Predicting Human Mobility via Graph Convolutional Dual-attentive Networks

Weizhen Dang
Tsinghua University
Beijing, China
dangweizhen@126.com

Haibo Wang*
Tsinghua University
Beijing, China
hbwang1994@gmail.com

Shirui Pan
Monash University
Australia
shirui.pan@monash.edu

Pei Zhang
Beijing University of Posts and Telecommunications
Beijing, China
zhangpei@bupt.edu.cn

Chuan Zhou
Chinese Academy of Sciences
Beijing, China
zhouchuan@amss.ac.cn

Xin Chen
Tsinghua University
Beijing, China
chenxinsteven@outlook.com

Jilong Wang
Tsinghua University
Beijing, China
wjl@cernet.edu.cn

ABSTRACT

Human mobility prediction is of great importance for various applications such as smart transportation and personalized recommender systems. Although many traditional pattern-based methods and deep models (*e.g.*, recurrent neural networks) based methods have been developed for this task, they essentially do not well cope with the sparsity and inaccuracy of trajectory data and the complicated high-order nature of the sequential dependency, which are typical challenges in mobility prediction. To solve the problems, this paper proposes a novel framework named **G**raph **C**onvolutional **D**ual-**a**ttentive **N**etworks (GCDAN), which consists of two modules: *spatio-temporal embedding* and *trajectory encoder-decoder*. The first module employs a bidirectional diffusion graph convolution to preserve the spatial dependency in the location embedding. The second module employs a dual-attentive mechanism based on a *Sequence to Sequence* architecture to effectively extract the long-range sequential dependency within a trajectory and the correlation between different trajectories for predictions. Extensive experiments on three real-world datasets show that GCDAN achieves significant performance gain compared with state-of-the-art baselines.

CCS CONCEPTS

• **Computer systems organization** → **Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability.

KEYWORDS

Mobility Prediction, Graph Convolution, Attention Mechanism, Sequential Modeling

ACM Reference Format:
Weizhen Dang, Haibo Wang, Shirui Pan, Pei Zhang, Chuan Zhou, Xin Chen, and Jilong Wang. 2022. Predicting Human Mobility via Graph Convolutional Dual-attentive Networks. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining (WSDM '22), February 21–25, 2022, Tempe, AZ, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3488560.3498400

1 INTRODUCTION

With the popularization of smart mobile devices [6], location-based services become ubiquitous. Human mobility prediction is of great importance in a wide spectrum of location-based application scenarios, ranging from intelligent transportation planning and scheduling to personalized recommendation. Therefore, how to make full use of recorded historical trajectories to predict future location has attracted significant research efforts.

Traditional studies on mobility prediction are mostly pattern-based methods [24]. They adopt pattern discovery methods such as matrix factorization [3, 16, 17, 34] to extract typical mobility patterns existing in historical trajectories and then make predictions accordingly. However, these methods generally suffer from the one-sided nature of the popular patterns and fail to capture the fine-grained sequential transition regularity. Therefore, some works introduce Markov chain to construct the location transition matrix [4, 19, 23, 26, 37]. However, the location independence assumption of these methods prevents them from capturing the complicated sequential information.

In recent years, some works attempt to leverage the powerful sequential modeling capacity of deep models such as recurrent neural networks (RNNs) to capture the complicated location relationships in trajectories [8, 10, 18, 20, 33]. Despite their promising performance compared with traditional methods, they do not effectively address the following two key challenges of the mobility prediction

*Haibo Wang is the corresponding author.

WSDM '22, February 21–25, 2022, Tempe, AZ, USA

ACM ISBN 978-1-4503-9132-0/22/02…$15.00
https://doi.org/10.1145/3488560.3498400

problem in essence. **(1) Data sparsity and inaccuracy**. The trajectory data is sparse and may be unreliable in nature. In particular, The location information is recorded only when users access the location service, thus the recorded data is low-sampling in practice. Besides, the coarse-grained locating method may not be able to distinguish locations close in physical space, which means that the recorded data is not reliable. **(2) High-order trajectories**. The sequential dependency in trajectories is usually high-order. This is due to the fact that the location to be predicted may exhibit close dependency with a distant location instead of an adjacent location. Also, the future locations depend on the user preference and habits. Unfortunately, existing deep models for mobility prediction neglect the high-order nature.

To solve the problems, in this paper, we propose a novel framework termed as **G**raph **C**onvolutional **D**ual-**a**ttentive **N**etworks (GCDAN for short). GCDAN consists of two modules: *spatio-temporal embedding* and *trajectory encoder-decoder*. In the first module, we construct a location dependency graph based on historical trajectories and design the bidirectional diffusion graph convolution to capture the spatial dependency of locations in trajectories, which could mitigate the effect of the sparsity and inaccuracy of the trajectory data by neighbor location information aggregation. In the second module, we employ a *Sequence to Sequence* architecture to predict the human mobility by extracting valuable sequential patterns of historical trajectories. Instead of using the RNN-based methods, we design a dual-attentive mechanism, which includes the intra-trajectory attention and inter-trajectory attention, to handle the high-order nature of the trajectory data. The intra-trajectory attention has the capacity to model the long-range sequential dependency within a trajectory and the inter-trajectory attention further captures the user preference by modeling the correlation between different trajectories. Extensive experiments on different real-world datasets show that GCDAN achieves significant performance gain compared with the state-of-the-art baselines.

The main contributions can be summarized as follows:

- To the best of our knowledge, we are among the first to introduce the graph convolution and dual-attentive mechanism to handle the sparsity and inaccuracy of the trajectory data and the high-order sequential nature in the problem of human mobility prediction.
- We propose a novel framework GCDAN for mobility prediction, which consists of two modules. The first module learns dense representations for locations in trajectories by simultaneously preserving the spatial dependency and temporal characteristics. On this basis, the second module further employs a *Sequence to Sequence* architecture, which fully considers the complicated sequential dependence and user preference, to predict the next location under the guidance of historical trajectories.
- We conduct extensive experiments on three real-world datasets (including two commonly used datasets and one newly collected datasets) to evaluate the effectiveness of the proposed framework. The results show that GCDAN consistently outperforms state-of-the-art baselines. Besides, we publicly publish the collected dataset as a new benchmark for the mobility prediction problem, which records user mobility trajectories in one of the largest campus wireless networks.

2 PROBLEM FORMULATION

In this section, we formally formulate the problem of user mobility prediction. Before that, we first define *spatio-temporal point* and *trajectory* as follows:

DEFINITION 1 (Spatio-temporal Point) Let $L = \{l_1, ..., l_{|L|}\}$ denote the set of location identifiers. A spatio-temporal point p is defined as a tuple of location identifier $l \in L$ and timestamp t, *i.e.*, $p = (l, t)$, which represents that a user arrived at location l at time t.

