World Wide Web (2023) 26:1649–1665
https://doi.org/10.1007/s11280-022-01095-2

# GCMT: a graph-contextualized multitask spatio-temporal joint prediction model for cellular trajectories

**Yu Sang[1] · Yuan Xu[2] · Bo Ning[3] · Zhenping Xie[1]**

Received: 13 June 2022 / Revised: 26 July 2022 / Accepted: 6 August 2022 /
Published online: 12 October 2022


## Abstract

Spatio-temporal joint prediction aims to simultaneously predict the next location and the corresponding switch time for a cellular trajectory. An accuracy prediction requires not only sequential information but also spatio-temporal context information. Although existing methods can utilize trajectory modeling to support the joint prediction, they fail to learn the complicated geographical influence, temporal dependencies and various context information. To this end, we propose a graph-contextualized multitask learning method for spatio-temporal joint prediction. Specially, to model each location's spatio-temporal dependencies, a graph embedding module is adopted to jointly capture the geographical influence and temporal cyclic effect by embedding three relational graphs (i.e., location-location, location-region, and location-time) into a shared low dimensional space. Moreover, considering the impact of traffic-related contexts on trajectory movement, we design a traffic encoder to model the dynamic of traffic flows, which comprises several spatio-temporal blocks combining temporal gated CNN with spatial graph convolution. In addition, a context-attention layer is proposed to fuse trajectory sequential information and traffic information based on various background factors. Finally, GCMT is evaluated on two real-world datasets to demonstrate its advantages.

**Keywords** Spatio-temporal joint prediction · Multitask learning · Cellular trajectory · Deep learning

Yu Sang and Yuan Xu contributed equally to this work.

This article belongs to the Topical Collection: *Special Issue on Spatiotemporal Data Management and Analytics for Recommenders*
Guest Editors: Shuo Shang, Xiangliang Zhang and Panos Kalnis

✉ Bo Ning
ningbo@dlmu.edu.cn

Zhenping Xie
xiezp@jiangnan.edu.cn

[1] School of Artificial Intelligence and Computer Science, Jiangnan University, Wuxi, China

[2] School of Computer Science and Technology, Soochow University, Suzhou, China

[3] Dalian Maritime University, Dalian, China

## 1 Introduction

In recent years, we are witnessing an exponential growth in cellular trajectories, with the rapid development of mobile communication technology [1–3]. Spatio-temporal joint prediction is an important problem in the construction of intelligent communication system. It aims to predict the next location and the corresponding switch time for a cellular trajectory at the same time, which is a multitask joint prediction process. It is of great significance to decide when user data will be scheduled to which base station. Besides, the joint prediction benefits many applications such as efficient resources management in mobile communications, location-aware advertisements and navigation services [4].

Most existing trajectory prediction methods only focus on a single task, i.e., location prediction or time prediction. Methods of location prediction [5–7] use sequential models (e.g., HMM and RNNs) to capture trajectories' spatial mobility regularities to predict the next location. Most methods of time prediction [8] utilize temporal point process (TPP) [9] to model temporal sequence. However, these single-task methods cannot simultaneously predict these two tasks to support the multitask requirements of trajectory prediction in most real scenarios. To achieve the joint prediction, multitask learning based methods [10, 11] are proposed to jointly utilize spatio-temporal signals to capture the mutual influence among these two tasks. Unfortunately, these multitask methods merely focus on the modeling of trajectories and ignore the importance of spatio-temporal contexts (e.g., traffic-related contexts), and accordingly, cannot provide accuracy prediction.

Accurate spatio-temoral joint prediction for cellular trajectories remains to be challenging due to complicated spatio-temporal dependencies and various context information. First, trajectory movement will be affected by the spatial distribution and temporal cycle of locations, because users usually tend to visit the nearby base stations and follow periodical patterns. Aside from the above geographical influence, traffic conditions also have a great influence on trajectory movements. Intuitively, the traffic congestion may affect the trajectory's movement speed and even the choice of the next location. Thus, traffic-related contextual information should be taken into account to achieve accurate predictions. Finally, trajectory prediction also requires to consider various background factors (e.g., departure time, weekdays and weather) due to different mobility patterns. For example, the trajectories of rush hours are more likely to encounter traffic congestion and thus require more attention to traffic context information, while daily trajectories may need to pay more attention to sequential information. However, most studies merely model the sequential information of trajectories without learning the spatio-temporal contexts, which results in inaccurate predictions.

Furthermore, we find that one trajectory always has consistent travel intention, and thus the state of a trajectory point is impacted by its follow-up points' states. The next location of a trajectory may be predicted inaccurately without the information of the travel intention from the trajectory. Nevertheless, once knowing the user's destination is airport, we can guess that he will follow the airport road and thus predict the next location. Hence, a cellular trajectory's location movement is influenced by its travel intention. Meanwhile, the travel intention of a trajectory can be predicted by a sequence of location switches. However, existing multitask methods ignore the signal of travel intention and thus cannot utilize it for spatio-temporal joint prediction.

To tackle the above issues, we propose a graph-contextualized multitask learning method called GCMT for spatio-temporal joint prediction, which adds travel intention prediction as

an auxiliary task on the basis of spatio-temporal joint prediction. Specifically, we design a graph-based representation module, which constructs three relational graphs (i.e., location-location, location-region and location-time) and embeds each vertex into a shared low dimensional space. In this way, it can capture sequential effect, geographical influence and temporal cyclic effect. Then, we adopt a self-attention network to effectively model long and dense cellular trajectories. In addition, in order to capture recent traffic condition, we adopt a traffic encoder to model traffic dynamic in traffic flow data. The encoder consists of multiple ST-blocks, which combine temporal gated CNN with spatial graph convolution, to jointly learn spatial and temporal dependencies. Finally, considering the influence of various background factors (e.g., departure time, weekdays and weather), a context-attention mechanism is designed to fuse sequential information and traffic information for a more comprehensive prediction. We summarize our contributions as follows:

- We propose a graph-contextualized multitask learning method for spatio-temporal joint prediction, which integrates representation module, trajectory encoder, traffic encoder, context modeling and task-specific decoder as a whole.
- We adopt a graph-based representation module to jointly capture the sequential effect, geographical influence and temporal effect. Moreover, to learn traffic condition, a spatio-temporal block is designed to model spatial and temporal dependencies in traffic flow data.
- Extensive experiments on two real-world trajectory datasets show that our model achieves the best performance among all state-of-the-art methods.

