbf{88.301 \pm 0.687}$ |
| Poisson | CDiff | $21.334 \pm 0.139$ | $3.002 \pm 0.067$ | $0.290 \pm 0.003$ | $80.091 \pm 0.652$ |
| | SCALEDIFF | $\mathbf{21.220 \pm 0.219}$ | $\mathbf{2.964 \pm 0.111}$ | $\mathbf{0.287 \pm 0.005}$ | $\mathbf{79.620 \pm 0.738}$ |

Table 5: Comparison of ScaleDiff using true and predicted scale across different datasets during Training.

| Dataset | Method | OTD | RMSE Type | RMSE Time | SMAPE |
|---|---|---|---|---|---|
| Amazon | ScaleDiff (True Scale) | **37.293±0.294** | **2.147±0.016** | 0.346±0.001 | 78.221±0.613 |
| | ScaleDiff (Predicted Scale) | 37.345±0.228 | 2.155±0.020 | **0.344±0.002** | **78.203±0.072** |
| Taobao | ScaleDiff (True Scale) | 42.268±0.275 | 2.458±0.063 | 0.406±0.001 | **139.582±1.252** |
| | ScaleDiff (Predicted Scale) | **42.081±0.284** | **2.443±0.065** | **0.405±0.002** | 139.708±1.712 |
| Taxi | ScaleDiff (True Scale) | 18.987±0.197 | 1.037±0.040 | 0.276±0.002 | 76.883±0.500 |
| | ScaleDiff (Predicted Scale) | **18.767±0.374** | **1.013±0.056** | **0.275±0.003** | **76.631±0.814** |
| Stack Overflow | ScaleDiff (True Scale) | 38.956±0.473 | **1.557±0.050** | **0.980±0.051** | 86.593±0.158 |
| | ScaleDiff (Predicted Scale) | **38.918±0.556** | 1.501±0.057 | 0.983±0.012 | **86.492±0.185** |
| Retweet | ScaleDiff (True Scale) | 58.895±0.281 | 2.925±0.053 | **20.986±0.593** | **84.945±0.284** |
| | ScaleDiff (Predicted Scale) | **58.832 ± 0.201** | 2.901 ± 0.060 | 21.074 ± 0.054 | 85.252 ± 0.349 |

**Global event-wise scales vs per-sequence scales** Further to investigate the impact of the choice of scale heuristic, we perform the following experiment where the scale heuristic computes global event-wise scales instead of per-sequence scales. Specifically, during training of the scale predictor, event-wise scales are obtained by applying min-max normalization to inter-arrival times, where the minimum and maximum inter-arrival times are computed across all training sequences, rather than independently for each sequence. Apart from this modification, all other components of ScaleDiff remain unchanged. The results are reported in Table 7, where we observe that the effectiveness of the scale normalization strategy is dataset-dependent. On the StackOverflow dataset, the global-scale variant achieves slightly better OTD, RMSE-Time, and sMAPE, while the per-sequence variant attains a lower RMSE-Type. In contrast, on the Taxi dataset, the per-sequence scaling strategy consistently outperforms the global-scale variant on OTD, RMSE-Type, and sMAPE, while both approaches achieve nearly identical RMSE-Time. These observations suggest that neither normalization strategy is uniformly superior across datasets. Instead, the appropriate choice depends on the characteristics of the underlying temporal data. In particular, datasets such as Taxi exhibit substantial sequence-level variation in inter-arrival times, where per-sequence normalization better preserves local temporal dynamics. Conversely, on datasets like STACKOVERFLOW, where the global and per-sequence inter-arrival time distributions are more similar, both normalization strategies produce comparable normalized scales, leading to similar overall performance

**Impacts of Different Scale Heuritics** To analyze the impact of different scale heuristics, we further replace the proposed min-max normalization with a quantile-based scaling strategy. Instead of assigning scaling factors according to the magnitude of the temporal scores, this variant first converts each score into its empirical percentile rank within the sequence. Let $s_i$ denote the temporal score of the $i$-th event. The scores are sorted to obtain their ranks $r_i \in \{0, \dots, T-1\}$, which are normalized as $q_i = \frac{r_i}{T-1}$ where $q_i \in [0, 1]$ represents the empirical quantile of the event. The corresponding scaling factor is then computed as $s_i^{final} = \kappa_l + q_i(\kappa_u - \kappa_l)$, where $\kappa_l$ and $\kappa_u$ denote the lower and upper scaling bounds, respectively.

