Copyright ⓒ 2020 KSII

Next Location Prediction with a Graph Convolutional Network Based on a Seq2seq Framework

Jianwei Chen, Jianbo Li*, Manzoor Ahmed, Junjie Pang, Minchao Lu and Xiufang Sun
Computer Science and Technology, Qingdao University
Qingdao, CN 266071- China
[e-mail: jianweichen0106@gmail.com;lijianbo@188.com;manzoor.achakzai@gmail.com;pangjj18@163.com;
yaoyaoch1@163.com;sunflowerxf@163.com]
*Corresponding author: Jianbo Li

*Received February 3, 2020; revised March 4, 2020; accepted April 3, 2020;
published May 31, 2020*

Abstract

Predicting human mobility has always been an important task in Location-based Social Network. Previous efforts fail to capture spatial dependence effectively, mainly reflected in weakening the location topology information. In this paper, we propose a neural network-based method which can capture spatial-temporal dependence to predict the next location of a person. Specifically, we involve a graph convolutional network (GCN) based on a seq2seq framework to capture the location topology information and temporal dependence, respectively. The encoder of the seq2seq framework first generates the hidden state and cell state of the historical trajectories. The GCN is then used to generate graph embeddings of the location topology graph. Finally, we predict future trajectories by aggregated temporal dependence and graph embeddings in the decoder. For evaluation, we leverage two real-world datasets, Foursquare and Gowalla. The experimental results demonstrate that our model has a better performance than the compared models.

Keywords: Location Prediction, Spatial-Temporal Dependence, Seq2seq Framework, GCN

This research was supported in part by National Key Research and Development Plan Key Special Projects under Grant No. 2018YFB2100303, Key Research and Development Plan Project of Shandong Province under Grant No. 2016GGX101032, Program for Innovative Postdoctoral Talents in Shandong Province under Grant No. 40618030001, National Natural Science Foundation of China under Grant No. 61802216, and Postdoctoral Science Foundation of China under Grant No.2018M642613.

http://doi.org/10.3837/tiis.2020.05.003 **ISSN : 1976-7277**

1. Introduction

With the rapid development of mobile device hardware, many location-based services easily collect various human activities records via intelligent equipment, including the number of walking steps and geographic location. Subsequently, these datasets can be explored for hidden information (such as user preference). Then, we can predict human movement from the information [1]. This research has social contributions as well. Prediction of human mobility provides better services, such as recommender systems. Moreover, such explorations of these datasets are useful to social security, such as recognizing human activity recognition [2] and preventing emergencies.

Currently, human mobility prediction has been extensively studied [3,4,5]. By studying the mobility patterns of anonymized mobile phone users, Song et al. [6] found a 93% potential predictability in user mobility. Besides, many research efforts on mobility prediction have been conducted based on this high predictability [7,8,9]. In the early methods, many research efforts predicted future trajectories with predefined mobile patterns, such as periodic patterns and most frequently accessed [10,11,12,13]. These methods consider fewer features and ignore complex transition regularities; therefore, they usually cannot obtain the ideal result. After that, many research efforts focused on designing predictive models [14,15,16]. For example, the Markov chain model is a commonly used method; it can capture the transition regularities of historical trajectories.

Although the above methods have inspiring results, there are still several key problems to be solved. First, the transition regularities are usually not simple; they are not only complex but also aggregate many mobility patterns. Second, many researchers only consider the temporal dependence [17,18,19] while ignoring the spatial dependence of continuous locations. The association between locations has an unexpected influence on future trajectories; for example, if some people went to some point of interests (POIs) such as entertainment venues, they may continue to play or relax. If we only predict the future trajectories according to the sequential transition regularities, we may obtain incorrect results in predicting continuing to play due to visiting many entertainment venues during this period. However, they may be ready to relax. Therefore, the spatial dependence of historical trajectories is needed to help us understand the human movement. Furthermore, the collected data tend to be incomplete, which leads to the sparsity of datasets. In other words, these problems collectively make it difficult for us to explore various patterns and build a model.