DEFINITION 2 (Trajectory) Let $U = \{u_1, ..., u_{|U|}\}$ denote the set of users. A trajectory for user $u \in U$ is defined as a time-ordered spatio-temporal point sequence $T_u = p_u^1 p_u^2 ... p_u^m$, where p_u^i represents the i^{th} spatio-temporal point in T_u. The length of different trajectories, *i.e.*, m, is not necessarily the same for each user.

For a user $u \in U$, let $S_u = \{T_u^1, T_u^2, ..., T_u^{|S_u|}\}$ denote the set of its historical trajectories, and let $\widetilde{T_u} = p_u^1 p_u^2 ... p_u^n$ denote its current trajectory. On this basis, we could formally define the the problem of user mobility prediction as follows:

PROBLEM (Mobility Prediction) For an arbitrary user u, given the set of its historical trajectories S_u and its current trajectory $\widetilde{T_u}$, the goal is to predict the next spatio-temporal point in the current trajectory, *i.e.*, p_u^{n+1}.

Following the usual practice, we quantify the time interval of adjacent spatio-temporal points in a trajectory into a fixed value, which simplifies the problem to predict the next location identification l in the next time interval.

3 OUR SOLUTION

To solve the problem of user mobility prediction, we propose a novel framework, namely **G**raph **C**onvolutional **D**ual-**a**ttentive **N**etworks (GCDAN for short), whose overview is shown in Fig. 1. GCDAN consists of two modules: (1) *spatio-temporal embedding* and (2) *trajectory encoder-decoder*. The first module aims to embed spatio-temporal points in trajectories into dense representations, which simultaneously captures their spatial dependency and temporal characteristics. The second module employs a *Sequence to Sequence* architecture, which considers the sequential dependence within a trajectory and the correlation between different trajectories. In the following parts, we will introduce the design of GCDAN in detail.

3.1 Spatio-temporal Embedding Module

For predicting the next spatio-temporal point of a given trajectory, the first step is to embed all spatio-temporal points in trajectories into dense representations. Note that the human mobility patterns are governed by multiple factors such as the location property and time, therefore, we expect that the learned dense representations should preserve the spatio-temporal semantic information for modeling the complicated user transition states. To achieve the goal, a natural idea is to separately learn the representations of locations and timestamps, and then concatenate them to generate the final representations.

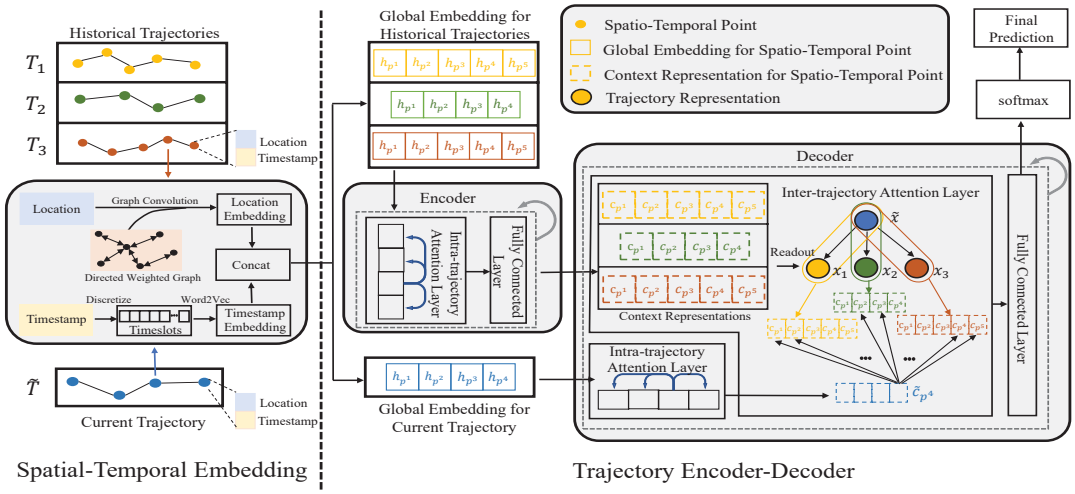

Figure 1: The overview of GCDAN. The spatial-temporal embedding module introduces the graph convolution and sparse embedding to learn representations of spatial-temporal points. The trajectory encoder-decoder module employs a dual-attentive mechanism. The intra-trajectory attention layer models the sequential dependence within a trajectory and generates the context representation for each spatial-temporal point. The inter-trajectory attention layer models the correlation between the historical trajectories and current trajectory and generates the final prediction.

Timestamp Embedding: Due to the continuous property of the real-value timestamp, it is impracticable to directly learn its representation. Inspired by previous works [8], we align all timestamps into a fixed time interval. The time interval can be set to one day or one week, which could generally reflect the mobility periodicity. Then we discretize the fixed time interval into ρ timeslots. For each timestamp t, we further map it into a corresponding timeslot and express it as a one-hot vector $\boldsymbol{v}_t$ with dimension ρ. Inspired by the word2vec [21] project, we convert $\boldsymbol{v}_t$ into a low-dimensional dense representation $\boldsymbol{h}_t$ using a transformation matrix $\Theta_T \in \mathbb{R}^{d_1 \times \rho}$, where d_1 is the dimension of $\boldsymbol{h}_t$, *i.e.*, $\boldsymbol{h}_t = \Theta_T \boldsymbol{v}_t$. The low-dimensional embedding can not only capture the precise temporal semantic information, but also avoid the curse of dimensionality and improve the follow-up computation.

Location Embedding: In general, the timestamp can be regarded as the accurate information since it is often automatically recorded by systems such as mobile devices. However, the location information in trajectories is much less accurate. It is because that the location information is generally recorded when users access the location service, which leads to the sparsity of the trajectory data. Furthermore, the locating method is often coarse-grained, some locations close in physical space could be difficult to distinguish accurately. To solve the problem, we introduce *graph convolution* to capture the spatial dependency and mitigate the inaccuracy of locations.