This paper is an extension of our previous work [12], and it makes the following major improvements:

- We design a bipartite graph embedding module to embed location-location, location-region, and location-time graphs into a shared low dimensional space, so as to jointly study geographical and temporal effects of locations.
- We adopt a spatio-temporal block to learn traffic information in traffic flow data, which combines temporal gated CNN with spatial graph convolution as an integration.
- We introduce a context attention mechanism to take account of various background factors, by which our model can adaptively assign reasonable weights to sequence and traffic information.

The remainder of this paper is organized as follows. We first introduce the related work of trajectory prediction and multitask learning in Section 2. Then, several definitions are given and the research problem is formulated in Section 3. In Section 4, we present our proposed method GCMT. Finally, we conduct extensive experiments in Section 5 and conclude the paper in Section 6.

## 2 Related work

### 2.1 Trajectory prediction

As an important part in the construction of smart city, trajectories have been studied for many years [13–18]. Researches on trajectory prediction mainly focus on location prediction task [19–21] or time prediction task [22, 23]. Most location prediction studies utilize

learning techniques like RNNs [6, 7] to model sequential information to make predictions in spatial domain. Considering the importance of spatial and temporal interval information, some methods [6, 7, 24] extend the framework of RNNs. HST-LSTM [6] introduces an add operation to existing gates of LSTM to merge spatial-temporal interval information. Flashback [24] does flashbacks on past hidden states to consider historical records with similar contexts. DeepMove [5] adds an attention mechanism to GRU to learn multi-level periodicity.

In addition, a few studies [8, 23] focus on the time prediction task. RCR [8] utilizes visitors and potential visitors' historical check-ins to extract features, and adopts censored regression for time predictions. A recurrent spatio-temporal point process model [23] is further proposed to utilize TPP to improve the performance. However, they require a pre-designated next location and cannot support the time prediction for trajectories with unknown next location.

In summary, these single-task methods neglect that trajectory prediction requires both location and time prediction in most real scenarios, which cannot directly support the prediction of another task or even the joint prediction.

### 2.2 Multitask learning

Multitask learning (MTL) [25] aims to exploit meaningful information from other related learning tasks to solve multiple tasks at the same time, which has been successfully applied in many fields, such as computer vision [26]. Inspired by the success, a few multitask based methods [10, 11, 27] study the spatio-temporal joint prediction of events or POIs. RMTPP [10] combines RNN with TPP to support the joint prediction for events. ARNPP-GAT [27] uses graph attention networks to model user's long term preference and combines it with an attention-based recurrent neural point process. IRNN [11] respectively utilizes RNN to model time and event sequence. DeepJMT [28] adopts a hierarchical RNN to capture temporal patterns and mobility regularities, which extracts location's semantics, user's periodicity, and social relationships to alleviate the data sparsity problem. IAMT [12] proposes an intention-aware multitask learning method, introducing travel intention prediction as an auxiliary task on the basis of spatio-temporal joint prediction to provide long-term intentional information. However, these methods ignore the influence of spatio-temporal contexts on trajectory movement. Thus, we propose our method to model complicated spatio-temporal dependencies and various context information.

## 3 Preliminaries

**Definition 1** (Trajectory) Let $\mathcal{L} = \{l_1, l_2, \cdots, l_{n_1}\}$ denote a set of locations. A spatio-temporal point $p$ is a tuple $(l, t)$, where the location $l \in \mathcal{L}$ can refer to a base station and the positive real number $t \in \mathcal{R}^+$ presents the timestamp switching to the location $l$. A trajectory $T$ is a time-ordered spatio-temporal point sequence $T = \{p_1, p_2, \cdots, p_m\}$. Besides, we represent the time interval of two consecutive spatio-temporal points as $\tau$, i.e., $\tau_k = t_k - t_{k-1}$.

**Definition 2** (Location-Location Graph) Location-Location graph is denoted as $\mathcal{G}_{ll} = (\mathcal{L} \cup \mathcal{L}, \mathcal{E}_{ll})$. $\mathcal{L}$ is a set of locations and $\mathcal{E}_{ll}$ is a set of edges between locations. Given a time

interval $\Delta T$, for each spatio-temporal point pair $\{(l_i, t_i), (l_j, t_j)\}$ in trajectory $T$, if $0 < t_j - t_i \leq \Delta T$, there will be an edge $e_{ij}$ from $l_i$ to $l_j$. The weight $w_{ij}$ of $e_{ij}$ is defined as the number of times that $l_j$ is visited after $l_i$ in the trajectory dataset $\mathcal{T}$ within the time interval $\Delta T$.

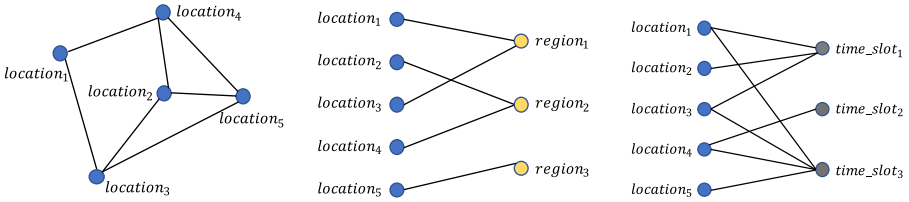
Location-Location graph, as a general graph, captures sequential information and locations' spatial distribution, which can be viewed as a bipartite graph when we set a location on one side and others on the other side. To further capture geographical and temporal effect, we construct Location-Region and Location-Time bipartite graphs as Figure 1 shows. Continuous values (i.e., geographical area and timestamps) are transformed into discrete ones due to the discrete vertices in graphs. We adopt a grid-based partition method [29, 30] and divide the geographical space into $N = \omega \times \omega$ regions $\mathcal{R}$. Besides, all timestamps are divided into a set of time slots $\mathcal{S}$ based on the hours of a day.

**Definition 3** (Location-Region Graph) Location-Region graph, denoted as $\mathcal{G}_{lr} = (\mathcal{L} \cup \mathcal{R}, \mathcal{E}_{lr})$, is a bipartite graph. $\mathcal{L}$ is a set of locations and $\mathcal{R}$ is a set of regions. $\mathcal{E}_{lr}$ is a set of edges between locations and regions. If location $l_i$ is in region $r_j$, there will be an edge $e_{ij}$ between them and the weight $w_{ij}$ is set as 1; otherwise, none.