To solve these problems, we propose a new deep learning method to predict the next location. An embedding technology was used to handle the problem of sparsity. We convert sparse location features (location id) into dense representations and take the latter as inputs of the model. Meanwhile, we build our model based on the seq2seq framework. We take the historical trajectory and current trajectory as the input of the encoder and decoder, respectively. With this method, this framework considers the impact of past and current trajectory, which helps us deeply capture the sequential transition regularities. Besides, we introduce a GCN model to capture spatial dependence of historical trajectory, which can extract the correlation of discrete locations. When we capture the temporal dependence from the seq2seq framework, we aggregate the temporal dependence and spatial dependence to perform our prediction task.

Our salient contributions are summarized as follows:

(1) We propose a new deep learning method with GCN based on the seq2seq framework, where GCN captures the spatial dependence between historical trajectory and the seq2seq framework is used to capture the time dynamics of the whole trajectory.

(2) We use a method to aggregate spatial dependence and temporal dependence with the GCN and seq2seq framework. Specifically, we generate a graph embedding with a mean-pooling strategy from GCN, and then we update a cell state generated from the encoder with graph embeddings.

(3) We evaluate our model on the Foursquare and Gowalla datasets. The results demonstrate that our model has a better performance compared with the traditional and other strong deep learning methods, which shows our model's effectiveness on location prediction.

The rest of the paper is organized as follows. We review relevant studies about location prediction in Section 2. Then, we detail our methods, which include the detailed architecture of the GCN and the seq2seq framework, in Section 3. Subsequently, we evaluate our model with real-world check-in datasets in Section 4. We also introduce the parameter settings of our model and analyze the results. Finally, we conclude the paper and introduce our future work in Section 5.

2. Related Work

In this section, we review several types of methods for location prediction, including model-based methods and pattern-based methods. Besides, we introduce related works of the seq2seq framework and the GCN.

Existing research efforts can be divided into two categories: model-driven methods and pattern-driven methods. First, the model-driven methods mainly explore the transition regularities of trajectories, including instantaneous and steady-state relationships. The commonly used model-driven methods are the Markov model and its variants [14,20,21,22]. The Markov model predicts future trajectories by building a transition matrix, which only captures the simple transition regularities, which leads to poor performance. Therefore, many variants of this model have been proposed. For example, Mathew et al. [14] clustered the historical locations according to their characteristics and trained an HMM for each cluster. These Markov-based methods had significant results on specific scenes, but they cannot capture the temporal dependence and deeply extract the sequential regularity.

Second, pattern-driven methods mainly discover the mobility patterns of trajectories and then predict future locations according to the analysis of patterns [23,24,25,26,27]. They tend to discover the relationship between users and excavate trajectory patterns. For example, Monreale et al. [23] proposed a location predictor T-pattern decision tree that finds the best matching path in the tree. Meanwhile, matrix factorization (MF) and MF-based methods [25,26] are frequently commonly used methods. Compared with pattern-driven methods, our model can share sequential transition regularities with all users due to deep learning technology.

With the rapid development of artificial intelligence, deep learning methods have received more attention. For example, the recurrent neural network (RNN) is good at processing time-series data. Besides, it has achieved great success in many research efforts, including language models [28], text processing and forecast tasks [29,30]. To date, many researchers have predicted human movement with RNN [9,17,18,19,31]. For example, Liu et al. [17] proposed a spatiotemporal RNN (ST-RNN) to model time and location; this model considers the geographic distance to predict the location. Then, Yao et al. [18] proposed a semantic

enriched recurrent model (SERM), which learns the embedding of multiple factors (user, location, time) and captures the semantic-aware spatiotemporal transition regularities. Compared with the above models, our model predicts location based on the seq2seq framework, which can deeply capture temporal dependence and transition regularities. We also use a graph convolutional network to capture spatial dependence between discrete locations from the historical trajectory.

In this paper, we build our model based on the seq2seq framework, which was proposed by Sutskever et al. [32] and was first used in machine translation tasks, which significantly improves the translation accuracy. Besides, the seq2seq framework has received more attention in various fields [33,34,35]. In the human mobility prediction area, Feng et al. [35] first proposed an attentional mechanism based on the seq2seq framework named DeepMove to predict the future location, which captures the multi-level periodicity of historical trajectories. Compared with DeepMove, we consider the spatial dependence of historical trajectory. In particular, we add a GCN architecture to perform the prediction task. GCN was proposed by Kipf et al. [36], which can capture the topologic structure of the graph network well. Meanwhile, GCN has proven to be effective in traffic forecasting tasks [37,38].