Let $G = (L, E, \boldsymbol{W})$ denote the directed weighted graph depicting the location dependence in physical space, where L represents the location set defined in Section 2, $E \subseteq L \times L$ represents the set of directed edges and $\boldsymbol{W} \in \mathbb{R}^{|L| \times |L|}$ represents the weighted adjacency matrix. The construction of G is based on the historical trajectory data. For each spatio-temporal point pair (l_i, l_j), we count the total number of times (denoted as c_{ij}) it appears in historical trajectories. We define:

$$\begin{cases} w_{ij} = \frac{c_{ij}}{\sum\limits_{l_r \in L} c_{ir}} & \frac{c_{ij}}{\sum\limits_{l_r \in L} c_{ir}} > \frac{1}{|L|}, \\ w_{ij} = 0 & otherwise, \end{cases} \tag{1}$$

where w_{ij} represents the weight of $\boldsymbol{W}$. $w_{ij} > 0$ means that there exists a directed edge between l_i and l_j (*i.e.*, $e_{ij} = 1$). Let $D_O = diag(\boldsymbol{W}\boldsymbol{1})$ and $D_I = diag(\boldsymbol{W}^{\mathrm{T}}\boldsymbol{1})$ denote the out-degree diagonal matrix and in-degree diagonal matrix respectively, where $\boldsymbol{1} \in \mathbb{R}^{|L|}$ represents the all one vector. Let $\boldsymbol{V}_L$ denote the one-hot representations for all locations. We could further define the graph convolution in the form of the bidirectional diffusion process, which is expressed as follows:

$$\boldsymbol{H}_L = \sum_{k=0}^{K-1} \sigma\Big((D_O^{-1}\boldsymbol{W})^k \boldsymbol{V}_L \Theta_k^0 + (D_I^{-1}\boldsymbol{W}^{\mathrm{T}})^k \boldsymbol{V}_L \Theta_k^1\Big), \tag{2}$$

where $\boldsymbol{H}_L = (\boldsymbol{h}_{l_1}, ..., \boldsymbol{h}_{l_{|L|}})^{\mathrm{T}}$ is the dense representations for all locations, K is the total diffusion steps, $\{\Theta_k^0, \Theta_k^1\}$ are the filters to be learnt in the k^{th} step, and σ is a non-linear activation function. The bidirectional diffusion graph convolution empowers the location representations more flexibility to capture the spatial dependency for locations in trajectories. In addition to the own information, the learned representations also contain information of nearby locations, which could reduce small location errors mentioned above during the mobility prediction. In this problem, we embed all locations into the d_2-dimensional space, *i.e.*, $\boldsymbol{H}_L \in \mathbb{R}^{|L| \times d2}$, $\Theta_k^0 \in \mathbb{R}^{|L| \times d2}$ and $\Theta_k^1 \in \mathbb{R}^{|L| \times d2}$.

Global Embedding: For a spatio-temporal point $p = (l, t)$, the global embedding $\boldsymbol{h}_p$ is obtained by concatenating its timestamp embedding and location embedding, *i.e.*, $\boldsymbol{h}_p = \boldsymbol{h}_t \parallel \boldsymbol{h}_l$, where $\parallel$ is the concatenation operation and $\boldsymbol{h}_p \in \mathbb{R}^{d1+d2}$.

Trajectory Position Coding: The trajectory is sensitive to the location order. For two trajectories with the same location

set but different location order, the embedding results should be distinguishable. Inspired by the work [27], we introduce the position coding mechanism, which could add information about position in a sequence into the global embedding. Let $T = p^1 p^2 ... p^m$ denote a trajectory. The global embedding of m spatio-temporal points in T is expressed as $\boldsymbol{H}_T = (\boldsymbol{h}_{p^1}, ..., \boldsymbol{h}_{p^m})^{\mathrm{T}}$, where $\boldsymbol{h}_{p^i}$ represents the embedding of the i^{th} spatio-temporal point in T. To further introduce the information of position i into $\boldsymbol{h}_{p^i}$, we construct a new position vector $\boldsymbol{pos}_i$ with dimension $(d1 + d2)$ as follows:

$$\boldsymbol{pos}_i(j) = \cos\left(\frac{i}{C^{\frac{j-(j\%2)}{d_1+d_2}}} - \frac{(j\%2)\pi}{2}\right); \tag{3}$$

where C is a constant that decides the function frequency for position coding. The final embedding for p^i in T is expressed as $\boldsymbol{h}_{p^i} = \boldsymbol{h}_{p^i} + \boldsymbol{pos}_i$, which serves as the input for the second module of GCDAN.

3.2 Trajectory Encoder-decoder Module

Based on Section 3.1, we could obtain the global embedding of spatio-temporal points for a given trajectory. Let $S = \{T^1, T^2, ..., T^{|S|}\}$ and $\widetilde{T} = p^1 p^2 ... p^n$ denote the historical trajectory set and the current trajectory of a user respectively. Their spatio-temporal point representations can be correspondingly expressed as $S_H = \{\boldsymbol{H}_{T^1}, \boldsymbol{H}_{T^2}, ..., \boldsymbol{H}_{T^{|S|}}\}$ and $\boldsymbol{H}_{\widetilde{T}}$. To predict the next spatio-temporal point in $\widetilde{T}$, the trajectory encoder-decoder module employs a *Sequence to Sequence* architecture, which is implemented based on a *dual-attentive mechanism* (including *intra-trajectory attention* and *inter-trajectory attention*). The encoder effectively extracts the sequential information contained in historical trajectories using the intra-trajectory attention as the guidance information for prediction. Specifically, it takes S_H as input and generates the context representations for spatio-temporal points in each trajectory. The decoder models the correlation between the current trajectory and historical trajectories using inter-trajectory attention. It takes $\boldsymbol{H}_{\widetilde{T}}$ and the context representations obtained by the encoder as input and generates the predicted location distribution vector. The detailed design of the encoder and decoder follows.

Encoder: The sequential transition in a trajectory usually exhibited with high-order nature, which means that the location may not depend on the adjacent location, but on a distant location in the trajectory sequence. Therefore, we design the intra-trajectory attention layer in the encoder, which could better model the long-distance dependence compared with the recurrent units such as RNN and LSTM.

Given a history trajectory $T = p^1 p^2 ... p^m \in S$ and the representations of its spatio-temporal points $\boldsymbol{H}_T = (\boldsymbol{h}_{p^1}, ..., \boldsymbol{h}_{p^m})^{\mathrm{T}}$, for each spatio-temporal point p^i, the intra-trajectory attention layer computes similarity between its representation $\boldsymbol{h}_{p^i}$ and the representations of all candidate spatio-temporal points in T and further generates its context representation $\boldsymbol{c}_{p^i}$ by the sum of candidate representations weighted by the similarity, which can be formulated as follows:

$$sim_p(\boldsymbol{h}_{p^i}, \boldsymbol{h}_{p^j}) = \frac{\exp\left(f(\boldsymbol{h}_{p^i}, \boldsymbol{h}_{p^j})\right)}{\sum_{r=1}^{m} \exp\left(f(\boldsymbol{h}_{p^i}, \boldsymbol{h}_{p^r})\right)}, \tag{4}$$

$$\boldsymbol{c}_{p^i} = \sum_{j=1}^{m} sim_p(\boldsymbol{h}_{p^i}, \boldsymbol{h}_{p^j}) \boldsymbol{h}_{p^j}, \tag{5}$$

where m is the length of the trajectory T, $\boldsymbol{h}_{p^j} \in \mathbb{R}^{d1+d2}$ is the j^{th} representations of $\boldsymbol{H}_T$ and $f(.)$ is the score function such as the dot product function or the bilinear function.