The results are reported in Table 8, demonstrating that the proposed min-max scaling consistently achieves lower OTD and RMSE-Time across all five datasets, and also yields better sMAPE on four of the five datasets. Although quantile scaling achieves slightly better RMSE-Type on three datasets (and ties on Retweet), it consistently underperforms on the primary temporal prediction metrics. These results suggest that preserving the magnitude of temporal scores, rather than only

Table 6: Performance comparison of ScaleDiff using predicted and true scale values. ↓ indicates that lower values are better.

| Dataset | Method | OTD↓ | RMSE Type↓ | RMSE Time↓ | sMAPE↓ |
|---|---|---|---|---|---|
| Amazon | ScaleDiff (Predicted Scale) | **37.345±0.228** | **2.155±0.020** | **0.344±0.002** | 78.203±0.072 |
| | ScaleDiff (True Scale) | 37.410 ± 0.139 | 2.158 ± 0.018 | 0.346 ± 0.002 | **78.154 ± 0.604** |
| Taobao | ScaleDiff (Predicted Scale) | **42.081±0.284** | **2.443±0.065** | **0.405±0.002** | **139.708±1.712** |
| | ScaleDiff (True Scale) | 42.220 ± 0.313 | 2.451 ± 0.073 | 0.406±0.002 | 139.778 ± 1.530 |
| Taxi | ScaleDiff (Predicted Scale) | **18.767±0.374** | **1.013±0.056** | **0.275±0.003** | **76.631±0.814** |
| | ScaleDiff (True Scale) | 18.944 ± 0.218 | 1.047 ± 0.023 | 0.276 ± 0.003 | 76.638 ± 0.714 |
| StackOverflow | ScaleDiff (Predicted Scale) | **38.918±0.556** | **1.501±0.057** | **0.983±0.012** | **86.492±0.185** |
| | ScaleDiff (True Scale) | 38.983±0.473 | 1.560±0.050 | 0.985±0.014 | 86.504±0.167 |

Table 7: Effect of scale granularity on StackOverflow and Taxi. Per-sequence scales are computed independently for each sequence, whereas global scales are computed event-wise across the dataset.

| Dataset | Method | OTD ↓ | RMSE-Type ↓ | RMSE-Time ↓ | sMAPE ↓ |
|---|---|---|---|---|---|
| **StackOverflow** | **ScaleDiff (Per-Sequence Scale)** | **38.918±0.556** | **1.501±0.037** | **0.983±0.012** | **86.492±0.185** |
| | ScaleDiff (Global Scale) | 38.969 ± 0.260 | 1.543 ± 0.024 | 0.986 ± 0.009 | 86.556 ± 0.197 |
| **Taxi** | **ScaleDiff (Per-Sequence Scale)** | **18.767±0.374** | **1.013±0.056** | **0.275±0.003** | **76.631±0.814** |
| | ScaleDiff (Global Scale) | 19.134 ± 0.471 | 1.030 ± 0.054 | 0.278 ± 0.003 | 77.389 ± 1.0241 |

their ordering, is important for effective adaptive noise scheduling in diffusion-based temporal point processes.

**Impact of $\gamma$ scheduler.** To directly investigate the role of the $\gamma$ scheduler, we performed an additional ablation using three alternative scheduler designs (Quadratic, Sigmoid, and Square-root), as well as a ScaleDiff (No Gamma) variant in which the scaling factor is kept constant throughout the diffusion process. Linear variant indicates original ScaleDiff. The results are summarized in Table 9.

Results indicate that the specific scheduler shape is not critical. Linear, Quadratic, Sigmoid, and Square-root schedules consistently achieve very similar performance across all datasets, indicating that ScaleDiff is robust to the exact functional form of the scheduler. However, progressively reducing the scaling factor is important. Removing the scheduler altogether (ScaleDiff-No-Gamma) generally leads to worse overall performance across datasets, particularly on OTD, RMSE-Time, and sMAPE. Although individual metrics may occasionally fluctuate, the overall trend consistently favors using a decreasing scheduler.

