Spatio-Temporal Graph Neural Point Process for Traffic Congestion Event Prediction

Guangyin Jin[1], **Lingbo Liu**[2], **Fuxian Li**[3],**Jincai Huang**[1]

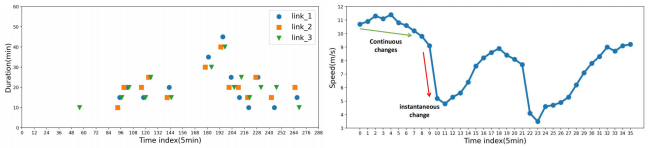
[1] School of System Engineering, National University of Defense Technology
[2] Department of Computer Sciences, The Hong Kong Polytechnic University
[3] Department of Electronic Engineering, Tsinghua University
jinguangyin18@nudt.edu.cn, liulingbo918@gmail.com, fuxianli2020@163.com, jincaihuang@nudt.edu.cn

Abstract

Traffic congestion event prediction is an important yet challenging task in intelligent transportation systems. Many existing works about traffic prediction integrate various temporal encoders and graph convolution networks (GCNs), called spatio-temporal graph-based neural networks, which focus on predicting dense variables such as flow, speed and demand in time snapshots, but they can hardly forecast the traffic congestion events that are sparsely distributed on the continuous time axis. In recent years, neural point process (NPP) has emerged as an appropriate framework for event prediction in continuous time scenarios. However, most conventional works about NPP cannot model the complex spatio-temporal dependencies and congestion evolution patterns. To address these limitations, we propose a spatio-temporal graph neural point process framework, named STGNPP for traffic congestion event prediction. Specifically, we first design the spatio-temporal graph learning module to fully capture the long-range spatio-temporal dependencies from the historical traffic state data along with the road network. The extracted spatio-temporal hidden representation and congestion event information are then fed into a continuous gated recurrent unit to model the congestion evolution patterns. In particular, to fully exploit the periodic information, we also improve the intensity function calculation of the point process with a periodic gated mechanism. Finally, our model simultaneously predicts the occurrence time and duration of the next congestion. Extensive experiments on two real-world datasets demonstrate that our method achieves superior performance in comparison to existing state-of-the-art approaches.

Introduction

Traffic congestion is one of the most serious problems in urban management, which is associated with more than 60% world-wide traffic accidents (Jain, Sharma, and Subramanian 2012). Since traffic congestion is a continuous process from generation to dissipation, each individual congestion event can be defined by two core elements: occurrence time and duration. Therefore, it is meaningful to predict when the next congestion event will occur and how long it will last for improving the traffic management and scheduling.



(a) The occurrence time and duration of traffic congestion in 24 hours (b) The continous change and instantaneous change tend of link speed

Figure 1: Example of the traffic congestion features and link speed trends from the Beijing dataset we adopted in this paper. In sub-figure (a), we select the traffic congestion statistics of three neighbor links (link 1 and link 3 are both adjacent to link 2) on 12 May 2021 to visualize the occurrence time and duration of traffic congestion in 24 hours. In sub-figure (b), we select the speed of link 1 from 7.am to 10.am on 12 May 2021 to visualize the change trend.

In recent years, many works have made some breakthroughs in intelligent transportation, especially the traffic flow prediction (Yu, Yin, and Zhu 2018; Li et al. 2018; Wu et al. 2019; Fang et al. 2021; Song et al. 2020; Zheng et al. 2020; Li and Zhu 2021). Most of these works adopt the architecture of spatio-temporal graph-based neural networks to capture the spatio-temporal correlations, and predict the states of links in future time slots by the homogeneous features in historical regular sequential time slots. Although this framework can fully exploit the spatio-temporal information to improve prediction, it is difficult to handle the traffic congestion event prediction task. There are at least two disadvantages in predicting congestion event: 1) The traditional traffic prediction framework only model dense variables on the road such as speed, not sparse ones such as congestion events. 2) Most traditional traffic prediction framework (Jin et al. 2022; Yu, Yin, and Zhu 2018; Wu et al. 2019; Jin et al. 2020; Fang et al. 2021; Song et al. 2020; Li and Zhu 2021; Wang, Zhang, and Tsui 2021; Jin et al. 2021) can only support the prediction in the given future time window (e.g., the next one hour), which is difficult to flexibly predict congestion occurring in arbitrary time intervals.

Neural point process is an appropriate framework for sparse event prediction in continuous-time scenarios (Shchur et al. 2021). However, this framework is still difficult to be adopted in traffic congestion event prediction directly. There are still two challenges as follows: 1) **How to effectively capture the spatio-temporal dependencies in road networks?** The traffic congestion can propagate in the spatial scale over time, thus each link could be affected by

the links to which they are adjacent. The periodic information could also bring significant impacts on the occurrence pattern of traffic congestion. For example, during the peak hours (e.g., 7.am$\sim$ 9.am, 5.pm$\sim$ 7.pm), congestion occurs more frequently, but less frequently during other periods. As shown in Fig. 1(a), the congestion frequency is higher during peak hours, and the patterns of occurrence time and duration of traffic congestion on adjacent links are similar. However, most previous works about neural point process (Mei and Eisner 2017; Du et al. 2016; Zuo et al. 2020; Zhang et al. 2020; Omi, Aihara et al. 2019; Xiao et al. 2019) have not fully captured spatio-temporal information for traffic congestion prediction. 2) **How to effectively model the continuous and instantaneous temporal dynamics simultaneously for each road?** The trend of the road states (e.g., speed) is a hybrid mode with continuous and instantaneous changes. When there is no congestion, the road states change gently, but when the congestion occurs, the road conditions could change instantaneously, as shown in Fig. 1(b). This makes the congestion prediction greatly different from some other event forecasting such as StackOverflow, 911 Calls and Electrical Medical Records in previous works (Du et al. 2016; Zuo et al. 2020; Xiao et al. 2019), because these events are completely discrete, they do not have the continuously dynamics characteristics similar to road conditions.