To obtain sufficient expressive power, the intra-trajectory attention layer is followed by a fully connected layer and the two layers form a basic unit of the encoder. In practice, the encoder is stacked by multiple basic units.

Decoder: Given the current trajectory $\widetilde{T} = p^1 p^2 ... p^n$ and the representations of its spatio-temporal points $\boldsymbol{H}_{\widetilde{T}} = (\boldsymbol{h}_{p^1}, ..., \boldsymbol{h}_{p^n})^{\mathrm{T}}$, similar to the encoder, we first employ a intra-trajectory attention layer to obtain the context vector of the spatio-temporal point p^n in $\widetilde{T}$, which is denoted as $\widetilde{\boldsymbol{c}}_{p^n}$. For predicting the next location, we expect to make full use of the guidance information of historical trajectories. To achieve the goal, we design the inter-trajectory attention layer, which computes the similarity between $\widetilde{\boldsymbol{c}}_{p^n}$ and the context representations of spatio-temporal points in historical trajectories obtained in the encoder and generates the weighted aggregated representation.

However, different historical trajectories for a user could show diverse spatio-temporal behavior patterns. They may have different effects when performing inter-trajectory attention on $\widetilde{\boldsymbol{c}}_{p^n}$, which is usually ignored by previous works [8, 20] and could reflect the user preference to some extent. Therefore, we further modify the inter-trajectory attention layer to capture the correlation between the current trajectory and different historical trajectories and pay more attention on trajectories that are more related to the current trajectory. Note that the length of different trajectories may be different. We employ a *readout* function to obtain the vector representation of a trajectory by aggregating the context representations of its spatio-temporal points. Let $\boldsymbol{x}$ be the representation for a trajectory of a user, which can be expressed as follows:

$$\boldsymbol{x} = \frac{1}{m} \sum_{j=1}^{m} \boldsymbol{c}_{p^j}, \tag{6}$$

where $\boldsymbol{c}_{p^j}$ is the context representation of the j^{th} spatio-temporal point in the trajectory. On this basic, the inter-trajectory attention layer can be formulated as follows:

$$sim_t(\widetilde{\boldsymbol{x}}, \boldsymbol{x}_i) = \frac{\exp\left(f(\widetilde{\boldsymbol{x}}, \boldsymbol{x}_i)\right)}{\sum_{r=1}^{|S|} \exp\left(f(\widetilde{\boldsymbol{x}}, \boldsymbol{x}_r)\right)}, \tag{7}$$

$$\boldsymbol{v}_{p^n} = \sum_{i=1}^{|S|} sim_t(\widetilde{\boldsymbol{x}}, \boldsymbol{x}_i) \sum_{j=1}^{m} sim_p(\widetilde{\boldsymbol{c}}_{p^n}, \boldsymbol{c}_{p^j}) \boldsymbol{c}_{p^j}, \tag{8}$$

where S is the historical trajectory set, $\boldsymbol{x}_i$ is the representation of the i^{th} historical trajectory, $\widetilde{\boldsymbol{x}}$ is the representation of the current trajectory and $\boldsymbol{v}_{p^n}$ represents the final output vector for the current trajectory. In addition, we adopt the *multi-head attention mechanism* [27, 28] to stabilize the learning process and capture the dependency from more aspects. It first splits the context vectors into multiple groups and respectively calculates the output vector for each group using Eq (8). After that, the output vectors of all groups are concatenated to generate the final output.

Similar to the architecture of the encoder, the inter-trajectory attention layer is followed by a fully connected layer. The intra-trajectory attention layer, the inter-trajectory attention layer and the fully connected layer form a basic unit of the decoder. In practice, the decoder is a stack of multiple basic units. In this case, the inter-trajectory attention in each unit is not only performed on the context vector of p^n, but also on context vectors of other spatio-temporal points in $\widetilde{T}$ and the output results will serve as the input of the next unit.

Objective Function: The final output $\boldsymbol{v}_{p^n}$ represents the context information of the next location to be predicted. In general, the problem of the mobility prediction can be regarded as a multi-classification problem. Therefore, we further transform $\boldsymbol{v}_{p^n}$ into the probability distribution of all possible locations in L, which is expressed as follows:

$$\hat{\boldsymbol{y}} = softmax(\hat{\boldsymbol{\Theta}}\boldsymbol{v}_{p^n}), \tag{9}$$

where $\hat{\boldsymbol{\Theta}} \in \mathbb{R}^{|L|\times(d1+d2)}$ is a trainable weight matrix. To train the model, we could minimize the cross-entropy loss $\mathcal{L}_c$ with the following form:

$$\mathcal{L}_c = -\sum_{u\in U}\sum_{i=1}^{|L|} \boldsymbol{y}_u^i \log \hat{\boldsymbol{y}}_{\boldsymbol{u}}^{\boldsymbol{i}}, \tag{10}$$

where U is the user set, $(\boldsymbol{y}_u^1, \boldsymbol{y}_u^2, \ldots, \boldsymbol{y}_u^{|L|})$ denotes the one-hot ground-truth distribution of the next location in the current trajectory for user u and $(\hat{\boldsymbol{y}}_u^1, \hat{\boldsymbol{y}}_u^2, \ldots, \hat{\boldsymbol{y}}_u^{|L|})$ denotes the predicted distribution for user u obtained by Eq. (9). To avoid overfitting, we also impose L_2 regularization on all parameters, which produces the final objective function as follows:

$$\mathcal{L} = \mathcal{L}_c + \alpha\mathcal{L}_{reg}, \tag{11}$$

where $\mathcal{L}_{reg}$ represents the sum of the L2-norm for all parameters and α is a balance hyperparameter.

3.3 Model Training and Optimization

The proposed framework GCDAN can be effectively trained in an end-to-end manner. The pseudo-code is presented in Algorithm 1. In the training phase, for each user, we split its trajectories (only training data is involved) into two sets with the same size. One set serves as the historical trajectories and the other set serves as the current trajectories to be predicted. We adopt the *stochastic gradient descent* to optimize the whole model.