These results suggest that the scheduler's primary role is not the particular decay function itself, but rather ensuring that the additional scale modulation gradually diminishes as the reverse diffusion process approaches cleaner samples. Maintaining the same scaling throughout all diffusion steps perturbs late-stage denoising, resulting in inferior overall performance.

## R SENSITIVITY ANALYSIS

**Removing partial history:** To evaluate the robustness of our method to predict errors, we randomly remove 10%–40% of the historical events during sampling and measure the resulting performance. We report the results in Table 10, indicating that the proposed approach is robust to moderate perturbations in the estimated scale/history, with only marginal degradation in performance even when up to 40% of the history is removed.

**Sensitivity to scale prediction error.** To further isolate the contribution of the proposed locality-aware noising mechanism, we perturb the predicted scales during inference. Specifically, the predicted scales are mixed with Gaussian noise according to

$$s_{\text{perturbed}} = (1-\phi)s_{\text{pred}} + \phi\epsilon, \qquad \epsilon \sim \mathcal{N}(0,1) \tag{72}$$

where $\phi$ is the mixing coefficient. We evaluate three perturbation levels, corresponding to $\phi = 0.2, 0.3$, and $0.4$. (for $\phi$ = 40%, error = 51.43%, for $\phi$ = 30%, error is 38.53%, and for $\phi$ = 20%, error

Table 8: Comparison of the proposed scale heuristic with a quantile-based scale heuristic across datasets.

| Dataset | Method | OTD ↓ | RMSE-Type ↓ | RMSE-Time ↓ | sMAPE ↓ |
|---|---|---|---|---|---|
| **Taxi** | ScaleDiff Quantile | $18.979 \pm 0.247$ | $1.027 \pm 0.067$ | $0.277 \pm 0.002$ | $76.969 \pm 0.471$ |
| | **ScaleDiff** | $\mathbf{18.767 \pm 0.374}$ | $\mathbf{1.013 \pm 0.056}$ | $\mathbf{0.275 \pm 0.003}$ | $\mathbf{76.631 \pm 0.814}$ |
| **Amazon** | ScaleDiff Quantile | $37.452 \pm 0.164$ | $\mathbf{2.134 \pm 0.022}$ | $0.346 \pm 0.003$ | $78.606 \pm 0.652$ |
| | **ScaleDiff** | $\mathbf{37.345 \pm 0.228}$ | $2.155 \pm 0.020$ | $\mathbf{0.344 \pm 0.002}$ | $\mathbf{78.203 \pm 0.672}$ |
| **StackOverflow** | ScaleDiff Quantile | $39.819 \pm 0.354$ | $\mathbf{1.533 \pm 0.030}$ | $0.998 \pm 0.007$ | $86.586 \pm 0.156$ |
| | **ScaleDiff** | $\mathbf{38.918 \pm 0.556}$ | $1.501 \pm 0.037$ | $\mathbf{0.983 \pm 0.012}$ | $\mathbf{86.492 \pm 0.185}$ |
| **Taobao** | ScaleDiff Quantile | $42.282 \pm 0.266$ | $\mathbf{2.461 \pm 0.060}$ | $0.408 \pm 0.001$ | $140.594 \pm 1.239$ |
| | **ScaleDiff** | $\mathbf{42.081 \pm 0.284}$ | $2.443 \pm 0.065$ | $\mathbf{0.405 \pm 0.002}$ | $\mathbf{139.708 \pm 1.712}$ |
| **Retweet** | ScaleDiff Quantile | $59.045 \pm 0.112$ | $\mathbf{2.874 \pm 0.080}$ | $21.149 \pm 0.067$ | $85.294 \pm 0.251$ |
| | **ScaleDiff** | $\mathbf{58.832 \pm 0.201}$ | $2.901 \pm 0.060$ | $\mathbf{21.074 \pm 0.054}$ | $\mathbf{85.252 \pm 0.349}$ |

Table 9: Ablation study of different $\gamma$ scheduling strategies across datasets.