**Definition 4** (Location-Time Graph) Location-Time graph, denoted as $\mathcal{G}_{ls} = (\mathcal{L} \cup \mathcal{S}, \mathcal{E}_{ls})$, is a bipartite graph, where $\mathcal{E}_{ls}$ is a set of weighted edges between locations and time slots. If location $l_i$ is visited at time slot $s_j$, there will be an edge $e_{ij}$ between them; otherwise, none. The weight $w_{ij}$ is set to the frequency of location $l_i$ visited during the time slot $s_j$.

**Definition 5** (Traffic Flow) Traffic flow information is described by inflow and outflow. At a given time interval, inflow is the total number of traffic flows entering a region while outflow is the total number of traffic flows leaving a region. At time $t$, we use $X_t \in \mathcal{R}^{N \times 2}$ to denote the traffic flow of $N$ regions, where $X_t[i, 0]$ and $X_t[i, 1]$ are the inflow and outflow of region $i$ at time $t$.

**Problem 1** (Spatio-temporal Joint Prediction) Given a set of trajectories $\mathcal{T} = \{T^1, T^2, \cdots, T^{|\mathbf{T}|}\}$, where $T^i = \{p_1^i, p_2^i, \cdots, p_m^i\}$ is the $i$-th trajectory, spatio-temporal joint prediction aims to predict the next spatio-temporal point $p_{m+1}^i$ of the trajectory $T^i$, including the location identification $l_{m+1}^i$ and the corresponding timestamp $t_{m+1}^i$ derived from the time interval $\tau_{m+1}^i$.




(a) Location-Location Graph (b) Location-Region Graph (c) Location-Time Graph

**Figure 1** Illustration of three relational graphs

# 4 Methodology

## 4.1 Overview

In Figure 2, we propose a graph-contextualized multitask learning model, which is made up of five parts, i.e., representation module, trajectory encoder, traffic encoder, context modeling, and task-specific decoder. In the model, we design a multitask framework which adds travel intention prediction as an auxiliary task on the basis of spatio-temporal joint prediction. Specifically, we construct three relational graphs and use graph-based embedding methods to embed each vertex into a low dimensional space. Then, we use self-attention network to model trajectories to obtain sequential information. Meanwhile, the traffic encoder utilizes spatio-temporal convolution network to capture traffic-related contexts. It is constructed by two ST-blocks, which combines temporal gated CNN with spatial graph convolution. Considering the influence of other background factors, a carefully designed context attention layer is used to adaptively assign different weights to sequential information and traffic information. Finally, the task-specific decoder makes the final predictions for each task and trains the entire network with multi-task losses.

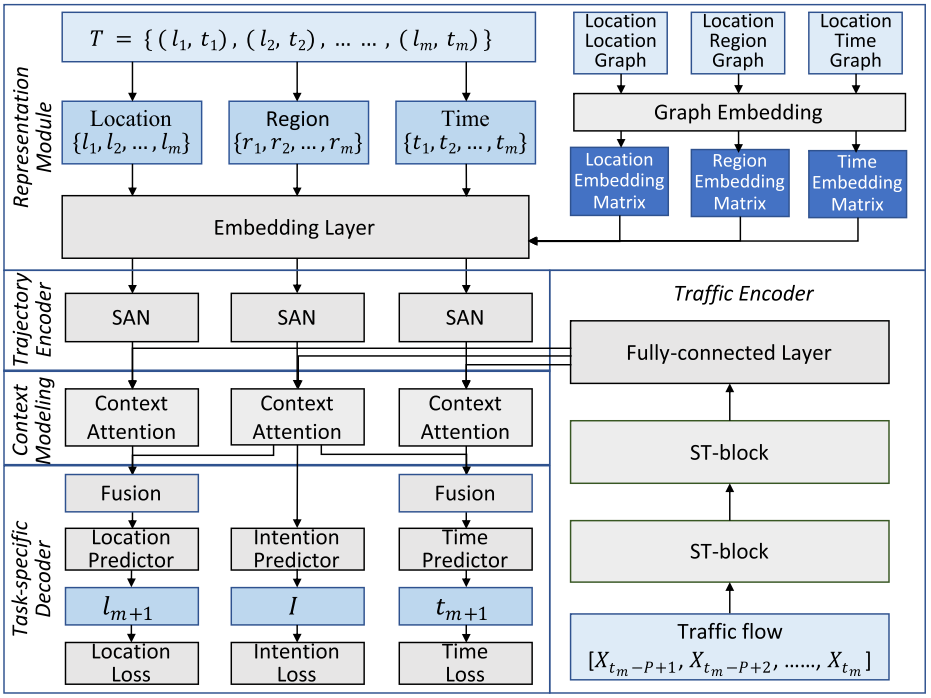



**Figure 2** Architecture of GCMT. It consists of representation module, trajectory encoder, traffic encoder, context modeling and task-specific decoder

### 4.2 Representation module

#### 4.2.1 Bipartite graph embedding

Trajectory movement is affected by complicated geographical influences and temporal cycle dependencies among locations. For example, at 12 a.m., users usually head to restaurants for lunch and tend to visit nearby locations. Thus, we construct location-location graph, location-region graph and location-time graph to jointly learn spatio-temporal information of locations. We adopt a probabilistic model [31] to learn embeddings of heterogeneous graph nodes.

Given a bipartite graph $\mathcal{G}_{AB} = (\mathcal{V}_A \cup \mathcal{V}_B, \mathcal{E}_{AB})$, the probability of observing a bipartite edge $e_{ij}$ between $v_i \in \mathcal{V}_A$ and $v_j \in \mathcal{V}_B$ is computed as:

$$p(e_{ij} = 1) = \frac{1}{1 + \exp(-(\overrightarrow{v}_j^{\mathrm{T}} \cdot \overrightarrow{v}_i))} \tag{1}$$

where $\overrightarrow{v}_i$ and $\overrightarrow{v}_j$ are embedding vectors of $v_i$ and $v_j$ respectively. However, (1) is only applicable to a bipartite edge. For a weighted bipartite graph $\mathcal{G}_{AB}$, its likelihood can be computed by:

$$O_{AB} = -\sum_{e_{ij} \in \mathcal{E}_{AB}} w_{ij} \log p(e_{ij} = 1) - \sum_{e_{ij} \in \overline{\mathcal{E}}_{AB}} \gamma_{ij} \log(1 - p(e_{ij} = 1)) \tag{2}$$

Where $\overline{\mathcal{E}}_{AB}$ is a set of negative edges, and $\gamma_{ij}$ is the weight of negative edge, $w_{ij}$ is the weight of its positive edge.