3. Methodology

In this section, we first formulate the problem in Subsection 3.1 and present an overview of our solution in Subsection 3.2. Then, we detail two important modules of our model. Subsection 3.3 introduces the GCN and the method of capturing spatial dependence. Subsequently, the method for extracting temporal dependence with the seq2seq framework is detailed in Subsection 3.4. Finally, we aggregate the two methods to build our model for prediction task in Subsection 3.5.

3.1 Problem Formulation

In this paper, we aim to predict the next location of the users (human). For this, we first collect historical locations of each user as their trajectories and then divide the whole trajectories into the historical trajectory and current trajectory. In this paper, we fix the time interval and take predicting the next location as the only task. Before going into detail, we will define trajectories' set *M*, and location graph set *G* first.

Definition 1: (Trajectories' set *M*). First, we create a set of users denoted as $Userset = \{u_1, u_2, u_3, \dots, u_m\}$. Then, we collect the trajectory of each user according to an hour's interval. Next, we divide each user's trajectory into many sub-trajectories with a time span of one week. Note that the period of each sub-trajectory is no more than one week, and the time span between such sub-trajectories may exceed one week. For convenience, we take the example of u_1 whose trajectories are set as $S_{u_1} = [S_{u_1}^1, S_{u_1}^2, S_{u_1}^3, \dots, S_{u_1}^{|S_{u_1}|}]$. In each sub-trajectory of S_{u_1}, we create a location trajectory $S_{u_1}^t = [l_{u_1}^{t_1}, l_{u_1}^{t_2}, l_{u_1}^{t_3}, \dots, l_{u_1}^{t_{T-1}}, l_{u_1}^{t_T}]$ where $l_{u_1}^{t_1}$ indicates the location where u_1 went at the t_T^{th} time, and T is the number of locations in this sub-trajectory. To obtain the complete set of trajectories, we generate all the users' sub-trajectories in the same manner, which is denoted as '*M*' and expressed as $M = \{S_{u_1}, S_{u_2}, S_{u_3}, \dots, S_{u_m}\}$.

Definition 2: (Location graph set *G*). As in Definition 1, we make a location trajectory S_u^t. Assume S_u^t is the current trajectory of u, location graph G_u corresponds to users' historical trajectory, i.e., $G_u = (V_u, E_u)$, where V_u is a set of locations, and E_u is a connection between locations. $V_u = set(S_u^1 + S_u^2 + \cdots + S_u^{t-1})$, here, the $+$ denotes merging of two

sub-trajectories. Similarly, E_u is a set of edges. We make the interconnection of every u's location sequence as the edges of G_u; in other words, we generate the edges by sequence order of locations, and we do not consider the duplication between locations here. In addition, we generate an adjacency matrix A_u to represent the connection of every location, the adjacency matrix A_u only has two elements 0 and 1, where '1' means a link exists between the two locations and '0' shows no link between the two locations, i.e., $A_u \in \mathrm{R}^{N_{V_u} * V_u}$, N_{V_u} is the number of locations in V_u. Therefore, the location graph's set is represented as, $G = \{G_{u_1}, G_{u_2}, G_{u_3}, \dots, G_{u_m}\}$.

Problem Formulation: In this paper, we predict the next location via location graph leveraging both the historical and current trajectories of S_u. Specifically, given the historical trajectories $S_u^1, S_u^2, \dots, S_u^{t-1}$ and current trajectory $l_u^{t_1}, l_u^{t_2}, l_u^{t_3}, \dots, l_u^{t_{T-1}}$ of S_u^t, and considering the location graph G_u, we predict the next location $l_u^{t_T}$.

3.2 Overview

In this section, we overview our solution for mobility prediction. Our solution is based on a variant of RNN, and the RNN model captures temporal dependence well due to its unique architecture. However, the length of the trajectory of each user tends to be very long, which has a negative influence on the RNN's performance. Considering the results of [35], we observe the RNN's sensitivity towards the trajectories' length; therefore, we also limit the trajectories' length.

Intuitively, we can perform this prediction task on a single RNN model, but it cannot perform well. The first reason is the sensitivity of long-range dependence; we have mentioned the shortage of RNN on long sequences. In this paper, we consider the impact of the whole trajectory to predict mobility, but when the length of the trajectory is very long, RNN can only capture the influence of the recent trajectory. Another reason is the sparsity of data. The trajectories collected tend to be incomplete, which has a negative influence on data continuity, which makes it more difficult for the single RNN to capture the sequential transition regularities. In short, the long-term nature and sparsity of data make it difficult for the single RNN to achieve good performance.