| Dataset | Type of Schedule | OTD ↓ | RMSE Type ↓ | RMSE Time ↓ | sMAPE ↓ |
|---|---|---|---|---|---|
| Amazon | Quad | $37.510 \pm 0.122$ | $2.147 \pm 0.049$ | $0.345 \pm 0.003$ | $78.500 \pm 0.665$ |
| | Sigmoid | $37.363 \pm 0.199$ | $2.153 \pm 0.018$ | $0.345 \pm 0.003$ | $78.144 \pm 0.450$ |
| | Sqrt | $37.402 \pm 0.209$ | $2.148 \pm 0.022$ | $0.350 \pm 0.003$ | $80.953 \pm 0.386$ |
| | No Gamma | $37.519 \pm 0.144$ | $\mathbf{2.138 \pm 0.028}$ | $0.350 \pm 0.002$ | $80.363 \pm 0.503$ |
| | **Linear** | $\mathbf{37.345 \pm 0.228}$ | $2.155 \pm 0.020$ | $\mathbf{0.344 \pm 0.002}$ | $\mathbf{78.203 \pm 0.672}$ |
| TaoBao | Quad | $42.145 \pm 0.346$ | $2.446 \pm 0.062$ | $0.406 \pm 0.007$ | $139.866 \pm 3.031$ |
| | Sigmoid | $42.144 \pm 0.342$ | $2.444 \pm 0.067$ | $0.405 \pm 0.006$ | $139.745 \pm 0.664$ |
| | Sqrt | $42.183 \pm 0.280$ | $\mathbf{2.439 \pm 0.063}$ | $0.412 \pm 0.005$ | $142.679 \pm 1.567$ |
| | No Gamma | $42.187 \pm 0.286$ | $\mathbf{2.439 \pm 0.069}$ | $0.409 \pm 0.001$ | $142.216 \pm 0.791$ |
| | **Linear** | $\mathbf{42.081 \pm 0.349}$ | $2.443 \pm 0.060$ | $\mathbf{0.405 \pm 0.002}$ | $\mathbf{139.708 \pm 1.712}$ |
| Taxi | Quad | $19.093 \pm 0.338$ | $1.025 \pm 0.053$ | $0.276 \pm 0.003$ | $76.697 \pm 0.636$ |
| | Sigmoid | $19.000 \pm 0.237$ | $1.048 \pm 0.041$ | $0.277 \pm 0.002$ | $76.801 \pm 0.628$ |
| | Sqrt | $18.938 \pm 0.568$ | $1.038 \pm 0.086$ | $0.278 \pm 0.002$ | $76.986 \pm 0.832$ |
| | No Gamma | $19.380 \pm 0.246$ | $1.058 \pm 0.048$ | $0.279 \pm 0.002$ | $77.840 \pm 0.446$ |
| | **Linear** | $\mathbf{18.767 \pm 0.374}$ | $\mathbf{1.013 \pm 0.056}$ | $\mathbf{0.275 \pm 0.003}$ | $\mathbf{76.631 \pm 0.814}$ |
| StackOverflow | Quad | $38.977 \pm 0.246$ | $1.534 \pm 0.029$ | $0.984 \pm 0.004$ | $86.503 \pm 0.143$ |
| | Sigmoid | $39.400 \pm 0.206$ | $1.505 \pm 0.028$ | $1.009 \pm 0.011$ | $87.007 \pm 0.132$ |
| | Sqrt | $39.113 \pm 0.172$ | $1.535 \pm 0.041$ | $1.013 \pm 0.006$ | $87.238 \pm 0.372$ |
| | No Gamma | $39.160 \pm 0.146$ | $1.532 \pm 0.023$ | $0.987 \pm 0.006$ | $87.004 \pm 0.338$ |
| | **Linear** | $\mathbf{38.918 \pm 0.556}$ | $\mathbf{1.501 \pm 0.057}$ | $\mathbf{0.983 \pm 0.012}$ | $\mathbf{86.492 \pm 0.185}$ |
| Retweet | Quad | $58.996 \pm 0.103$ | $2.887 \pm 0.110$ | $21.177 \pm 0.058$ | $85.293 \pm 0.325$ |
| | Sigmoid | $58.909 \pm 0.205$ | $2.917 \pm 0.055$ | $21.083 \pm 0.062$ | $85.282 \pm 0.288$ |
| | Sqrt | $59.004 \pm 0.236$ | $2.805 \pm 0.140$ | $21.318 \pm 0.273$ | $86.243 \pm 0.451$ |
| | No Gamma | $59.022 \pm 0.210$ | $2.837 \pm 0.083$ | $21.352 \pm 0.053$ | $86.364 \pm 0.212$ |
| | **Linear** | $\mathbf{58.832 \pm 0.278}$ | $\mathbf{2.901 \pm 0.060}$ | $\mathbf{21.074 \pm 0.054}$ | $\mathbf{85.252 \pm 0.349}$ |