To address the problems above, we propose a novel model named Spatio-Temporal Graph Neural Point Process (STGNPP) for traffic congestion event prediction. To be specific, we introduce Transformer and Graph Convolution Network (GCN) to jointly capture the spatio-temporal dependencies from traffic states data. Then we extract the contextual link representations to incorporate with congestion event information for modeling the history of the point process. To encode the hidden evolution patterns of each road, we present a novel continuous Gated Recurrent Unit (GRU) layer with neural flow architecture. In addition, considering the effect of periodic patterns on congestion events, we propose a fully connected network with periodic gated mechanism to calculate the intensity function of the point process. Based on the learned intensity function, we compute the likelihood function of traffic congestion events to support the next prediction. Our main contributions in this paper are summarized as follows:

- To the best of our knowledge, it is the first work to propose spatio-temporal graph neural point process for traffic congestion event prediction. In particular, our model can simultaneously predict when the next traffic congestion will happen and how long it will last.
- We take account into the continuous and instantaneous dynamics in road networks, thus propose continuous GRU to model the sequential congestion events. And we also improve the calculation of intensity function by involving the periodic information.
- We conduct extensive experiments on two real-world traffic datasets. The experimental results demonstrates that our proposed model significantly outperforms than other methods. In addition, we also perform the case studies to further demonstrate the practical value of our model.

Related Works

Traffic Prediction

In recent years, many deep learning algorithms based on spatio-temporal graph modeling have been widely introduced in traffic prediction. Most of them integrated various spatial graph convolution networks (GCNs) and temporal learning modules to extract the complex spatio-temporal dependencies from the structural data. STGCN (Yu, Yin, and Zhu 2018) is the first work to combine the GCN with 1D CNN for spatio-temporal learning. ASTGCN (Guo et al. 2019) involved the attention mechanism based on STGCN. Both DCRNN (Li et al. 2018) and T-GCN (Zhao et al. 2019) integrated the gated recurrent unit (GRU) and GCN for traffic prediction. To capture long-range temporal dependencies, both STGNN (Wang et al. 2020) and GMAN (Zheng et al. 2020) employed self-attention mechanism. Graph WaveNet (Wu et al. 2019), AGRNN (Bai et al. 2020) and DMSTGNN (Han et al. 2021) proposed adaptive graph convolution to enhance the spatial representation from the predefined graphs. To capture the continuous spatio-temporal dynamics, STGODE (Fang et al. 2021) first combined the neural ODE with the normal GCN. However, most previous works about traffic prediction can only predict the sequential data in regular time snapshots and fixed-length time window. This framework is hard to be directly adopted in traffic congestion prediction because congestion may be absent or sparsely distributed over a fixed-length time window.

Neural Point Process

Neural point process has been widely applied in event forecasting of different domains such as electronic medical records (Enguehard et al. 2020), social web (Okawa et al. 2019; Zhang, Lipani, and Yilmaz 2021) and mobility (Wu, Cheng, and Sun 2021; Zhu et al. 2021a). RMTPP (Du et al. 2016) first adopted RNN to encode the historical sequential events to obtain the intensity function. Mei et al. (Mei and Eisner 2017) proposed an exponential decay based continuous LSTM to model the event sequences. Zuo et al. (Zuo et al. 2020) introduced Transformer as the encoder for long-term sequential event learning. To address calculation of integral term for likelihood optimization, Omi et al. (Omi, Aihara et al. 2019) proposed to model the cumulative intensity function. However, these previous works only focus on the temporal point process. To consider the spatio-temporal dynamics, a few spatio-temporal neural point process models (Zhu et al. 2021b; Zhou et al. 2021; Chen, Amos, and Nickel 2020) have been proposed in recent years. However, these works can only handle the applications in the spatio-temporal continuous scenarios, which can not be introduced into the traffic congestion event prediction because the traffic networks are discrete in the spatial scale.

Preliminaries

Task Definition

Given a road network with N links $V(|V| = N)$, it can be defined as a graph $\mathcal{G} = (V, E, A)$. E denotes the set of edges, whose connections between different links are characterized by the adjacency matrix A. The traffic states $\mathcal{X}_n$

(eg., link speed) on each link V_n are dense features in the snapshots of certain time granularity. The sequential congestion events on each link V_n can be defined as a finite set $\mathcal{S}_n = \{s_{n,i}\}(i = 1, 2, \ldots, |\mathcal{S}_n|)$, where $|\mathcal{S}_n|$ denotes the length of the set. In this paper, each congestion event can be defined as a two-element tuple $s_{n,i} = \langle t_{n,i}, d_{n,i} \rangle$, where $t_{n,i}$ and $d_{n,i}$ respectively denote the occurrence time and the duration of the i_{th} congestion event on link V_n. Given a fixed-length historical time window T for each sample, the traffic congestion event prediction task aims to predict the occurrence time and duration of the next congestion event based on the historical congestion events and traffic states.

Point Process Definition

The point process is one type of stochastic process to simulate the sequential events in a given observation time interval $[0, T]$. The process can be characterized by the conditional intensity function $\lambda(t|H_t)$, which represents the intensity function of events at time point t depended on the historical sequential events H_t up to t. The computation of intensity function can be given as:

$$\lambda(t|H_t) = \lim_{\Delta t \to 0} \frac{P(\text{one event occurs in } [t, t + \Delta t)|H_t)}{\Delta t}, \quad (1)$$

When the conditional intensity function and the time points $\{t_1, t_2, \ldots, t_i\}$ of the historical events are given, the probability density function can be obtained as follows:

$$p(t_{i+1}|t_1, t_2, \ldots, t_i) = \lambda(t_{i+1}|H_{t_{i+1}}) \exp\left\{-\int_{t_i}^{t_{i+1}} \lambda(t|H_t)dt\right\}, \quad (2)$$

where the exponential term in the above equation denotes the probability that no events occur in the time interval $[t_i, t_{i+1})$. In addition, we can also obtain the probability density function to observe an event sequence $\{t_i\}_{i=1}^n$, which is defined as follows:

$$p(\{t_i\}_{i=1}^n) = \prod_{i=1}^{n} \lambda(t_i|H_{t_i}) \exp\left\{-\int_0^{\tau} \lambda(t|H_t)dt\right\}. \quad (3)$$

where τ is the inter-event time that illustrates the time interval between two different events. Many previous related works (Du et al. 2016; Omi, Aihara et al. 2019; Shchur, Biloš, and Günnemann 2019; Zhang et al. 2020) directly use the inter-event time as the basic feature of each event. Note that, the most crucial part of the point process is the intensity function, which is difficult to characterize in real-world applications. Hence, neural networks can be a fruitful tool to approximate it.