In the training phase, we apply three training skills, namely *trajectory division*, *trajectory padding* and *trajectory reuse*. We will respectively introduce them as follows:

Trajectory Division: As mentioned in Section 2, we divide the spatio-temporal point sequence of a user into multiple trajectories based on the time interval of two adjacent spatio-temporal points, which could bring two benefits. (1) The division method allows the inter-trajectory attention mechanism proposed in Section 3.2 to capture the contribution of different subsequences to mobility prediction. (2) Let N denote the number of divided trajectories. Let $\bar{x}$ denote the average length, which is generally a small value. The division method reduces the number of calculations of intra-trajectory attention in the encoder from $O(N^2\bar{x}^2)$ to $O(N\bar{x}^2)$, which greatly improves the computing efficiency.

Procedure 1 GCDAN

Input: User set U, trajectory set A_u for each user $u \in U$.
Output: All parameters of the trained model.

```
//prepare the training data
Initialize the training data set D = ∅;
for u ∈ U do
    Split A_u into two sets with the same size, namely a historical trajectory set S_u and a current trajectory set C_u;
    Add (S_u, C_u) into D;
end for
Construct the directed weighted graph G based on the historical trajectory sets for all users using Eq. (1);
//model training
Initialize all parameters for GCDAN;
while L does not converge do
    for (S_u, C_u) ∈ D do
        Calculate h_p for each spatio-temporal point of all trajectories in S_u and C_u;
        Calculate c_p for each spatio-temporal point of historical trajectories in S_u and c̃_{p^n} for the last spatio-temporal point of each current trajectory in C_u using Eq. (5);
        Calculate x for each historical trajectory in S_u and x̃ for each current trajectory in C_u using Eq. (6);
        Calculate v_{p^n} for each current trajectory using Eq. (8);
        Calculate the objective function using Eq. (11);
        Update all parameters by stochastic gradient descent;
    end for
end while
```

Trajectory Padding: The length of different divided trajectories is not necessarily the same, which makes learning the context representations of spatio-temporal points in all historical trajectories inefficient. To solve the problem, we design the skill of trajectory padding, which could make the length of all trajectories consistent with the longest trajectory by adding *virtual spatio-temporal points*, so that all context representations could be obtained through efficient matrix operations. To eliminate their effect when performing the attention operation, we introduce a mask vector $\boldsymbol{M}$, which indicates the positions of the virtual spatio-temporal points. Specifically, assume that the length of the longest trajectory is $\hat{m}$. For a trajectory with the length m, we need to add $(\hat{m} - m)$ virtual spatio-temporal points, which means that the first m dimensions of its mask vector are 1 and the last $(\hat{m} - m)$ dimensions are 0. Then we could rewrite Eq. (4) as follows:

$$sim_p(\boldsymbol{h}_{p^i}, \boldsymbol{h}_{p^j}) = \frac{M(j)\exp\left(f(\boldsymbol{h}_{p^i}, \boldsymbol{h}_{p^j})\right)}{\sum_{r=1}^{m} M(r)\exp\left(f(\boldsymbol{h}_{p^i}, \boldsymbol{h}_{p^r})\right)}, \tag{12}$$

where $M(j)$ represents the value of the j^{th} dimension of M. By this way, the virtual spatio-temporal points will make no contribution when calculating the attention value.

Trajectory Reuse: To further strengthen the training process, we could reuse current trajectories to be predicted for more supervised information. For a given current trajectory $\widetilde{T} = p^1p^2...p^n$, instead of only predicting the spatio-temporal point p^{n+1}, we expect to generate another $n-1$ trajectories $\{\widetilde{T}^i = p^1p^2...p^i\}_{i=1,..,n-1}$,

and predict p^{i+1} for each $\widetilde{T}^i$. Note that although the full trajectory sequence information is known, when predicting p^{i+1} for $\widetilde{T}^i$, we should not use the future information, *i.e.*, spatio-temporal points after p^{i+1}. Therefore, we adopt the trajectory mask skill again to ensure that each spatio-temporal point in $\widetilde{T}$ only has the knowledge of the spatio-temporal point sequence before it when performing intra-trajectory attention in the decoder.

3.4 Complexity Analysis

In this section, we analyze the complexity of GCDAN. Our framework can be divided into 2 components in series, namely spatio-temporal embedding module and trajectory encoder-decoder module. The spatio-temporal embedding module mainly consists of a timestamp embedding function and a location embedding function. The computational complexity of the timestamp embedding function is $O(d_1\rho)$, where ρ represents the number of timeslots and d_1 is the dimension of the low-dimensional dense representation after the embedding operation. The location embedding function is a graph convolution function, whose computational complexity is $O(|E|)$, where $|E|$ represents the number of edges in the constructed graph. As for the trajectory encoder-decoder module, the computational complexity of the attention function is $O\left(\hat{m}^2(d_1+d_2)\right)$, where $\hat{m}$ represents the trajectory length after trajectory padding. Given the number of trajectories for user u (*i.e.*, $|S_u|$), the overall computational complexity is $O\left(|S_u|\hat{m}^2(d_1+d_2)\right)$.

4 EVALUATION

In this section, we empirically evaluate the performance of the proposed GCDAN framework.

4.1 Experiment Setup

4.1.1 Datasets. In this paper, we conduct comprehensive experiments on three real-world datasets: (1) Gowalla [5], (2) Foursquare [32] and (3) WiFi-Trace. The first two are commonly used datasets in previous works [8, 20]. They record the user check-in information, which consists of user ID, timestamp and GPS location. Inspired by previous work[29], we collect the third dataset records wireless association information in the large-scale wireless network of T universit. For protecting the user privacy, all sensitive information (*e.g.*, device MAC address) is anonymized. User identifiable information cannot be traced back. For better reproducibility of experimental results in this paper, after obtaining the consent of network administrators in T university, we publicly publish the WiFi-Trace dataset as a new benchmark for the mobility modeling problem [1].

For each dataset, we first extract a sequence of spatio-temporal points for each user. Then we divide the sequence into different trajectories if the time interval between two adjacent spatio-temporal points exceeds a predefined threshold as mentioned in Section 2. The threshold settings vary in different datasets. For Gowalla and Foursquare, the threshold is set to 24 hours, while for WiFi-Trace, the threshold is set to 1 hour. Table 1 shows the detailed statistics for each dataset. In our experiments, we select the first 75% of trajectories for each user as the training data and the rest 25% as the testing data. In addition, 10% of the training data is used as the validation data.