is 25.69%) The results are reported in Table 11 where we observe that the model remains reasonably robust under a moderate perturbation ($\phi = 0.2$), with only a small degradation across all evaluation metrics. However, increasing the perturbation to 30% and 40% causes a dramatic deterioration in performance, indicating that corrupting the learned scales severely disrupts the diffusion process. This experiment provides direct evidence that the learned scale predictor ($f_{\theta_s}$) captures meaningful locality-aware temporal information rather than acting as a superficial auxiliary component.

## S Computational overhead

We analyze and compare the computational overhead of ScaleDiff with baselines. Table 12 reports the peak GPU memory usage and inference time measured on the same hardware. Further, to quantify the overhead introduced by the additional scale prediction network Table 13 reports the training cost of the scale prediction network ($f_\theta(S)$), which is trained only once before diffusion model training.

Table 10: Robustness of ScaleDiff to removal of temporal events.

| Dataset | Fraction Removed | OTD↓ | RMSE Type↓ | RMSE Time↓ | sMAPE↓ |
|---|---|---|---|---|---|
| Amazon | 0.1 | 37.596 ± 0.194 | 2.167 ± 0.019 | 0.344 ± 0.002 | 78.966 ± 0.754 |
| | 0.2 | 37.593 ± 0.153 | 2.198 ± 0.016 | 0.344 ± 0.003 | 78.457 ± 0.879 |
| | 0.3 | 37.674 ± 0.114 | 2.212 ± 0.007 | 0.345 ± 0.003 | 79.060 ± 0.912 |
| | 0.4 | 37.800 ± 0.129 | 2.228 ± 0.023 | 0.345 ± 0.003 | 79.690 ± 1.065 |
| Taxi | 0.1 | 19.327 ± 0.471 | 1.051 ± 0.035 | 0.276 ± 0.003 | 76.649 ± 0.540 |
| | 0.2 | 19.357 ± 0.447 | 1.081 ± 0.054 | 0.276 ± 0.003 | 76.786 ± 0.582 |
| | 0.3 | 19.513 ± 0.350 | 1.109 ± 0.064 | 0.276 ± 0.003 | 76.716 ± 0.544 |
| | 0.4 | 19.860 ± 0.194 | 1.103 ± 0.064 | 0.277 ± 0.003 | 76.778 ± 0.603 |
| StackOverflow | 0.1 | 39.002 ± 0.547 | 1.561 ± 0.057 | 0.984 ± 0.012 | 86.507±0.139 |
| | 0.2 | 39.377 ± 0.377 | 1.582 ± 0.056 | 0.984 ± 0.011 | 86.675 ± 0.178 |
| | 0.3 | 39.661 ± 0.451 | 1.599 ± 0.070 | 0.985 ± 0.010 | 86.744 ± 0.329 |
| | 0.4 | 40.292 ± 0.193 | 1.614 ± 0.062 | 0.986±0.009 | 87.247 ± 0.456 |

Table 11: Performance of ScaleDiff with different noise perturbations in percentage $\alpha$ on STACKOVERFLOW dataset.

| Dataset | Method | OTD ↓ | RMSE-Type ↓ | RMSE-Time ↓ | sMAPE ↓ |
|---|---|---|---|---|---|
| **StackOverflow** | **ScaleDiff** ($\alpha = 0$) | **38.918 ± 0.556** | **1.501 ± 0.057** | **0.983 ± 0.012** | **86.492 ± 0.185** |
| | ScaleDiff ($\alpha = 0.2$) | 39.885 ± 0.336 | 1.520 ± 0.032 | 0.986 ± 0.006 | 86.903 ± 0.483 |
| | ScaleDiff ($\alpha = 0.3$) | 68.007 ± 0.007 | 4.613 ± 0.000 | 1.275 ± 0.000 | 189.843 ± 0.278 |
| | ScaleDiff ($\alpha = 0.4$) | 68.108 ± 0.000 | 4.613 ± 0.000 | 1.287 ± 0.000 | 200.000 ± 0.000 |

Here we observe that ScaleDiff incurs additional computation and memory compared to CDiff, reflecting the cost of modeling event-specific scales. However, this overhead remains practical across all benchmark datasets, and the one-time training cost of the scale prediction network is modest relative to the overall training pipeline.