Our Model

The overview of our proposed model STGNPP is illustrated in Fig. 2. The initial input data contains four parts: road network, historical traffic states, spatio-temporal indexes and congestion event information. The road network and historical traffic states are fed into spatio-temporal graph learning module to obtain the spatio-temporal hidden representation. The contextual link representation can be extracted from spatio-temporal hidden representation according to the spatio-temporal indexes. And then we integrate the contextual link representation and congestion event information for congestion event learning module. Finally, our model outputs the occurrence time and duration of the next congestion at the same time.

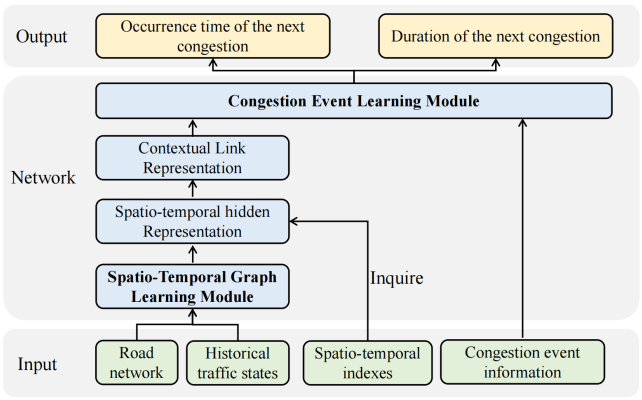


Figure 2: The overview of STGNPP

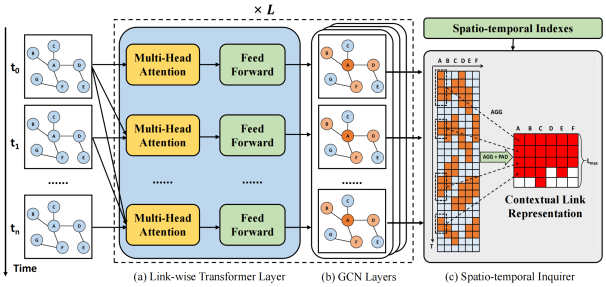


Figure 3: The detailed architecture of the spatio-temporal graph learning module. (a) is the link-wise Transformer layer to capture the long-range temporal dependencies. (b) is the graph convolution layer to further extract the spatial dependencies. (c) is the spatio-temporal inquirer that can select the corresponding hidden representations according to the spatio-temporal indexes. The spatio-temporal indexes characterize when and where the historical traffic congestion events occupied (time slots painted orange). Then we aggregate the latent representations for each congestion event to obtain the contextual link representations.

Spatio-Temporal Graph Learning Module

First, we adopt a fully connected layer to map the historical traffic states into high-dimensional representation $Z \in R^{N \times T \times D}$. N, T and D represent the number of links, the length of time window and hidden dimension respectively. The detail architecture of the spatio-temporal graph learning module is illustrated in Fig. 3.

Link-Wise Transformer Layer Since we need larger time window to include more historical congestion events, we have to learn the long-range temporal dependencies of the traffic states. Compared with temporal convolution networks (Yu and Koltun 2016) and recurrent neural networks (Cho et al. 2014; Hochreiter and Schmidhuber 1997),

Transformer (Vaswani et al. 2017) is a more powerful architecture to capture the long-range dependencies. The framework of link-wise Transformer layer adopted in our model is illustrated in Figure 3(a). Note that, the Transformer layer is weight-sharing for different links. To characterize the sequential relations more explicitly, We employ trigonometric functions-based position encoding method (Vaswani et al. 2017) in this case. The core architecture in the Transformer layer is the self-attention network. We pass the input data into it and compute the attention output by:

$$S = M_D(SoftMax(\frac{Q \cdot K^T}{\sqrt{D}}) \cdot V), \tag{4}$$

$$Q = W_Q \cdot Z,\ K = W_K \cdot Z,\ V = W_V \cdot Z, \tag{5}$$

where Q, K and V respectively denote the query, key, and value matrices obtained by three linear transformations $W_Q.W_K, W_V \in R^{D\times D}$, where D denotes the dimension of the self-attention network. $Q \cdot K^T \in R^{N\times T\times T}$ denotes the dot product over the T dimension. M_D represents the mask operation that sets the value of the upper triangle of the attention matrix to 0. This can prevent information of future time steps from being exploited by past time steps. To stabilize the fitting capability of self-attention network, we also employ the multi-head attention mechanism, similar to (Vaswani et al. 2017). Then we pass the attention output into the two-layer position-wise feed-forward neural network, generating the sequential hidden representation of each time snapshot:

$$H = W_{F2} \cdot (ReLU(W_{F1} \cdot S + b_{F1})) + b_{F2}, \tag{6}$$

$$h(t_i) = H(:, i, :), \tag{7}$$

where $W_{F1} \in R^{D\times D}$, $W_{F2} \in R^{D\times D}$, $b_{F1} \in R^D$ and $b_{F2} \in R^D$ are learnable parameters of the two-layer feed-forward neural network. $H \in R^{N\times T\times D}$ is the output of the Transformer layer and $h(t_i) \in R^{N\times D}$ is the hidden representation at time snapshot t_i.

Graph Convolution Layer In addition to the temporal dependencies of each link, adjacent links may influence each other, thus we employ the graph convolution layer to capture the spatial dependencies. In this case, we involve the adaptive learnable matrices to characterize the spatial relations that cannot be represented by the predefined adjacency matrix. And we adopt the simple graph convolution operation (Kipf and Welling 2017) with mix-hop aggregation, which is defined as follows:

$$\hat{A} = A + SoftMax(ReLU(\alpha_1 \dot{\alpha}_2^T)), \tag{8}$$

$$H_i = \sigma(\hat{A} \cdot H_{i-1} \cdot \Theta_i), \tag{9}$$

$$H_g = SumPooling(H_1, H_2, \ldots, H_i), \tag{10}$$

where A is the normalized predefined adjacency matrix, $\alpha_1, \alpha_2 \in R^{N\times D'} (D' << N)$ are two learnable matrices to adaptively characterize the latent spatial relations through the back propagation process, Θ_i is the learnable weight for each convolution layer, $H_i \in R^{N\times T\times d}$ is the output from each layer of GCN. Note that, the initial input of the GCN layer, $H_0 \in R^{N\times T\times D}$ is the hidden states from Transformer layer and $H_g \in R^{N\times T\times D}$ is the output of the sum pooling operation from multi-hop hidden states.