[1]The published WiFi-Trace dataset and our reproducible code are available at https://github.com/GCDAN/GCDAN.

Table 1: Dataset statistics.

Dataset	Gowalla	Foursquare	WiFi-Trace
Time duration	1 year	1 year	2 week
Users	857	886	3642
#(Record)	129679	82575	1011049
#(Trajectory)	8568	7974	41937
#(Location)	20006	10497	251

4.1.2 Baselines. To demonstrate the superiority of GCDAN, we compare it with five state-of-the-art baselines.

MC [9]: Markov Chain is a traditional but widely used approach for sequential prediction. It regards locations as states and builds a transition matrix between these states to make prediction.

FPMC [23]: Factorize Personal Markov Chain combines Markov chain with matrix factorization to capture both sequential patterns and user preferences for mobility prediction.

LSTM [25]: Long Short-Term Memory network is a recurrent neural network. It is designed to capture the long-term sequential influence, and it can be applied to mobility prediction.

ST-RNN [18]: Spatial Temporal Recurrent Neural Network extends RNN to model the local temporal and spatial contexts in trajectories for mobility prediction.

DeepMove [8]: DeepMove designs a multi-modal embedding recurrent network together with a historical attention model to capture both spatial and temporal dependency.

4.1.3 Performance Metrics and Parameter Settings. In our experiments, we adopt the top-k accuracy (Acc@k) to evaluate the performance of GCDAN and baselines. Acc@k represents the proportion of the ground truth that lies in the top-k highest possible locations in our prediction, which can be formulated as follows:

$$Acc@k = \frac{|\{s|s \in P, l(s) \in Top\_k(s)\}|}{|P|} \tag{13}$$

where $|P|$ is the number of all predictions, $l(s)$ is the ground truth of a prediction s, and $Top\_k(s)$ represents the top-k most likely locations in prediction s. In this paper, we adopt Acc@1, Acc@5 and Acc@10 to evaluate the performance. In our experiments, we train each model for 20 times and report the average results for the three performance metrics.

For all baselines, we adopt the optimal parameter settings reported in corresponding papers. For our proposed GCDAN framework, all hyperparameters are tuned on the validation set to achieve the optimal performance. For the spatio-temporal embedding module, the dimension d_1 for timestamp embedding and the dimension d_2 for location embedding are set to 32 and 512 respectively. The total diffusion steps K is set to 2. The activation function in Eq. (2) is ReLU. For the trajectory encoder-decoder module, the number of stacking units of the encoder and decoder is set to 3. The number of heads of the multi-head mechanism is set to 4. We set the balance hyperparameter α in Eq. (11) to 1e-5 and adopt Adam [14] with the learning rate of 5e-4 to optimize the model until converging.

Table 2: Performance comparison between GCDAN and state-of-the-art baselines on Gowalla.

Metrics	MC	FPMC	LSTM	ST-RNN	DeepMove	GCDAN
Acc@1	0.1188	0.1204	0.1252	0.1279	0.1338	**0.1377**
Acc@5	0.2056	0.2066	0.2453	0.2568	0.2731	**0.3086**
Acc@10	0.2306	0.2307	0.2907	0.3112	0.3299	**0.3780**

Table 3: Performance comparison between GCDAN and state-of-the-art baselines on Foursquare.

Metrics	MC	FPMC	LSTM	ST-RNN	DeepMove	GCDAN
Acc@1	0.0818	0.0835	0.0912	0.1057	0.1245	**0.1613**
Acc@5	0.1753	0.1762	0.1824	0.2368	0.2790	**0.3417**
Acc@10	0.2252	0.2257	0.2120	0.2787	0.3360	**0.4093**

Table 4: Performance comparison between GCDAN and state-of-the-art baselines on WiFi-Trace.

Metrics	MC	FPMC	LSTM	ST-RNN	DeepMove	GCDAN
Acc@1	0.1612	0.3231	0.5685	0.5702	0.5729	**0.5912**
Acc@5	0.3173	0.5169	0.7821	0.7918	0.8030	**0.8064**
Acc@10	0.3882	0.6843	0.8473	0.8523	0.8707	**0.8726**

4.2 Performance Analysis

In this section, we compare the performance of GCDAN and baselines on three real-world datasets. The experimental results are respectively shown in Table 2, Table 3 and Table 4. We highlight the best performance in **bold**. We can see that traditional methods (*i.e.*, MC and FPMC) perform poorly on all three datasets. While in contrast, deep models (*i.e.*, LSTM, ST-RNN and DeepMove) achieve better performance, as they benefit from their more powerful sequential modeling capabilities. In particular, DeepMove introduces delicately designed attention mechanisms to capture complex sequential regularity, which further improves the performance. However, the trajectory data is generally sparse and may be unreliable, which would inevitably affect the accuracy of sequential modeling for these deep models. Meanwhile, these deep models also cannot well capture the user preference and habits, which limits their performance.

Compared with baselines, we observe that GCDAN achieves higher performance on all datasets in terms of all metrics. It is worth mentioning that on Foursquare, compared to DeepMove (the best baseline), GCDAN yields nearly 30% improvement in terms of Acc@1 and more than 20% improvement in terms of Acc@5 and Acc@10. The significant improvement mainly benefits from two aspects. First, the graph convolution mitigates the effect of the sparsity and inaccuracy of the trajectory data. Second, the dual-attentive mechanism empowers GCDAN stronger capacity to model the high-order sequential nature. Besides, we observe that the performance improvement of Acc@5 and Acc@10 on WiFi-Trace is not as significant as that on Foursquare and Gowalla. It is because that the sequential regularity and mobility patterns for the trajectory data on campus is relatively easier to capture than user check-in data. In addition, the data collection method for WiFi-Trace determines that the trajectory data is more complete and is with less noise. This is helpful for the learning of all deep models.

Table 5: Ablation Study.

	Gowalla	Foursquare	WiFi-Trace
GCDAN-base	0.1311	0.1580	0.5768
GCDAN-tc	0.1340	0.1607	0.5815
GCDAN-gc	0.1336	0.1597	0.5895
GCDAN-full	0.1377	0.1613	0.5912

The above experimental results demonstrate the effectiveness and superiority of GCDAN.

4.3 Parameter Sensitivity Analysis

In this section, we conduct experiments on all three datasets to study the effect of 4 main hyperparameters on the performance (Acc@1) of GCDAN.