## T RECONSTRUCTION ERROR ANALYSIS

Further, to justify our motivation, we performed an additional analysis where we partition the events into bins according to their ground-truth inter-arrival times (from dense to sparse regions), and the reconstruction error is evaluated separately for each bin. The results are reported in Figure 8.

Across all benchmark datasets, ScaleDiff consistently achieves lower reconstruction error than CDiff in the dense regions (small inter-arrival times), where the adverse effect of uniform Gaussian perturbation is expected to be the strongest. This is precisely the regime that motivates our scale-aware forward diffusion process. While the performance gap naturally becomes smaller in sparser regions, ScaleDiff remains competitive and often continues to outperform CDiff. This experiment directly validates that the observed performance improvements indeed arise from improved reconstruction of dense temporal regions.

## U HYPER-PARAMETER TUNING

To further investigate the sensitivity of ScaleDiff to the lower scaling bound, we vary the parameter $\kappa_l$ over the range $0.1, 0.2, 0.4, 0.6$ and report the corresponding performances in Table 14. The proposed setting corresponds to $\kappa_l = 0.8$. Overall, increasing $\kappa_l$ consistently improves the temporal prediction metrics (OTD, RMSE-Time, and sMAPE) across all benchmark datasets. Although intermediate values occasionally achieve slightly better RMSE-Type, the proposed setting ($\kappa_l = 0.8$) consistently provides the best overall trade-off and yields the strongest performance on the primary temporal metrics. These results indicate that assigning a higher minimum noise scale improves the effectiveness of the proposed adaptive diffusion process while maintaining robustness across diverse datasets.

Table 12: Inference cost comparison between ScaleDiff and CDiff.

| Dataset | ScaleDiff | | CDiff | |
|---|---|---|---|---|
| | Memory (MB) | Time (s) | Memory (MB) | Time (s) |
| Amazon | 5128 | 1022 | 872 | 745 |
| Retweet | 6554 | 1294 | 848 | 845 |
| TaoBao | 2358 | 451 | 1394 | 376 |
| Taxi | 868 | 862 | 412 | 707 |
| StackOverflow | 2306 | 651 | 1409 | 469 |

Table 13: Training cost of the scale predictor in ScaleDiff.

| Dataset | Memory (MB) | Training Time (s) |
|---|---|---|
| Amazon | 4392 | 2530 |
| Retweet | 2822 | 1889 |
| TaoBao | 674 | 2161 |
| Taxi | 410 | 2274 |
| StackOverflow | 678 | 2600 |