Spatio-temporal Inquirer After obtaining the spatio-temporal graph hidden representations from the Transformer layer and graph convolution layer, in order to characterize the latent features of congestion events, we select the corresponding hidden representations according to the spatio-temporal indexes, as shown in Fig. 3(c). Each spatio-temporal index contains two elements v_n and t_s. Specifically, v_n denotes the index of target link and t_s denotes the collection of relative time periods of the congestion event in the historical time window $[0, T]$. From these indexes, we can easily obtain the corresponding hidden representations, named contextual link representations, which is defined as:

$$t_s = [t_i, t_{i+1}, \ldots, t_{i+L_s}], t_{i+L_s} < T,$$

$$H_c = \|^N [PAD(\|^{L_e} [AGG(H_g(v_n, t_s, :))])]. \tag{11}$$

where $H_g(v_n, t_s, :) \in R^{L_s\times D}$ denotes the latent representations of one congestion event on link v_n, L_s is the number of time slots occupied by one congestion event and L_s varies for different congestion events. $AGG(\cdot)$ denotes the aggregation function on the dimension L_s and we use the simple sum function in this case. $\|^{L_e}$ denotes the concentrate operation for the congestion event sequences on each link. Note that, for different links, the lengths of congestion event sequences L_e are different, thus we need to adopt the zero padding operation $PAD(\cdot)$ to ensure that the sequences are of equal length L_{max}. $\|^N$ denotes the concentration for the latent representations on different links and the size of output H_c is $N \times L_{max} \times D$.

Congestion Event Learning Module

From the spatio-temporal learning module, we can obtain the contextual link representation H_c for sequential congestion events. Therefore, we define the congestion event representation as follows:

$$H_e = W_e \cdot [H_c, d_e] + b_e, \tag{12}$$

where $d_e \in R^{N\times L_{max}\times 1}$ denotes the historical duration of each congestion event after zero padding, $H_e \in R^{N\times L_{max}\times D}$ is the output representation.

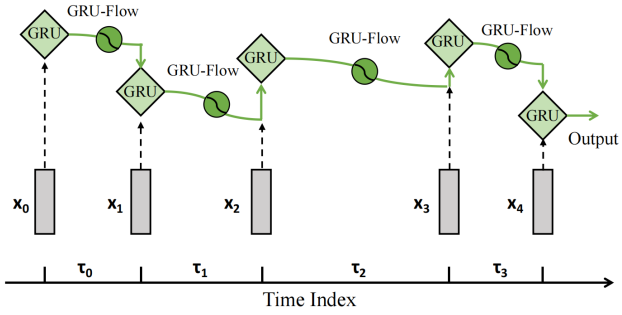


Figure 4: The detailed structure of the continuous GRU layer in our model. The green squares denote the moment when the congestion event occurred. The green curves and arrows represent continuous and instantaneous changes in the hidden representation of link states, which are learned by GRU-flow and discrete GRU, respectively. The grey strip denotes the input contextual information at each time step.

Continuous GRU Layer Different from some other scenarios such as StackOverflow, 911 Calls and Electrical Medical Records in previous related works (Du et al. 2016; Zuo et al. 2020; Xiao et al. 2019), the congestion event prediction task is more special: the traffic state for each link is a combination of continuous changes and instantaneous changes. Hence, it is necessary to take these two situations into account. Some continuous system modeling methods are introduced in previous works such as continuous LSTM (Mei and Eisner 2017), neural ODE (Chen et al. 2018), RNN-ODE (Rubanova, Chen, and Duvenaud 2019) and GRU-ODE (Brouwer et al. 2019). However, these traditional methods especially the ODE-based models suffered from huge computational overhead, which could be a serious problems for real-world applications. Motivated by a recent work (Biloš et al. 2021), we can replace the ODE operations by a more efficient architecture, neural flows, which is defined as:

$$F(t, x) = x + \phi(t) \cdot \Gamma(t, x), \tag{13}$$

where $\phi(t)$ is a continuous function that satisfies two properties: i) $\phi(0) = 0$ and ii) $|\phi(t)| < 1$, $\Gamma(t, x)$ is an arbitrary contractive neural network. In this case, we choose *Tanh*, the simplest function for $\phi(t)$. This architecture can transfer the residual connection to the continuous one and reduce the computational overhead brought by ODESolver. Inspired by GRU-ODE (Brouwer et al. 2019), the normal GRU Cell can be converted to the continuous type. The equations of normal GRU are defined as:

$$\begin{aligned} r_t &= \sigma(W_r x_t + U_r h_{t-1} + b_r), \\ z_t &= \sigma(W_z x_t + U_z h_{t-1} + b_z), \\ g_t &= \tanh(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h), \\ h_t &= z_t \odot h_{t-1} + (1 - z_t) \odot g_t, \end{aligned} \tag{14}$$

From the above equations, we can derive its differential form:

$$\Delta h_t = h_t - h_{t-1} = (1 - z_t) \odot (g_t - h_{t-1}), \tag{15}$$

Hence, we propose to combine the differential form of GRU with neural flows, named GRU-flow. We obtain the continuous GRU cell to characterize the hidden state between two adjacent time steps in our model as follows:

$$\begin{aligned} F(\tau_i, h_i^l) = h_i^l + \phi(W_\tau \cdot \tau_i) \cdot (1 - z([\tau_i, h_i^l])) \\ \odot (g([\tau_i, h_i^l]) - h_i^l), \end{aligned} \tag{16}$$

where τ_i is the inter-event time at each step, $W_\tau \in R^{1 \times D}$ is a transformation weight to ensure dimensional consistency, h_i^l denotes the l_{th} layer's hidden state at each step, $[\cdot]$ denotes the concentrate operation. The initial input of the continuous GRU cell is the event embedding at step i. For capturing the instantaneous dynamics, we directly adopt the discrete GRU cell, which is as follows:

$$h_{i+1}^0 = GRUCell(H_e[i + 1, :], F(\tau_i, h_i^l)), \tag{17}$$

where $GRUCell(\cdot)$ denotes the discrete GRU cell. The two input elements of discrete GRU cell are respectively the hidden state from the GRU-flow $F(\tau_i, h_i^l)$ and the contextual link representation $H_e[i + 1, :]$ at the next time step. The output of the discrete GRU cell can be treated as the initial input for the GRU-flow at the next time step. The detailed structure of our continuous GRU Layer is show in Fig. 4 and the hidden states from the discrete GRU cell are fed into the intensity function network.