Location Embedding Dimension: Fig. 2(a) presents the sensitivity analysis $w.r.t.$ the location embedding dimension d_2. We observe that the performance degrades significantly when the embedding dimension is lower than 256. As the dimension increases, the performance becomes stable. It is because that each head needs a certain dimension of representation to perform attention operation when adopting the multi-head attention mechanism. In our experiments, to obtain sufficient representation capacity and avoid the risk of overfitting, we set the location embedding dimension d_2 to 512.

Number of Basic Units in Encoder and Decoder: Fig. 2(b) presents the effect of the number of basic units in encoder and decoder on performance. We observe that the performance degrades when the number is very small (<3) and stays stable when the number grows. Considering that the convergence rate will decrease as the number grows, we set the hyperparameter to 3 in our experiments.

Number of Heads: Fig. 2(c) presents the sensitivity analysis $w.r.t.$ the number of heads in the multi-head attention mechanism. We can see that the performance could slightly reduce when the number of heads is too large. Therefore, in our experiments, to capture the sequential dependency from more aspects while avoiding the performance limitation of the excessive number of heads, we set the head number to 4.

Balance Hyperparameter α: α is the hyperparameter that balances the objective function $\mathcal{L}_c$ and the L_2 regularization term $\mathcal{L}_{reg}$ in Eq. (11). Fig. 2(d) shows the trend of Acc@1 when α takes values under different orders of magnitude. We can see that the performance reduces when α is large. In our experiments, we set α to 10^{-5} to achieve desirable performance.

4.4 Ablation Study

In GCDAN, we design the *graph convolution* in Eq. (2) to handle the sparse and inaccurate trajectory data and introduce the *trajectory correlation* into the inter-trajectory attention layer in Eq. (8) to capture the complex user perference. In this section, we mainly conduct ablation study to demonstrate their effectiveness. First, we detach both of them from GCDAN, which produces a basic model (denoted as GCDAN-base) that only considers the intra-trajectory attention and simplified inter-trajectory attention [2]. Then we construct three

[2]It is constructed without considering the trajectory correlation.

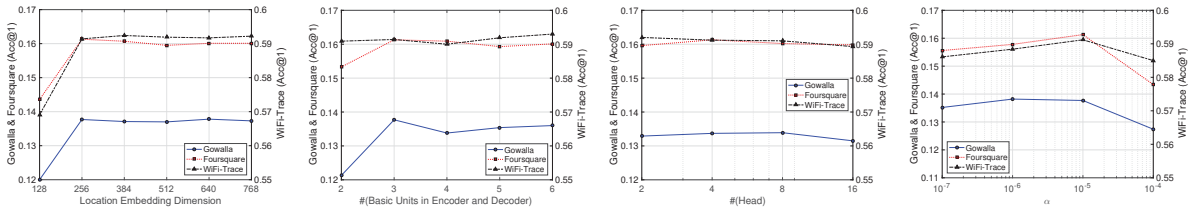

Figure 2: Results of parameter sensitivity $w.r.t.$ (a) the location embedding dimension, (b) the number of basic units in encoder and decoder, (c) the number of heads of the multi-head attention mechanism and (d) the balance hyper-parameter α in Eq. (11)

variants on the basis of GCDAN-base. The first one solely introduces graph convolution (denoted as GCDAN-gc), the second one solely introduces trajectory correlation (denoted as GCDAN-tc), and the third one contains both of them (denoted as GCDAN-full). The experimental results on three datasets in terms of Acc@1 are shown in Table 5. We can see that GCDAN-base achieves higher performance compared with other deep models such as LSTM and ST-RNN (refer to Table 2-Table 4), which demonstrates the dual-attentive mechanism has stronger sequential modeling capacity. Meanwhile, we observe that for all datasets, GCDAN-gc and GCDAN-tc outperform GCDAN-base, which means that both graph convolution and trajectory correlation are helpful for mobility prediction and have positive effect to our framework. Besides, we find that GCDAN-full performs best among all variants, which demonstrates that they can work together to improve the prediction performance.

5 RELATED WORK

Human mobility prediction: Early approaches are designed to find mobility patterns in trajectories and then make prediction accordingly [11, 22, 35, 36]. Matrix factorization is widely applied to this problem [3, 16, 17, 34]. Cheng *et al*. [3] first fuse matrix factorization with geographical and social influence for location recommendation. Lian *et al*. [17] take the spatial clustering in human mobility into consideration and exploit the weighted matrix factorization method. However, these approaches generally fail to capture fine-grained sequential and periodical characteristics in human mobility. Due to the capacity to model sequential behaviors, Markov chain is also usually used for mobility prediction [4, 19, 23, 26, 37]. For example, Mathew *et al*. [19] present a hybrid method that clusters location histories according to their characteristics and then trains an Hidden Markov Model (HMM) for each cluster to predict human mobility. Nevertheless, these model-based approaches are difficult to model complex and high-order sequential transition regularity in trajectories. With the rapid development of deep learning, deep models, particularly RNNs are proved to be outstanding in trajectory sequential modeling. Liu *et al*. [18] propose ST-RNN, which extends RNN to model temporal and spatial features. Yao *et al*. [33] present SERM, a RNN based model that handles multiple features in human mobility. Feng *et al*. [8] propose an attentional recurrent network for mobility prediction. Recently, CNNs are also adopted to model human mobility. Gao *et al*. [10] use CNN to capture individual's long-term moving patterns and Miao *et al*. [20] regard trajectories as images and propose an attentive convolutional network model for trajectory prediction. Despite their impressive performance, they cannot cope well with the sparsity and inaccuracy of the trajectory information. In addition, previous works do not consider the correlation between different trajectories. In this paper, we propose a novel framework GCDAN, which employs the graph convolution and dual-attentive mechanism to overcome the above challenges.

Graph Neural Networks: GNNs generalize the convolution operation from traditional data (*e.g*., images) to graph data, and they could be categorized as spectral-based and spatial-based methods [1, 30, 31, 38]. Bruna *et al*. [2] propose the first generation spectral-based GNN based on spectrum of the graph Laplacian. Defferrard *et al*. [7] then adopt a K-order Chebyshev polynomial to approximate the convolutional filter, which greatly decreases the computational complexity. Kipf *et al*. [15] further adopt a localized first-order approximation to constrain the number of parameters and overcome overfitting. Spatial-based GNNs aggregate neighbor information in spatial domain [12, 13]. For example, Gilmer *et al*. [12] propose message passing neural networks (MPNNs) that generalize spatial-based methods with a message-passing mechanism. Inspired by these works, in this paper, we design the bidirectional diffusion graph convolution to handle the sparse and inaccurate location information during the mobility prediction.