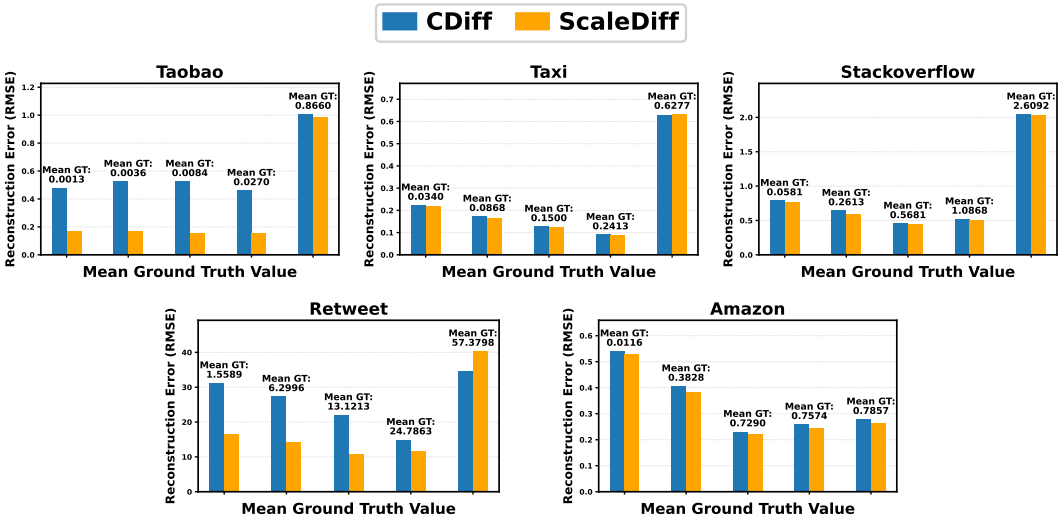



Figure 8: Reconstruction Loss for SCALEDIFF, and CDIFF for the TAOBAO, TAXI, STACKOVERFLOW, RETWEET, and AMAZON datasets.

## V ERROR ANALYSIS ON STACKOVERFLOW

To better understand the increased $\text{RMSE}_e$ on the StackOverflow dataset, we performed an additional analysis beyond the metrics reported in the paper. $\text{RMSE}_e$ measures the discrepancy between the aggregate event-type distributions of the predicted and ground-truth sequences by comparing the counts of each event type over the prediction horizon. It does not evaluate whether individual events are predicted at the correct positions. To investigate whether the higher $\text{RMSE}_e$ corresponds to poorer event predictions, we additionally computed the event-type accuracy, which compares the predicted event type at each prediction step with the corresponding ground-truth event type over the 20-step prediction horizon. Although this metric was not included in the paper, it serves as a useful diagnostic tool. On the StackOverflow dataset, ScaleDiff achieves slightly higher event-type accuracy than CDiff (0.3368 vs. 0.3324), despite having a higher $\text{RMSE}_e$. This indicates that ScaleDiff improves local event-level reconstruction, while the increase in $\text{RMSE}_e$ is primarily due to differences in the overall event-type histogram rather than incorrect temporal alignment. This behavior is expected for StackOverflow because it contains 22 event types with a highly imbalanced

Table 14: Sensitivity analysis of $\kappa_l$ on different real-world datasets.