Directly optimizing (2) will result in high computational cost, since it requires to calculate massive negative edges. Thus, we adopt a negative sampling method [32] to sample multiple negative edges for each positive edge, and the weights of negative edges are assumed to be equal to the weight of their corresponding positive edge. We reformulate the objective function as:

$$O_{AB} = -\sum_{e_{ij} \in \mathcal{E}_{AB}} w_{ij} \Big[ \log p(e_{ij} = 1) + \sum_{k=1}^{M} \mathrm{E}_{v_k \sim P_n(v)} \log(1 - p(e_{ik} = 1)) \Big] \tag{3}$$

where $M$ is the number of negative edges, $P_n(v) \propto d_v^{3/4}$, and $d_v$ is the degree of vertex $v$. We use asynchronous stochastic gradient algorithm (ASGD) [33] to optimize (3). To collectively embed our three relational graphs into a shared low-dimension space, we minimize the sum of all objective functions.

$$O = O_{ll} + O_{lr} + O_{ls} \tag{4}$$

All edges in $\mathcal{E}_{ll}$, $\mathcal{E}_{lr}$ and $\mathcal{E}_{lt}$ are firstly merged together. Considering that the weights of edges between different graphs are not comparable, we adopt the joint embedding training algorithm [31] to alternatively sample from the three sets of edges to update the model. Hence, we can obtain the embedding matrices of location $M_l \in \mathcal{R}^{n_1 * d}$, region $M_r \in \mathcal{R}^{N * d}$ and time slot $M_s \in \mathcal{R}^{n_2 * d}$.

#### 4.2.2 Trajectory embedding

Both temporal and spatial sequence can provide meaningful knowledge for trajectory modeling, and meaningful mobility patterns may exist in different spatial granularity. Thus,

we embed not only spatio-temporal multi-sequence, but also spatial multi-granularity. For trajectory $T$, a sequence of base stations $T_l = \{l_1, l_2, \cdots, l_m\}$ and a sequence of timestamps $T_t = \{t_1, t_2, \cdots, t_m\}$ can be directly obtained. Besides, we derive coarse-grained region sequence $T_r = \{r_1, r_2, \cdots, r_m\}$ from $T_l$ for spatial multi-granularity modeling. To capture temporal information, a sequence of continuous points' time interval, i.e., $T_\tau = \{\tau_1, \tau_2, \cdots, \tau_m\}$, is further obtained. To obtain sequences with fixed-length, we adopt zero-padding at the end of them, e.g., $T_l = \{l_1, l_2, \cdots, l_n\}$, where $n$ is the predefined maximum length. Then, we can retrieve the embedding of location sequence $E_l \in \mathcal{R}^{n*d}$ through $M_l \in \mathcal{R}^{n_1*d}$, where $E_{l,i} = M_{l,T_{l,i}}$, that is, selecting the specified row's vector from $M_l$ according to the base station's identification. Then, the embedding of time interval sequence $E_\tau \in \mathcal{R}^{n*d}$ and region sequence $E_r \in \mathcal{R}^{n*d}$ can be obtained in the same way.

### 4.3 Trajectory encoder

Considering that cellular trajectory has strong sequentiality and dense sampling points, self-attention network [34] is adopted to capture long-term dependencies of the whole sequence. Since multiple heads can jointly focus on different representation subspace information, we further use multi-head self-attention to encode trajectory sequences. The framework of the self-attention network (SAN) is shown as Figure 3(a). After obtaining the embedding of each sequence $E_{task}$, SAN is adopted to model the sequential information.

$$S_{task} = SAN(E_{task}) \tag{5}$$

where $task \in \{l, r, \tau\}$. Besides, we take the last point's representation of $S_{task}$ as task-specific sequential representation, i.e., $S_{l,m}$, $S_{r,m}$, $S_{\tau,m}$.

### 4.4 Traffic encoder

Trajectory movement is greatly affected by traffic condition. For example, traffic congestion will make trajectory move slower, and thus affect the state of next movement. Hence, we aim to model complex spatio-temporal dependencies in traffic flows to consider traffic-related contexts. We select traffic flows from recent $P$ time intervals, i.e., $\mathcal{X} = [X_{t_m-P+1}, X_{t_m-P+2}, \cdots, X_{t_m}]$. $\mathcal{X}$ will be fed into traffic encoder composed of two ST-blocks and a fully-connected layer. To extract spatial and temporal correlations, ST-block is formed by temporal gated CNN and spatial graph convolution as Figure 3(b) shows.

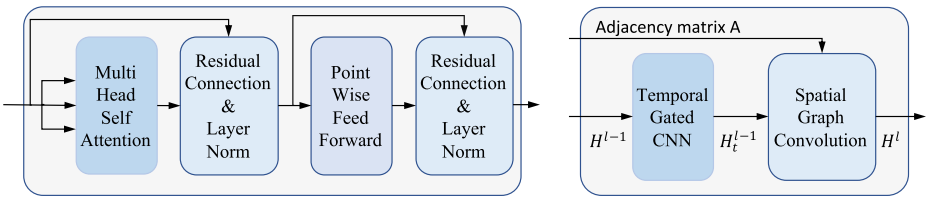



**Figure 3** The framework of SAN and ST-block

#### 4.4.1 Temporal gated CNN

Although RNNs are widely used in time series modeling, its performance is limited by the non-parallel training procedures and time-consuming iterations. Inspired by the superiority of CNN [35], we adopt temporal gated CNN to capture temporal dynamics of traffic flows. Specifically, it takes historical traffic as input $H \in \mathcal{R}^{P\times N\times c}$, where $c$ is the number of input channels. The convolution operation with two convolution kernels $\Gamma_1, \Gamma_2 \in [\mathcal{R}^{t_k\times c\times 1,o}]$ is adopted to integrate temporal information within $t_k$ steps, where $o$ is the number of output channels. These two parts are input to gating mechanism to capture long-term temporal memory. The output $H_t \in \mathcal{R}^{(P-t_k)\times N\times o}$ is computed by:

$$H_t = (\Gamma_1 \otimes H) \odot \sigma(\Gamma_2 \otimes H) \tag{6}$$

where $\odot$ denotes the element-wise product, and $\sigma$ is the sigmoid function for controlling the propagation of temporal information [36].

#### 4.4.2 Spatial graph convolution

As we all know, the current traffic condition of a region is influenced by not only its recent time period, but also its surrounding regions due to the geographical connection. Hence, graph convolution network is adopted to model the spatial dependencies between different regions.