Intensity Function Network To approximate the distribution of inter-event time, many traditional models (Du et al. 2016; Zuo et al. 2020) optimized the logarithmic form of the probability density function in eq. (3), called log-likelihood function, which is defined as:

$$\log L(\{\tau\}) = \sum_i \left[\log \lambda(\tau | h_i) - \int_0^\tau \lambda(\tau | h_i) d\tau \right]. \tag{18}$$

where τ denotes the inter-event time, h_i denotes the event hidden state at i_{th} step. However, the integral term can only be approximated by interpolation or Monte Carlo method (Zuo et al. 2020). To address this problem, the cumulative intensity function is introduced to reformulate the log-likelihood function and the intensity function is computed by a multi-layer fully connected network. Since the congestion frequency could vary by peak hours or non-peak hours, as discussed in the introduction section, the intensity function could be impacted by the periodic patterns. To characterize the effect of periodic patterns, we first propose a periodic gated unit to adjust the intensity function. The mathematical form can be defined as the basic intensity term multiplied by the periodic gate term, which is as follows:

$$\begin{aligned} \log L(\{t_i\}) = \sum_i \left[\log \left\{ \frac{\partial}{\partial \tau} \lambda(\tau = t_{i+1} - t_i | h_i) \right\} - \lambda(\tau | h_i) \right], \\ \lambda(\tau | h_i) = f_l^+([h_i, \tau]) \odot \sigma(f_p([P_i^d, P_i^w])). \end{aligned} \tag{19}$$

where $f_l^+(\cdot)$ denotes the fully connected layer for computing the basic intensity function, $f_p(\cdot)$ denotes the fully connected layer for periodic gated unit, P_i^d, P_i^w respectively denote the time of day and the day of week of the i_{th} event.

Optimization and Prediction

As a multi-task learning framework, our model simultaneously optimizes the negative log-likelihood of the probability density function of the inter-event time and the absolute error of the duration prediction during the training phase:

$$L = \sum_i \left[\lambda(\tau | h_i) - \log \left\{ \frac{\partial}{\partial \tau} \lambda(\tau | h_i) \right\} \right] + \alpha \| f_d(h_i) - d_{i+1} \|, \tag{20}$$

where $f_d(\cdot)$ denotes the fully connected layer for duration prediction of the next traffic congestion, α denotes the trade-off ratio. We set α as 1 by default.

During the inference phase, the duration prediction of the next congestion can be obtained directly by the output of $f_d(\cdot)$. For the occurrence time of the next congestion event, our model can also efficiently generate the prediction in the following way. Given the historical congestion event occurred at time point $\{t_1, t_2, \ldots, t_i\}$, the predictive probability density function $p^*(t | t_1, t_2, \ldots, t_i)$ of the time point t_{i+1} of the next congestion event is calculated from eq. (2). In this case, we utilize the median time point t_{i+1}^* of the probability density distribution p^* to estimate t_{i+1}. Since the integral

of the intensity function over $[t_i, t_{i+1}]$ follows the exponential distribution with mean 1, derived by directly integrating eq. (2), we use the median relation of the exponential distribution, $\lambda(t^*_{i+1} - t_i|h_i) = \log(2)$, to estimate t^*_{i+1}. Further, we can adopt the bisection method to efficiently obtain t^*_{i+1}. Given the interval $[T_{min}, T_{max}]$, if the intensity function $\lambda(\tau|h_i)$ is less than $log(2)$, then we reassign the lower bound as $T_{min} = (T_{min} + T_{max})/2$. In contrast, if $\lambda(\tau|h_i)$ is greater than $log(2)$, then we reassign the upper bound as $T_{max} = (T_{min}+T_{max})/2$. Through such multiple rounds of loop calculation, we will get a relatively accurate prediction $t^*_{i+1} = (T_{min} + T_{max})/2$.

Experiments

Datasets and Settings

We evaluate our model on Beijing dataset and Chengdu dataset which are collected from the mobile application databases, as shown in Table 1. Each dataset is chronologically split with 60% for training, 20% for validation and 20% for testing. We utilize the traffic states, congestion event information and spatio-temporal indexes in the last six hours to predict the occurrence time and duration of the next congestion event. Note that, we only employ the link speed data with condition labels as traffic states data. The condition label is a binary variable that describes whether the road is congested or not for each time slot. And the congestion events are also collected based on the condition labels. Congestion event information includes inter-event times, duration and periodic features. Our model is implemented by Pytorch 1.5 with NVIDIA TESLA V100 GPU. We set the number of stacks of Transformer and GCN as 2, the number of self-attention heads as 4 in Transformer layer, the number of GCN layers as 2 and the number of flow layers in continuous GRU as 2 by default. The dimension of hidden representations in our model is set as 64 and the dimension of random matrix for adaptive graph is set as 10. We set the optimizer as Adam with learning rate 0.001 and the batch size as 16. During the training phase, we employ the early stopping strategy with tolerance 20 for 100 epochs. During the inference phase, we set the lower bound and the upper bound of the time interval as 0 and 360 (min) respectively.

Table 1: Dataset description and statistics.

Datasets	#links	#Congestion Events	Time Range	Time Slot
Beijing	573	249464	5/12/2021 - 11/12/2021	5min
Chengdu	435	204768	5/12/2021 - 11/12/2021	5min

Overall Performance

We compare our model with ten state-of-art baselines, which can be divided into three categories. **Simple models:** Historical Average (HA), Gradient Boosting Decision Tree (GBDT) (Ye et al. 2009) and GRU (Cho et al. 2014). **Spatio-temporal graph-based models:** DCRNN (Li et al. 2018), Graph WaveNet (Wu et al. 2019) and STGODE (Fang et al. 2021). **Neural point process-based models:** NHTPP (Mei and Eisner 2017), RMTPP (Du et al. 2016), THPP (Zuo et al. 2020) and FNN-TPP (Omi, Aihara et al. 2019). For the simple models, we only use the traffic congestion event information as the input features to directly predict the next congestion events. For the spatio-temporal graph-based models, similar to most traffic flow prediction tasks, we employ the core architectures of spatio-temporal graph networks for spatio-temporal representation learning and predict the link condition labels in the next six hours (the six-hour time window is long enough to guarantee coverage of next congestion events). And then we can obtain the occurrence time and duration of the next traffic congestion event for each link according to the predicted link condition labels in the future time window. For neural point process-based models, we use congestion event information for point process modeling to predict the next congestion events.