| Dataset | $\kappa_l$ | OTD ↓ | RMSE Type ↓ | RMSE Time ↓ | SMAPE ↓ |
|---|---|---|---|---|---|
| Taobao | 0.1 | $42.725 \pm 0.280$ | $2.476 \pm 0.036$ | $0.427 \pm 0.005$ | $149.001 \pm 1.094$ |
| | 0.2 | $42.446 \pm 0.120$ | $2.462 \pm 0.049$ | $0.421 \pm 0.005$ | $147.268 \pm 1.596$ |
| | 0.4 | $42.181 \pm 0.253$ | $2.451 \pm 0.060$ | $0.415 \pm 0.004$ | $144.502 \pm 0.956$ |
| | 0.6 | $42.205 \pm 0.288$ | $2.452 \pm 0.068$ | $0.410 \pm 0.004$ | $141.867 \pm 1.050$ |
| | 0.8 | $\mathbf{42.081 \pm 0.349}$ | $\mathbf{2.443 \pm 0.060}$ | $\mathbf{0.405 \pm 0.002}$ | $\mathbf{139.708 \pm 1.712}$ |
| Taxi | 0.1 | $18.945 \pm 0.374$ | $1.071 \pm 0.056$ | $0.277 \pm 0.003$ | $76.823 \pm 0.565$ |
| | 0.2 | $18.834 \pm 0.385$ | $1.033 \pm 0.055$ | $0.277 \pm 0.003$ | $76.682 \pm 0.566$ |
| | 0.4 | $18.911 \pm 0.624$ | $1.024 \pm 0.066$ | $0.277 \pm 0.004$ | $76.856 \pm 1.453$ |
| | 0.6 | $19.145 \pm 0.769$ | $1.060 \pm 0.050$ | $0.277 \pm 0.005$ | $76.949 \pm 1.629$ |
| | 0.8 | $\mathbf{18.767 \pm 0.374}$ | $\mathbf{1.013 \pm 0.056}$ | $\mathbf{0.275 \pm 0.003}$ | $\mathbf{76.631 \pm 0.814}$ |
| Stack Overflow | 0.1 | $39.280 \pm 0.226$ | $1.545 \pm 0.019$ | $1.015 \pm 0.019$ | $87.632 \pm 0.502$ |
| | 0.2 | $39.217 \pm 0.302$ | $1.553 \pm 0.021$ | $1.013 \pm 0.023$ | $87.482 \pm 0.397$ |
| | 0.4 | $39.048 \pm 0.286$ | $1.542 \pm 0.026$ | $0.996 \pm 0.010$ | $86.950 \pm 0.176$ |
| | 0.6 | $38.983 \pm 0.292$ | $1.547 \pm 0.027$ | $0.987 \pm 0.009$ | $86.661 \pm 0.229$ |
| | 0.8 | $\mathbf{38.918 \pm 0.556}$ | $\mathbf{1.501 \pm 0.057}$ | $\mathbf{0.983 \pm 0.012}$ | $\mathbf{86.492 \pm 0.185}$ |
| Amazon | 0.1 | $37.479 \pm 0.237$ | $\mathbf{2.151 \pm 0.032}$ | $0.346 \pm 0.002$ | $78.844 \pm 0.551$ |
| | 0.2 | $37.491 \pm 0.300$ | $2.156 \pm 0.047$ | $0.346 \pm 0.002$ | $79.119 \pm 0.707$ |
| | 0.4 | $37.439 \pm 0.226$ | $2.153 \pm 0.033$ | $0.345 \pm 0.002$ | $78.807 \pm 0.638$ |
| | 0.6 | $37.379 \pm 0.235$ | $2.157 \pm 0.033$ | $0.345 \pm 0.003$ | $78.241 \pm 0.356$ |
| | 0.8 | $\mathbf{37.345 \pm 0.228}$ | $2.155 \pm 0.020$ | $\mathbf{0.344 \pm 0.002}$ | $\mathbf{78.203 \pm 0.672}$ |
| Retweet | 0.1 | $59.100 \pm 0.324$ | $2.951 \pm 0.075$ | $21.246 \pm 0.067$ | $86.153 \pm 0.277$ |
| | 0.2 | $59.109 \pm 0.212$ | $2.913 \pm 0.065$ | $21.216 \pm 0.028$ | $86.001 \pm 0.163$ |
| | 0.4 | $59.022 \pm 0.215$ | $2.924 \pm 0.072$ | $21.172 \pm 0.038$ | $85.728 \pm 0.115$ |
| | 0.6 | $58.929 \pm 0.266$ | $2.919 \pm 0.082$ | $21.147 \pm 0.039$ | $85.547 \pm 0.178$ |
| | 0.8 | $\mathbf{58.832 \pm 0.278}$ | $\mathbf{2.901 \pm 0.060}$ | $\mathbf{21.074 \pm 0.054}$ | $\mathbf{85.252 \pm 0.349}$ |

long-tailed distribution. While two event types account for roughly two-thirds of all events, many classes occur very infrequently. Consequently, small deviations in predicting these rare event types can noticeably alter the aggregate type counts, which $\text{RMSE}_e$ penalizes, even when more individual event types are predicted correctly. This interpretation is further supported by the comparative prediction performance in the Taxi dataset, where the event-type distribution is simpler (10 event types) and dominated by two frequent classes. In this setting, improvements in local reconstruction also translate to improvements in the global type distribution, leading to gains in both $\text{RMSE}_e$ (1.131→ 1.005) and event-type accuracy (0.4621→0.4845). Therefore, the higher $\text{RMSE}_e$ on StackOverflow should be interpreted as a distribution-matching issue rather than evidence of poorer event-level prediction quality.

## W LIMITATIONS

Our proposed framework is primarily designed for datasets exhibiting significant variance in temporal event distributions. Consequently, its advantages may be less substantial for datasets with relatively uniform temporal dynamics. Furthermore, the current approach mainly leverages temporal locality among events and does not explicitly model other structural relationships that may exist within event streams.

In addition, the proposed noising strategy is closely dependent on the underlying characteristics of the dataset, which may affect its generalizability across domains with substantially different temporal behaviors. Our current study also focuses predominantly on time-specific properties, while potentially informative type-specific characteristics remain unexplored. Investigating whether similar adaptive strategies can be designed based on event types or other modalities constitutes an important direction for future work.

Finally, the present framework is developed specifically within the DDPM paradigm. Extending the proposed structure-aware noising mechanism to other classes of diffusion models remains an open research challenge.