Specifically, we treat different regions as the nodes in the graph, and the adjacency matrix $A$ is computed according to the distances among regions.

$$a_{ij} = \begin{cases} \exp\left(\frac{d_{ij}^2}{\sigma^2}\right), & if\ i \neq j\ and\ \exp\left(\frac{d_{ij}^2}{\sigma^2}\right) \geq \epsilon, \\ 0, & otherwise. \end{cases} \tag{7}$$

where $a_{ij}$ is the weight of the edge $e_{ij}$, and $d_{ij}$ is their distance. $\sigma^2$ and $\epsilon$ are thresholds to control the distribution and sparsity of matrix $A$.

To integrate the temporal information, we use $H_{t_m}$ obtained by the temporal gated CNN to initialize the representation of each region. Inspired by [37], we define the graph convolution operation in l-th layer as:

$$H_s^{(l+1)} = \sigma(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}H_s^{(l)}W^{(l)}) \tag{8}$$

where $H_s^{(l)}$ is the input of the l-th hidden layer, and $H_s^{(0)} = H_t$. $\hat{A} = A + I$, $I$ is an identity matrix. $\hat{D}$ is the diagonal node degree matrix of $\hat{A}$ and $\hat{D}_{ii} = \sum_j \hat{A}_{ij}$. $W^{(l)}$ is trainable parameters, and $\sigma(\cdot)$ is a non-linear activation function (e.g., ReLU). Since a graph convolution layer can aggregate information from 1-hop neighbors, we stack $k$ layers to expand the receptive field and gain information from k-hop neighbors.

Finally, we further use a fully-connected layer to obtain the current traffic condition $H_{t_m}$ and extract the information of current region $r_m$ to represent traffic-related contexts, i.e., $H_c = H_{t_m,r_m}$.

### 4.5 Context modeling

So far, we have obtained trajectory sequence information and traffic-related information. Intuitively, trajectories under different backgrounds will have different mobility patterns, and the effect of different information may be different. For example, traffic contexts may have a greater impact on trajectories of traffic congestion, while sequence information may

have greater effects on daily trajectories. Hence, a context attention mechanism is proposed to model the influence of background factors.

After trajectory and traffic encoders, we obtain two vectors $S_{task,m}$ and $H_c$ for each task, and we respectively project these vectors to new query vectors $p_s$ and $p_c$. Besides, we concatenate all the background factors together as context embedding $e_c$ for trajectory $T$. Then, the attention coefficients are measured by computing the similarity between query vector and $e_c$ as follows:

$$a_i = softmax(p_i e_c) = \frac{\exp(p_i e_c)}{\sum_i \exp(p_i e_c)} \tag{9}$$

where $i \in \{s, c\}$. Then we can get a new vector integrating trajectory sequence information and traffic context information based on the attention coefficients.

$$Z_{task} = \sum_i a_i p_i \tag{10}$$

where $task \in \{l, r, \tau\}$. Thus, we can effectively fuse trajectory sequential information and traffic information based on the trajectory's background factors.

### 4.6 Task-specific decoder

#### 4.6.1 Fusion layer

After obtaining the sequential information and intentional information from trajectories, an effective fusion method is required to aggregate these two information. In order to consider the importance of each information for trajectory prediction, we design a gating mechanism to ensure that the fusion representation can remain both knowledge with different proportions, and it tends to focus more on the more informative features for current movement. The gating mechanism can be written as follows,

$$\begin{aligned} G_{task,1} &= sigmoid(W_{task,1} Z_{task} + W_{task,r,1} Z_r + b_{task,1}) \\ G_{task,2} &= sigmoid(W_{task,2} Z_{task} + W_{task,r,2} Z_r + b_{task,2}) \\ G_{task} &= G_{task,1} \odot Z_{task} + G_{task,2} \odot Z_r \end{aligned} \tag{11}$$

where $W_{task,1}, W_{task,r,1}, b_{task,1}, W_{task,2}, W_{task,r,2}, b_{task,2}$ are learnable parameters, and $task \in \{l, \tau\}$.

#### 4.6.2 Prediction layer

**Next location predictor** It predicts the next location $l_{m+1}$ based on the final representation $G_l$, and the probability of $l_{m+1}$ is calculated as follows:

$$P_l = softmax(G_l W_l + b_l) \tag{12}$$

where $W_l$ and $b_l$ are learnable parameters.

**Intention predictor** Similar to the next location predictor, the probability of intention $I$ is also calculated based on the intentional representation $S_{g,m}$.

$$P_r = softmax(Z_r W_r + b_r) \tag{13}$$

**Switch time predictor** We adopt TPP to model time sequences to make time prediction. Inspired by the previous work [10], we use the output of deep neural network to calculate

the density function $f^*(t)$, which calculates the probability that next location switch occurs at time $t$ given $T$.

$$f^*(t) = \exp\{W_t^T G_\tau + W_\tau(t - t_m) + \lambda_0 + \frac{1}{W_\tau}\exp(W_t^T G_\tau + \lambda_0) - \frac{1}{W_\tau}\exp(W_t^T G_\tau + W_\tau(t - t_m) + \lambda_0)\} \tag{14}$$

where $W_t$, $W_\tau$, $\lambda_0$ are trainable parameters. The next switch time is computed as $t_{m+1} = \int_{t_m}^{\infty} t f^*(t)$, which can be calculated with numerical integration [38].

#### 4.6.3 Loss layer

**Location loss** We apply multi-class logarithmic loss function cross entropy as our location loss function, which is calculated by:

$$\mathcal{L}_l = -\hat{l}_{m+1} \log P_l \tag{15}$$

where $\hat{l}_{m+1}$ is the one-hot represented ground truth and $P_l$ is the predicted probability distribution of each base station.

**Time loss** We define the loss function for the next switch time prediction task based on the definition of TPP as follows:

$$\mathcal{L}_\tau = -\log f^*(t_{m+1}) \tag{16}$$

where $f^*(t_{m+1})$ is the density function, which is calculated by (14).

**Intention loss** Considering that traditional multi-class loss function treats multiple categories as independent individuals, and thus cannot capture the complicated spatial associations among different regions, such as distance and direction consistency. Thus, a distribution-aware loss function is designed to effectively capture the intention information.

$$\mathcal{L}_r = \frac{\sum_{i \in topk\_r} D_{r_j,i} \cdot P_{r,i}}{\sum_{i \in topk\_r} P_{r,i}} \tag{17}$$

where *topk_r* is a set of regions at the top k of $P_r$; $D_{r_j,i}$ is the distance between the ground truth $r_j$ and $r_i$; $j = m + m * 0.5$, $m$ is current trajectory's length; $P_{r,i}$ is the predicted probability of future goal at the $i$-th region.