The evaluation metrics are mean absolute errors (MAE), mean absolute percentage errors (MAPE) and negative log-likelihood (NLL). Note that, we use NLL, MAE and MAPE to evaluate the prediction of occurrence time. For the prediction of duration, we only use MAE and MAPE. In subsequent experiments, MAE-t and MAPE-t denote the metrics for occurrence time while MAE-d and MAPE-d denote the metrics for duration. From the results in Table 2, we can observe that our model STGNPP consistently outperforms the sub-optimal baselines with around 10% improvements in terms of all metrics on the two datasets, which demonstrates the superiority of our proposed method. From the experimental results, both of spatio-temporal graph-based baselines and neural point process-based baselines are significantly stronger than simple baselines. This is because the simple baselines can neither model inter-event dependencies nor capture spatio-temporal hidden information. The spatio-temporal graph-based baselines can fully exploit spatio-temporal information but neglect to model the congestion events, while neural point process-based baselines are the opposite. By contrast, our proposed model can not only fully exploit traffic-related spatio-temporal dependencies, but also model the sequence of congestion events, thereby outperforming other methods with a large margin.

Ablation Study

We conduct ablation study on both of the two datasets to evaluate the effectiveness of each crucial module in our model. As shown in Table 3, we compared STGNPP with following ablation variants: 1) *GWNPP*, which replaces the spatio-temporal graph learning module in our model with that in Graph WaveNet 2) *w/o GCN*, which removes all the GCN layers from our models 3) *w/o Trans*, which removes the Transformer layers from our model. 4) *w/o GRU*, which replaces the continuous GRU with the fully connected networks. 5) *w/o continuous*, which replaces the continuous GRU with the discrete GRU. 6) *w/o Gated*, which removes the periodic gated unit from the intensity function networks.

From Table 3, we can find that our complete model STGNPP outperforms all the ablation variants. Since Graph WaveNet is a recognized excellent model in traffic prediction, the superiority of our model to *GWNPP* suggests that Transformer can capture long-term dependencies better than temporal convolutions in traffic congestion prediction scenarios. Our model significantly outperforms *w/o GCN* and *w/o Trans* can demonstrate that capturing either spatial and

Table 2: Performance comparison of baseline models and STGNPP. We run all machine learning algorithms five times with different random seeds and calculated the mean and standard deviation. Note that, only our model and neural point process-based models need to compute NLL and the sub-optimal results are marked by the asterisk.

Method	Beijing					Chengdu				
	NLL	MAPE-t(%)	MAE-t(min)	MAPE-d(%)	MAE-d(min)	NLL	MAPE-t(%)	MAE-t(min)	MAPE-d(%)	MAE-d(min)
HA	/	48.87	25.45	54.62	18.70	/	63.41	36.35	62.71	42.67
GBDT	/	42.63 ± 1.12	23.81 ± 0.56	33.60 ± 0.83	16.21 ± 0.25	/	52.78 ± 1.32	32.39 ± 0.72	34.87 ± 0.51	38.13 ± 0.63
GRU	/	41.37 ± 1.24	23.18 ± 0.67	31.88 ± 0.92	15.75 ± 0.34	/	44.84 ± 1.58	30.67 ± 0.69	33.18 ± 0.57	37.45 ± 0.68
DCRNN	/	38.43 ± 1.07	19.36 ± 0.52	26.85 ± 0.62	14.63 ± 0.28	/	42.51 ± 1.24	27.76 ± 0.55	31.92 ± 0.48	35.75 ± 0.53
Graph WaveNet	/	37.82 ± 0.32	19.01 ± 0.16	26.39 ± 0.25	14.46 ± 0.10	/	41.89 ± 0.91	27.48 ± 0.22	31.81 ± 0.13	35.79 ± 0.16
STGODE	/	38.75 ± 0.73	19.68 ± 0.32	26.51 ± 0.41	14.55 ± 0.27	/	43.05 ± 1.02	28.27 ± 0.34	32.03 ± 0.25	36.69 ± 0.29
NHTPP	5.84 ± 0.04	38.86 ± 0.14	19.17 ± 0.11	26.29 ± 0.20	14.47 ± 0.11	5.90 ± 0.06	42.73 ± 0.65	28.10 ± 0.21	32.07 ± 0.17	36.59 ± 0.15
RMTPP	6.02 ± 0.07	38.72 ± 0.18	19.64 ± 0.13	27.31 ± 0.19	14.52 ± 0.09	6.18 ± 0.06	42.89 ± 0.59	28.65 ± 0.23	32.16 ± 0.15	36.45 ± 0.17
THPP	5.75 ± 0.04	38.16 ± 0.12	19.27 ± 0.14	25.83* ± 0.16	14.38* ± 0.11	5.94 ± 0.05	42.76 ± 0.41	28.39 ± 0.20	31.78 ± 0.12	35.86 ± 0.13
FNN-TPP	5.46* ± 0.05	37.54* ± 0.09	18.86* ± 0.11	26.72 ± 0.13	14.41 ± 0.08	5.68* ± 0.04	41.38* ± 0.37	27.13* ± 0.14	31.82* ± 0.08	35.75* ± 0.06
STGNPP (ours)	**4.87 ± 0.03**	**34.16 ± 0.05**	**16.95 ± 0.08**	**24.02 ± 0.10**	**13.15 ± 0.05**	**5.02 ± 0.02**	**37.54 ± 0.21**	**24.52 ± 0.10**	**29.83 ± 0.06**	**32.94 ± 0.08**

Table 3: Ablation experiments.