6 CONCLUSION

In this paper, we propose a novel framework GCDAN for human mobility prediction. GCDAN employs graph convolution to preserve the spatial dependency in location representations, which could mitigate the sparsity and inaccuracy of the trajectory data. Furthermore, it employs a dual-attentive mechanism to model the long-range sequential dependency within a trajectory and the correlation between different trajectories, which could handle the high-order nature of the trajectory data. Extensive experiments on three real-world datasets show that GCDAN significantly outperforms state-of-the-art baselines. As to the future works, we would like to further introduce the relationships across users and group behavior into the modeling of mobility prediction.

ACKNOWLEDGMENTS

The authors would like to thank those who support and give help in this work, also to thank the anonymous reviewers for their comments. Haibo Wang is corresponding author.

REFERENCES

[1] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. 2017. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine* 34, 4 (2017), 18–42.

[2] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. 2013. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203* (2013).

[3] Chen Cheng, Haiqin Yang, Irwin King, and Michael R Lyu. 2012. Fused matrix factorization with geographical and social influence in location-based social networks.. In *Aaai*, Vol. 12. 17–23.

[4] Chen Cheng, Haiqin Yang, Michael R Lyu, and Irwin King. 2013. Where you like to go next: Successive point-of-interest recommendation. In *Twenty-Third international joint conference on Artificial Intelligence*.

[5] Eunjoon Cho, Seth A Myers, and Jure Leskovec. 2011. Friendship and mobility: user movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. 1082–1090.

[6] Cisco.Com/. 2018. *Cisco Annual Internet Report (2018–2023) White Paper.* Retrieved March 9, 2020 from https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html

[7] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*. 3844–3852.

[8] Jie Feng, Yong Li, Chao Zhang, Funing Sun, Fanchao Meng, Ang Guo, and Depeng Jin. 2018. Deepmove: Predicting human mobility with attentional recurrent networks. In *Proceedings of the 2018 world wide web conference*. 1459–1468.

[9] Sébastien Gambs, Marc-Olivier Killijian, and Miguel Núñez del Prado Cortez. 2012. Next place prediction using mobility markov chains. In *Proceedings of the first workshop on measurement, privacy, and mobility*. 1–6.

[10] Qiang Gao, Fan Zhou, Goce Trajcevski, Kunpeng Zhang, Ting Zhong, and Fengli Zhang. 2019. Predicting human mobility via variational attention. In *The World Wide Web Conference*. 2750–2756.

[11] Fosca Giannotti, Mirco Nanni, Fabio Pinelli, and Dino Pedreschi. 2007. Trajectory pattern mining. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. 330–339.

[12] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for quantum chemistry. *arXiv preprint arXiv:1704.01212* (2017).

[13] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in neural information processing systems*. 1024–1034.

[14] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[15] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[16] Xutao Li, Gao Cong, Xiao-Li Li, Tuan-Anh Nguyen Pham, and Shonali Krishnaswamy. 2015. Rank-geofm: A ranking based geographical factorization method for point of interest recommendation. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*. 433–442.

[17] Defu Lian, Cong Zhao, Xing Xie, Guangzhong Sun, Enhong Chen, and Yong Rui. 2014. GeoMF: joint geographical modeling and matrix factorization for point-of-interest recommendation. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 831–840.

[18] Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. Predicting the next location: A recurrent model with spatial and temporal contexts. In *Thirtieth AAAI conference on artificial intelligence*.

[19] Wesley Mathew, Ruben Raposo, and Bruno Martins. 2012. Predicting future locations with hidden Markov models. In *Proceedings of the 2012 ACM conference on ubiquitous computing*. 911–918.

[20] Congcong Miao, Ziyan Luo, Fengzhu Zeng, and Jilong Wang. 2020. Predicting Human Mobility via Attentive Convolutional Network. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. 438–446.

[21] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[22] Anna Monreale, Fabio Pinelli, Roberto Trasarti, and Fosca Giannotti. 2009. Wherenext: a location predictor on trajectory pattern mining. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. 637–646.

[23] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World wide web*. 811–820.

[24] Andrey Rudenko, Luigi Palmieri, Michael Herman, Kris M Kitani, Dariu M Gavrila, and Kai O Arras. 2020. Human motion trajectory prediction: A survey. *The International Journal of Robotics Research* 39, 8 (2020), 895–935.

[25] Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing* 45, 11 (1997), 2673–2681.

[26] Hongzhi Shi, Yong Li, Hancheng Cao, Xiangxin Zhou, Chao Zhang, and Vassilis Kostakos. 2019. Semantics-aware hidden Markov model for human mobility. *IEEE Transactions on Knowledge and Data Engineering* (2019).

[27] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*. 5998–6008.

[28] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).

[29] Haibo Wang, Jessie Hui Wang, Jilong Wang, Weizhen Dang, Fenghua Li, Jinzhe Shan, et al. 2020. Squeezing the gap: An empirical study on DHCP performance in a large-scale wireless network. *IEEE/ACM Transactions on Networking* 28, 2 (2020), 832–845.

[30] Haibo Wang, Chuan Zhou, Xin Chen, Jia Wu, Shirui Pan, and Jilong Wang. 2020. Graph stochastic neural networks for semi-supervised learning. *Advances in Neural Information Processing Systems* (2020).

[31] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems* (2020).

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

[33] Di Yao, Chao Zhang, Jianhui Huang, and Jingping Bi. 2017. Serm: A recurrent model for next location prediction in semantic trajectories. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 2411–2414.

[34] Lina Yao, Quan Z Sheng, Yongrui Qin, Xianzhi Wang, Ali Shemshadi, and Qi He. 2015. Context-aware point-of-interest recommendation using tensor factorization with social regularization. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*. 1007–1010.

[35] Josh Jia-Ching Ying, Wang-Chien Lee, Tz-Chiao Weng, and Vincent S Tseng. 2011. Semantic trajectory mining for location prediction. In *Proceedings of the 19th ACM SIGSPATIAL international conference on advances in geographic information systems*. 34–43.

[36] Chao Zhang, Jiawei Han, Lidan Shou, Jiajun Lu, and Thomas La Porta. 2014. Splitter: Mining fine-grained sequential patterns in semantic trajectories. *Proceedings of the VLDB Endowment* 7, 9 (2014), 769–780.

[37] Chao Zhang, Keyang Zhang, Quan Yuan, Luming Zhang, Tim Hanratty, and Jiawei Han. 2016. Gmove: Group-level mobility modeling using geo-tagged social media. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1305–1314.

[38] Ziwei Zhang, Peng Cui, and Wenwu Zhu. 2020. Deep learning on graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering* (2020).