**Multi-task loss** The entire network is trained by minimizing the weighted loss sum of location prediction, time prediction and intention prediction.

$$\mathcal{L}(\Theta) = \beta_\tau \mathcal{L}_\tau + \beta_r \mathcal{L}_r + (1 - \beta_\tau - \beta_g)\mathcal{L}_l \tag{18}$$

where $\Theta$ are all learnable parameters; $\beta_\tau$ and $\beta_r$ are hyper-parameters for tuning relative influence of $\mathcal{L}_\tau$ and $\mathcal{L}_r$.

## 5 Experiments

### 5.1 Datasets

Table 1 shows the statistics of two real cellular trajectory datasets, which are respectively collected on June 29, 2019 in Chengdu and on June 9, 2019 in Xiamen. A base station is

**Table 1** Statistics of two datasets

| Datasets | # Users | # Base stations | # Records | # Trajectories |
|---|---|---|---|---|
| Hangzhou | 10825 | 13902 | 403867 | 10825 |
| Xiamen | 763 | 7255 | 683972 | 33583 |

viewed as a location that can provide signals to the area around it. If a user enters the area, the base station identifies the user and records the corresponding time. A trajectory is a time-ordered spatio-temporal point sequence in a taxi's trip order. We remove the trajectories with less than 5 points and take half length of each trajectory as its input length.

### 5.2 Baselines

To evaluate the effectiveness of GCMT for two main tasks, we respectively compare it with single-task methods for location prediction (STRNN, DeepMove, HST-LSTM, and Flashback), single-task methods for time prediction (Avg, THP) and multi-task methods (RMTPP, IRNN, ARNPP-GAT, IAMT).

- **STRNN** [7]. It uses distance-specific and time-specific matrices to extend standard RNN framework for location prediction.
- **DeepMove** [5]. It combines a historical attention mechanism with GRU to predict the next location over lengthy and sparse trajectories.
- **HST-LSTM** [6]. It adopts an add operation on three existing gates of LSTM to consider geographic distance and time interval.
- **Flashback** [24]. It does flashbacks on past hidden states of RNN to consider historical points with similar context for next location prediction.
- **Avg**. It takes the average value of historical spatio-temporal points' time interval as the predicted result.
- **THP** [39]. It introduces a transformer-based architecture into Hawkes process to make time predictions for event sequence.
- **RMTPP** [10]. It utilizes RNN to construct intensity function of recurrent point process, which is used for events' spatio-temporal joint prediction.
- **IRNN** [11]. It adopts two unshared RNNs to respectively model the event and time sequence, and combines with TPP for the joint prediction.
- **ARNPP-GAT** [27]. It utilizes GAN to model user long-term preferences and adopts attention-based recurrent point process for next check-in inference. GAN is removed due to the lack of user social graph in our datasets.
- **IAMT** [12]. It adds travel intention prediction as an auxiliary task to provide long-term mobility information for spatio-temporal joint prediction.

### 5.3 Parameter setup and metrics

All methods are implemented with PyTorch. We randomly select 70% of our datasets for training, and the remaining 10% and 20% for validation and test. For parameter setup, $M$ is set as 5 and kernel size of CNN is 3. We select the previous 6 intervals traffic flow data (60 minutes). Other settings are the same as IAMT [12]. To evaluate the performance for location prediction, we use four widely metrics: Accuracy (ACC), Mean Reciprocal

**Table 2** Performance comparison results for next location prediction

| Datasets | Hangzhou | | | | Xiamen | | | |
|---|---|---|---|---|---|---|---|---|
| Method | ACC | MRR | Recall | macro-F1 | ACC | MRR | Recall | macro-F1 |
| STRNN | 0.0471 | 0.0833 | 0.0254 | 0.0118 | 0.3143 | 0.4665 | 0.1836 | 0.1479 |
| DeepMove | 0.0480 | 0.0970 | 0.0222 | 0.0138 | 0.3375 | 0.4914 | 0.1708 | 0.1434 |
| HSTLSTM | 0.0476 | 0.0886 | 0.0220 | 0.0115 | 0.3156 | 0.4686 | 0.1818 | 0.1482 |
| Flashback | 0.0513 | 0.0956 | 0.0261 | 0.0203 | 0.3411 | 0.4883 | 0.2062 | 0.1737 |
| RMTPP | 0.0483 | 0.0961 | 0.0228 | 0.0150 | 0.3213 | 0.4784 | 0.1770 | 0.1498 |
| IRNN | 0.0559 | 0.0986 | 0.0253 | 0.0146 | 0.3422 | 0.4901 | 0.2069 | 0.1731 |
| ARNPP | 0.0619 | 0.1077 | 0.0339 | 0.0215 | 0.3497 | 0.5186 | 0.2103 | 0.1745 |
| IAMT | 0.0721 | 0.1302 | 0.0424 | 0.0279 | 0.3783 | 0.5458 | 0.2321 | 0.1988 |
| GCMT-E | 0.0767 | 0.1507 | 0.0449 | 0.0291 | 0.3852 | 0.5624 | 0.2398 | 0.2044 |
| GCMT-T | 0.0776 | 0.1499 | 0.0429 | 0.0288 | 0.3879 | 0.5641 | 0.2386 | 0.2031 |
| GCMT-C | 0.0795 | 0.1511 | 0.0456 | 0.0297 | 0.3897 | 0.5658 | 0.2419 | 0.2064 |
| GCMT | **0.0844** | **0.1536** | **0.0472** | **0.0311** | **0.3974** | **0.5712** | **0.2496** | **0.2153** |

The bold entries shows the experimental results of our model, which outperforms all baselines

Rank (MRR), Recall, and macro-F1. Besides, Mean Absolute Error (MAE) and Root Mean-Squared Error (RMSE) are used to evaluate models for time prediction. To be fair, each method is run three times and the average value is taken as the final result.

## 5.4 Performance comparison

### 5.4.1 Next location prediction

From the results in Table 2, we can find that our model achieves the best performance among all baselines. Specifically, traditional single-task based methods (STRNN, DeepMove, HST-LSTM, and Flashback) perform worse than our model GCMT. The reason is that the mutual influence of spatio-temporal signals in a trajectory is neglected in these methods. In addition, GCMT performs better than multi-task methods (RMTPP, IRNN, ARNPP-GAT, and IAMT), which is due to the fact that they ignore complex spatio-temporal contexts. The superior of our model proves that trajectory movement is also influenced by complicated spatio-temporal dependencies and various context information. Besides, Hangzhou dataset has more base stations and fewer records than Xiamen, and thus it is more sparse and difficult to train a good model to learn trajectory movement, which results in the worse performance in Hangzhou.