Dataset	Model&Variants	NLL	MAPE-t(%)	MAE-t	MAPE-d(%)	MAE-d
Beijing	STGNPP	**4.87**	**34.16**	**16.95**	**24.02**	**13.15**
	GWNPP	4.98	35.48	17.12	24.31	13.43
	w/o GCN	5.06	36.08	18.17	24.83	13.92
	w/o Trans	5.17	36.46	18.45	25.01	14.10
	w/o GRU	5.43	36.95	18.68	24.97	14.32
	w/o Continous	5.01	36.12	18.38	24.75	13.94
	w/o Gated	5.05	36.28	18.29	24.31	13.89
Chengdu	STGNPP	**5.02**	**37.54**	**24.52**	**29.83**	**32.94**
	GWNPP	5.11	38.23	24.91	30.28	33.36
	w/o GCN	5.36	39.30	26.34	31.86	35.43
	w/o Trans	5.58	39.82	27.10	31.76	35.62
	w/o GRU	5.73	40.16	27.89	32.12	35.81
	w/o Continous	5.24	39.13	27.65	31.79	34.97
	w/o Gated	5.18	38.96	26.11	32.06	34.95

temporal dependencies of traffic states in road networks can be beneficial for congestion event prediction. The variant *w/o GRU* can illustrate that the continuous GRU layer can capture the contextual correlations in historical congestion event sequences for more accurate prediction. To further investigate the effectiveness of continuous modeling of GRU flow architectures, we compare STGNPP with *w/o continuous*. The results indicate that either the instantaneous dynamics or the continuous dynamics of the contextual link representations have a non-negligible effect on the prediction of traffic congestion event. We also compare our model with the *w/o gated* to investigate the effectiveness of periodic gated unit in intensity function network. The comparison results reflect the importance of involving periodic information into intensity function.

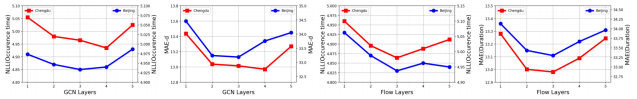
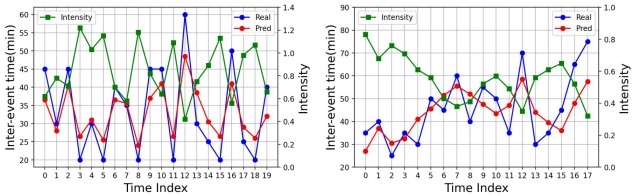


Figure 5: Studies on hyper-parameters.

Parameters Study

Since some parameters could significantly affect the learning capability, we conduct parameter study to further investigate the effectiveness of our model. We select the number of GCN layers l_g and the number of flow layers l_f in continuous GRU, because l_g is critical for the spatial dependencies learning and l_f is critical for modeling the continuous sequential congestion events. The experimental results are shown in Fig. 5. We can find that the best setting for l_g is 3 for the two datasets. When l_g increases, the NLL of Beijing and duration MAE of the two datasets become worse. The reason may be that the over-smoothing problem of GCNs limits the improvement of performance. For l_f, the best value for the two datasets is also 3. With the increase of l_f, the NLL of Chengdu and duration MAE of the two datasets become worse. This is because too many layers for continuous GRU could cause the over-fitting problem.



(a) Prediction Curve on Beijing dataset (a) Prediction Curve on Chengdu dataset

Figure 6: Visualization of the learned intensity functions attached to prediction curves on two datasets.

Case Study

To further investigate how our model characterizes the timing features by the learned intensity function, we visualize the curves of intensity functions (green lines), the predicted inter-event times (red lines) and the ground truths (blue lines) of the most congested link on the May 12, 2021 for the two datasets respectively. As shown in Fig. 6, we can find that the peaks and troughs of the intensity function curves are opposite to the peaks and troughs of the ground truths and prediction curves. This means the learned intensity functions can characterize the trends of traffic congestion event: The larger the value of intensity function, the greater the probability of congestion occurrence, so the smaller the time interval between two different congestion events.

Conclusion

We propose a novel spatio-temporal graph neural point process framework for traffic congestion event prediction. In this paper, we give a first attempt to utilize the spatio-temporal graph to incorporate with neural point process for traffic congestion event modeling and we also take account into some important traffic-related characteristics such as periodic features, continuous and instantaneous dynamics, to improve the inter-event dependencies learning. We conduct extensive experiments on two large-scale real-world datasets and the experimental results demonstrate the superiority of our model compared with other traditional methods.

References

Bai, L.; Yao, L.; Li, C.; Wang, X.; and Wang, C. 2020. Adaptive Graph Convolutional Recurrent Network for Traffic Forecasting. In *NIPS*.

Biloš, M.; Sommer, J.; Rangapuram, S. S.; Januschowski, T.; and Günnemann, S. 2021. Neural Flows: Efficient Alternative to Neural ODEs. *NIPS* 34.

Brouwer, E. D.; Simm, J.; Arany, A.; and Moreau, Y. 2019. GRU-ODE-Bayes: continuous modeling of sporadically-observed time series. In *NIPS*, 7379–7390.

Chen, R. T.; Amos, B.; and Nickel, M. 2020. Neural Spatio-Temporal Point Processes. In *ICLR*.

Chen, R. T.; Rubanova, Y.; Bettencourt, J.; and Duvenaud, D. 2018. Neural ordinary differential equations. In *NIPS*, 6572–6583.

Cho, K.; van Merrienboer, B.; Çaglar Gülçehre; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *EMNLP*.

Du, N.; Dai, H.; Trivedi, R.; Upadhyay, U.; Gomez-Rodriguez, M.; and Song, L. 2016. Recurrent marked temporal point processes: Embedding event history to vector. In *Proceedings of the 22nd ACM SIGKDD conference*, 1555–1564.

Enguehard, J.; Busbridge, D.; Bozson, A.; Woodcock, C.; and Hammerla, N. 2020. Neural temporal point processes for modelling electronic health records. In *Machine Learning for Health*, 85–113. PMLR.

Fang, Z.; Long, Q.; Song, G.; and Xie, K. 2021. Spatial-Temporal Graph ODE Networks for Traffic Flow Forecasting. In *Proceedings of the 27th ACM SIGKDD Conference*, 364–373.

Guo, S.; Lin, Y.; Feng, N.; Song, C.; and Wan, H. 2019. Attention Based Spatial-Temporal Graph Convolutional Networks for Traffic Flow Forecasting. *Proceedings of the AAAI Conference* 33: 922–929.

Han, L.; Du, B.; Sun, L.; Fu, Y.; Lv, Y.; and Xiong, H. 2021. Dynamic and Multi-faceted Spatio-temporal Deep Learning for Traffic Speed Forecasting. In *Proceedings of the 27th ACM SIGKDD Conference*, 547–555.

Hochreiter, S.; and Schmidhuber, J. 1997. Long Short-term Memory. *Neural computation* 9: 1735–80.

Jain, V.; Sharma, A.; and Subramanian, L. 2012. Road traffic congestion in the developing world. In *Proceedings of the 2nd ACM Symposium on Computing for Development*, 1–10.

Jin, G.; Cui, Y.; Zeng, L.; Tang, H.; Feng, Y.; and Huang, J. 2020. Urban ride-hailing demand prediction with multiple spatio-temporal information fusion network. *Transportation Research Part C: Emerging Technologies* 117: 102665.

Jin, G.; Li, F.; Zhang, J.; Wang, M.; and Huang, J. 2022. Automated Dilated Spatio-Temporal Synchronous Graph Modeling for Traffic Prediction. *IEEE Transactions on Intelligent Transportation Systems* .

Jin, G.; Yan, H.; Li, F.; Huang, J.; and Li, Y. 2021. Spatio-Temporal Dual Graph Neural Networks for Travel Time Estimation. *arXiv preprint arXiv:2105.13591* .

Kipf, T. N.; and Welling, M. 2017. Semi-supervised classification with graph convolutional networks. In *Proc. of ICLR*.

Li, M.; and Zhu, Z. 2021. Spatial-Temporal Fusion Graph Neural Networks for Traffic Flow Forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence* 35(5): 4189–4196. URL https://ojs.aaai.org/index.php/AAAI/article/view/16542.

Li, Y.; Yu, R.; Shahabi, C.; and Liu, Y. 2018. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *Proc. of ICLR*.

Mei, H.; and Eisner, J. M. 2017. The Neural Hawkes Process: A Neurally Self-Modulating Multivariate Point Process. *NIPS* 30.

Okawa, M.; Iwata, T.; Kurashima, T.; Tanaka, Y.; Toda, H.; and Ueda, N. 2019. Deep mixture point processes: Spatio-temporal event prediction with rich contextual information. In *Proceedings of the 25th ACM SIGKDD Conference*, 373–383.

Omi, T.; Aihara, K.; et al. 2019. Fully Neural Network based Model for General Temporal Point Processes. *NIPS* 32: 2122–2132.

Rubanova, Y.; Chen, R. T.; and Duvenaud, D. 2019. Latent ODEs for irregularly-sampled time series. In *NIPS*, 5320–5330.

Shchur, O.; Biloš, M.; and Günnemann, S. 2019. Intensity-Free Learning of Temporal Point Processes. In *ICLR*.

Shchur, O.; Türkmen, A. C.; Januschowski, T.; and Günnemann, S. 2021. Neural Temporal Point Processes: A Review. *arXiv preprint arXiv:2104.03528* .

Song, C.; Lin, Y.; Guo, S.; and Wan, H. 2020. Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting. In *Proceedings of the AAAI Conference*, volume 34, 914–921.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention Is All You Need. *CoRR* abs/1706.03762. URL http://arxiv.org/abs/1706.03762.

Wang, T.; Zhang, Z.; and Tsui, K.-L. 2021. PSTN: Periodic Spatial-temporal Deep Neural Network for Traffic Condition Prediction. *arXiv preprint arXiv:2108.02424* .

Wang, X.; Ma, Y.; Wang, Y.; Jin, W.; Wang, X.; Tang, J.; Jia, C.; and Yu, J. 2020. Traffic Flow Prediction via Spatial Temporal Graph Neural Network. In *Proceedings of The Web Conference 2020*, 1082–1092. ISBN 9781450370233.

Wu, Y.; Cheng, Z.; and Sun, L. 2021. Individual Mobility Prediction via Attentive Marked Temporal Point Processes. *arXiv preprint arXiv:2109.02715* .

Wu, Z.; Pan, S.; Long, G.; Jiang, J.; and Zhang, C. 2019. Graph WaveNet for Deep Spatial-Temporal Graph Modeling. In *Proc. of IJCAI*.

Xiao, S.; Yan, J.; Farajtabar, M.; Song, L.; Yang, X.; and Zha, H. 2019. Learning time series associated event sequences with recurrent point process networks. *IEEE transactions on neural networks and learning systems* 30(10): 3124–3136.

Ye, J.; Chow, J.-H.; Chen, J.; and Zheng, Z. 2009. Stochastic gradient boosted distributed decision trees. In *Proceedings of the 18th ACM CIKM Conference*, 2061–2064.

Yu, B.; Yin, H.; and Zhu, Z. 2018. Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting. 3634–3640.

Yu, F.; and Koltun, V. 2016. Multi-scale context aggregation by dilated convolutions. In *ICLR*.

Zhang, Q.; Lipani, A.; Kirnap, O.; and Yilmaz, E. 2020. Self-attentive hawkes process. In *ICML*, 11183–11193. PMLR.

Zhang, Q.; Lipani, A.; and Yilmaz, E. 2021. Learning Neural Point Processes with Latent Graphs. In *Proceedings of the Web Conference 2021*, 1495–1505.

Zhao, L.; Song, Y.; Zhang, C.; Liu, Y.; Wang, P.; Lin, T.; Deng, M.; and Li, H. 2019. T-gcn: A temporal graph convolutional network for traffic prediction. *IEEE Transactions on Intelligent Transportation Systems* 21(9): 3848–3858.

Zheng, C.; Fan, X.; Wang, C.; and Qi, J. 2020. GMAN: A Graph Multi-Attention Network for Traffic Prediction. *Proceedings of the AAAI Conference* 34: 1234–1241.

Zhou, Z.; Yang, X.; Rossi, R.; Zhao, H.; and Yu, R. 2021. Neural Point Process for Learning Spatiotemporal Event Dynamics. *arXiv preprint arXiv:2112.06351* .

Zhu, S.; Ding, R.; Zhang, M.; Van Hentenryck, P.; and Xie, Y. 2021a. Spatio-temporal point processes with attention for traffic congestion event modeling. *IEEE Transactions on Intelligent Transportation Systems* .

Zhu, S.; Li, S.; Peng, Z.; and Xie, Y. 2021b. Imitation Learning of Neural Spatio-Temporal Point Processes. *IEEE Transactions on Knowledge and Data Engineering* .

Zuo, S.; Jiang, H.; Li, Z.; Zhao, T.; and Zha, H. 2020. Transformer hawkes process. In *ICML*, 11692–11702. PMLR.