### 5.4.2 Next switch time prediction

Table 3 shows that our model achieves the best performance. In detail, GCMT improves the performance of single-task methods (Avg and THP), because it jointly utilizes spatio-temporal signals for collaborative trajectory modeling. Besides, although multi-task baselines (RMTPP, IRNN, ARNPP-GAT and IAMT) utilize both spatio-temporal signals, they ignore the travel speed of a trajectory will be affected by spatio-temporal dependencies of locations and traffic-related contextual information, thus performing worse than GCMT.

**Table 3** Performance comparison results for next switch time prediction

| Datasets | Hangzhou | | Xiamen | |
|---|---|---|---|---|
| Method | MAE | RMSE | MAE | RMSE |
| Avg | 2.45 | 3.88 | 2.76 | 4.97 |
| THP | 2.36 | 3.22 | 2.51 | 3.51 |
| RMTPP | 2.25 | 3.24 | 2.38 | 3.27 |
| IRNN | 2.22 | 3.15 | 2.32 | 3.21 |
| ARNPP-GAT | 2.17 | 3.12 | 2.29 | 3.16 |
| IAMT | 2.08 | 3.04 | 2.21 | 3.03 |
| GCMT-E | 1.95 | 2.88 | 2.09 | 2.98 |
| GCMT-T | 2.06 | 2.95 | 2.18 | 3.02 |
| GCMT-C | 1.97 | 2.91 | 2.12 | 3.01 |
| GCMT | **1.89** | **2.84** | **2.05** | **2.96** |

The bold entries shows the experimental results of our model, which outperforms all baselines

### 5.5 Ablation study

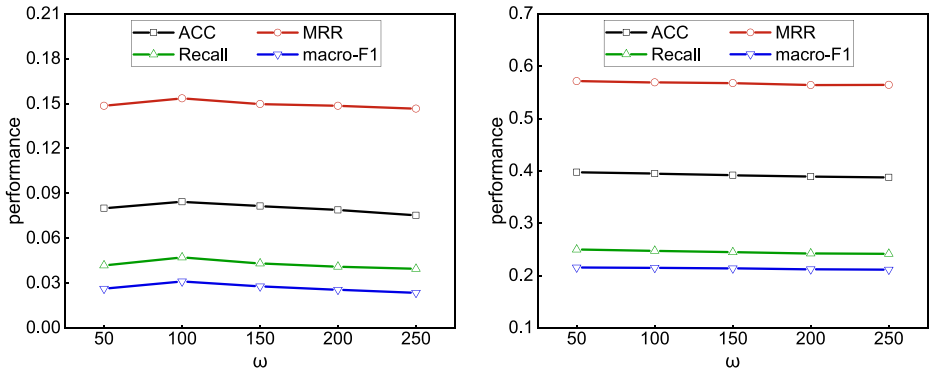
We compare GCMT with variants to study the usefulness of each component.

- **GCMT-E** removes the graph embedding, which uses a lookup layer to transform one-hot vectors of sequence into dense vector representations.
- **GCMT-T** removes the traffic encoder, which ignores the traffic-related contextual information.
- **GCMT-C** removes context attention mechanism, which sums trajectory sequence representation and traffic context representation.

As shown in Tables 2 and 3, all variants perform better than baselines, which confirms the advantage of our methods. In detail, GCMT-E performs worse than GCMT, indicating that graph-based embedding can effectively extract geographical and temporal cyclic effect. The comparison of GCMT-T and GCMT proves that traffic contexts are important for accuracy spatio-temporal joint prediction. Besides, GCMT outperforms GCMT-C, which shows that the context attention mechanism can fuse sequence information and traffic information according to trajectory's various background factors. GCMT successfully achieves the best result by utilizing complicated spatio-temporal contexts.

### 5.6 Effect of different grid granularity

Figure 4 shows the performance of GCMT with grid granularity $\omega$ from $\{50, 100, 150, 200, 250\}$ on next location prediction task. We can see that on Hangzhou dataset, when $\omega$ is less than 100, the performance increases as the number increases, because the direction reflected by future region is too large to provide useful information. Besides, when $\omega$ is greater than 100 on Hangzhou and 50 on Xiamen, increasing the number may result in a slight performance degradation, because the future region is too small to be accurately predicted and the predicted direction may have large deviations. Finally, the grid granularity is set as 100 on Hangzhou and 50 on Xiamen.




(a) Performance on Hangzhou (b) Performance on Xiamen

**Figure 4** Effects of different grid granularity

## 6 Conclusion

In this paper, we study the spatio-temporal joint prediction for cellular trajectories and propose a graph-contextualized multitask learning method which can learn the complicated spatio-temporal context information. Specifically, we introduce graph embedding module to utilize geographical influence and temporal cyclic effect to obtain meaningful representation for each location. To capture traffic-related contextual information, we combine temporal gated CNN and spatial graph convolution to learn the dynamic of traffic flows. In addition, a context attention mechanism is well-designed to fuse sequence information and traffic information according to trajectory's background factors. Finally, extensive experiment results on two real trajectory datasets have verified that GCMT can achieve accuracy spatio-temporal joint prediction.

**Acknowledgements** This work is supported by National Natural Science Foundation of China (No. 61872166), Six Talent Peaks Project of Jiangsu Province (2019 XYDXX-161).

**Author Contributions** Yu Sang and Yuan Xu wrote the main manuscript text. Bo Ning and Zhenping Xie participated in model design and technical discussion.

**Funding** The funding concludes the National Natural Science Foundation of China (No. 61872166), Six Talent Peaks Project of Jiangsu Province (2019 XYDXX-161).

**Availability of supporting data** Hangzhou and Xiamen datasets are non-public datasets.

### Declarations

**Competing interests** We declare that we have no conflict of interest.

## References

1. Blondel, V.D., Decuyper, A., Krings, G.: A survey of results on mobile phone datasets analysis. EPJ Data Sci. **4**(1), 10 (2015)

2. Chen, L., Shang, S., Feng, S., Kalnis, P.: Parallel subtrajectory alignment over massive-scale trajectory data. In: IJCAI, pp. 3613–3619 (2021)
3. Chen, L., Shang, S., Jensen, C.S., Yao, B., Zhang, Z., Shao, L.: Effective and efficient reuse of past travel behavior for route recommendation. In: KDD (2019)
4. Shang, S., Ding, R., Zheng, K., Jensen, C.S., Kalnis, P., Zhou, X.: Personalized trajectory matching in spatial networks. VLDB J. **23**(3), 449–468 (2014)
5. Feng, J., Li, Y., Zhang, C., Sun, F., Meng, F., Guo, A., Jin, D.: Deepmove: Predicting human mobility with attentional recurrent networks. In: WWW, pp. 1459–1468 (2018)
6. Kong, D., Wu, F.: HST-LSTM: A hierarchical spatial-temporal long-short term memory network for location prediction. In: IJCAI, pp. 2341–2347 (2018)
7. Liu, Q., Wu, S., Wang, L., Tan, T.: Predicting the next location: A recurrent model with spatial and temporal contexts. In: AAAI, pp. 194–200 (2016)
8. Yang, G., Cai, Y., Reddy, C.K.: Spatio-temporal check-in time prediction with recurrent neural network based survival analysis. In: IJCAI, pp. 2976–2983 (2018)
9. Aalen, O., Borgan, O., Gjessing, H.: Survival and event history analysis: a process point of view. Springer Science & Business Media (2008)
10. Du, N., Dai, H., Trivedi, R., Upadhyay, U., Gomez-rodriguez, M., Song, L.: Recurrent marked temporal point processes: Embedding event history to vector. In: SIGKDD, pp. 1555–1564 (2016)
11. Xiao, S., Yan, J., Yang, X., Zha, H., Chu, S.M.: Modeling the intensity function of point process via recurrent neural networks. In: AAAI, pp. 1597–1603 (2017)
12. Xu, Y., Xu, J., Fang, J., Liu, A., Zhao, L.: When multitask learning make a difference: Spatio-temporal joint prediction for cellular trajectories. In: DASFAA, ser, LNCS, vol. 13245, pp. 207–223 (2022)
13. Zheng, K., Shang, S., Yuan, N.J., Yang, Y.: Towards efficient search for activity trajectories. In: ICDE, pp. 230–241 (2013)
14. Shang, S., Chen, L., Jensen, C.S., Wen, J., Kalnis, P.: Searching trajectories by regions of interest. TKDE **29**(7), 1549–1562 (2017)
15. Shang, S., Chen, L., Wei, Z., Jensen, C.S., Zheng, K., Kalnis, P.: Parallel trajectory similarity joins in spatial networks. VLDB J. **27**(3), 395–420 (2018)
16. Shang, S., Chen, L., Zheng, K., Jensen, C.S., Wei, Z., Kalnis, P.: Parallel trajectory-to-location join. TKDE **31**(6), 1194–1207 (2019)
17. Yang, C., Chen, L., Wang, H., Shang, S.: Towards efficient selection of activity trajectories based on diversity and coverage. In: AAAI, pp. 689–696 (2021)
18. Han, P., Wang, J., Yao, D., Shang, S., Zhang, X.: A graph-based approach for trajectory similarity computation in spatial networks. In: KDD, pp. 556–564 (2021)
19. Sun, H., Xu, J., Zheng, K., Zhao, P., Chao, P., Zhou, X.: MFNP: A meta-optimized model for few-shot next POI recommendation. In: IJCAI, pp. 3017–3023 (2021)
20. Xu, J., Zhao, J., Zhou, R., Liu, C., Zhao, P., Zhao, L.: Predicting destinations by a deep learning based approach. TKDE **33**(2), 651–666 (2021)
21. Sun, H., Xu, J., Zhou, R., Chen, W., Zhao, L., Liu, C.: HOPE: A hybrid deep neural model for out-of-town next POI recommendation. WWW **24**(5), 1749–1768 (2021)
22. Xu, S., Zhang, R., Cheng, W., Xu, J.: Mtlm: a multi-task learning model for travel time estimation. GeoInformatica, no. 1 (2020)
23. Yang, G., Cai, Y., Reddy, C.K.: Recurrent spatio-temporal point process for check-in time prediction. In: CIKM, pp. 2203–2211 (2018)
24. Yang, D., Fankhauser, B., Rosso, P., Cudré-Mauroux, P.: Location prediction over sparse user mobility traces using rnns: Flashback in hidden states. In: IJCAI, pp. 2184–2190 (2020)
25. Zhang, Y., Yang, Q.: A survey on multi-task learning. CoRR, 1707.08114 (2017)
26. Fang, Y., Ma, Z., Zhang, Z., Zhang, X., Bai, X.: Dynamic multi-task learning with convolutional neural network. In: IJCAI, pp. 1668–1674 (2017)
27. Liang, W., Zhang, W.: Learning social relations and spatiotemporal trajectories for next check-in inference, TNNLS (2020)
28. Chen, Y., Long, C., Cong, G., Li, C.: Context-aware deep model for joint mobility and time prediction. In: WSDM, pp. 106–114 (2020)
29. Xue, A.Y., Zhang, R., Zheng, Y., Xie, X., Huang, J., Xu, Z.: Destination prediction by sub-trajectory synthesis and privacy protection against such prediction. In: ICDE, pp. 254–265 (2013)
30. Zhao, J., Xu, J., Zhou, R., Zhao, P., Liu, C., Zhu, F.: On prediction of user destination by sub-trajectory understanding: A deep learning based approach. In: CIKM, ACM, pp. 1413–1422 (2018)
31. Xie, M., Yin, H., Wang, H., Xu, F., Chen, W., Wang, S.: Learning graph-based POI embedding for location-based recommendation. In: CIKM, pp. 15–24 (2016)

32. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: NIPS, pp. 3111–3119 (2013)
33. Recht, B., Ré, C., Wright, S.J., Niu, F.: Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In: NIPS, pp. 693–701 (2011)
34. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: NIPS, pp. 5998–6008 (2017)
35. Li, Y., Li, K., Chen, C., Zhou, X., Zeng, Z., Li, K.: Modeling temporal patterns with dilated convolutions for time-series forecasting. ACM Trans. Knowl. Discov. Data **16**(3), 14:1-14:22 (2022)
36. Dauphin, Y.N., Fan, A., Auli, M., Grangier, D.: Language modeling with gated convolutional networks. In: ICML, vol. 70, pp. 933–941 (2017)
37. Kipf, T.N., Welling, M.: "Semi-supervised classification with graph convolutional networks. In: ICLR (2017)
38. Seiler, M.C., Seiler, F.A., et al.: Numerical recipes in c: the art of scientific computing. Risk Anal. **9**(3), 415–416 (1989)
39. Zuo, S., Jiang, H., Li, Z., Zhao, T., Zha, H.: Transformer hawkes process. In: ICML, vol. 119, pp. 11692–11702 (2020)