Based on the aforementioned facts, we build our model based on a seq2seq framework. We use the encoder and decoder to capture the impact of historical and current trajectory, respectively, which effectively alleviates the shortage of the single RNN model. Then, an embedding layer is used to process the sparsity feature (location id), and we utilize the Bidirectional Long Short-Term Model (Bi-LSTM) as the basic architecture of the encoder, which improves the ability to capture sequential transition regularities. Furthermore, we introduce a GCN model that captures the spatial dependence between locations in the historical trajectory, and we generate the graph embedding of the location graph. The bridge (the cell state) of the encoder and decoder enables us to aggregate spatial-temporal dependence. The updated cell state and current trajectory are then fed into the decoder. Finally, we predict human mobility via spatial-temporal dependence.

3.3 Spatial dependence modeling

Introduction of GCN

We consider a convolutional filter from a spectral approach on graphs, which is defined as the multiplication of graph signal x with a filter g_θ, i.e.,

$$g_\theta * x = g_\theta(L)x = g_\theta(U \Lambda U^T)x \tag{1}$$

where $L = I_1 - D^{-\frac{1}{2}}AD^{-\frac{1}{2}} = U\Lambda U^T$ is the normalized graph Laplacian, $x \in \mathrm{R}^N$, and U is an eigenvector matrix of L. From the definition of L, I_1 is the identity matrix, $D_{ii} = \sum_j A_{ij}$ is the degree matrix, and Λ is a diagonal matrix of L's eigenvalues. The calculation of matrix multiplication causes a problem of high computational complexity. Therefore, Hammond et al. [39] proposed a K^{th} order polynomial filter $g_{\theta'}(\Lambda)$, i.e.,

$$g_{\theta'}(\Lambda) \approx \sum_{k=0}^{K} \theta'_k T_k(\tilde{\Lambda}) \tag{2}$$

with $\tilde{\Lambda} = \frac{2}{\lambda_{max}}\Lambda - I_N$, λ_{max} is the largest eigenvalue of L. $\theta'_k \in \mathrm{R}^N$ is a vector of Chebyshev coefficients, the $T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$ with $T_0(x) = 1$ and $T_1(x) = x$. Through the above definition and filter $g_{\theta'}$, $g_{\theta'} * x$ is

$$g_{\theta'} * x = g_{\theta'}(L)x \approx \sum_{k=0}^{K} \theta'_k T_k(\tilde{L})x \tag{3}$$

Kipf et al. [36] proposed a layer-wise linear model, and he limited the convolution operation to K=1 and approximate $\lambda_{max} \approx 2$, so that $g_{\theta'} * x$ is a linear function of L, i.e.,

$$g_{\theta'} * x \approx (\theta'_0 - \theta'_1 D^{-\frac{1}{2}}AD^{-\frac{1}{2}})x \tag{4}$$

Subsequently, he set the θ'_0 and θ'_1 of the linear function as a single parameter $\theta = \theta'_0 = -\theta'_1$, i.e.

$$g_{\theta'} * x \approx \theta(I_N + D^{-\frac{1}{2}}AD^{-\frac{1}{2}})x \tag{5}$$

In this paper, we mainly consider directed graphs as input x. For directed graphs, we take a nonsymmetric normalized adjacency matrix in the preprocess step [40], i.e., $D^{-1}A$. For $X \in \mathrm{R}^{N*F}$ with N locations, F-dimensional feature vectors and M filters, and Z is the convolved graph, which is defined as:

$$Z = D^{-1}AX\Theta \tag{6}$$

where $Z \in \mathrm{R}^{N*M}$ and $\Theta \in \mathrm{R}^{F*M}$ is a matrix for filter parameters.

Capturing spatial dependence

Many experiments [41,42] have proven that the traditional CNN performs well in obtaining spatial features, and the convolution kernel obtains the spatial feature map by calculating the weighted sum of the central pixel and the adjacent pixel. Therefore, CNN plays an important role in processing Euclidean space, such as images and video data. In this paper, our location graph is a form of the graph rather than images or a two-dimensional grid, and the spatial dependence of locations in the location graph cannot be captured with a CNN. As mentioned above, CNN performs well on Euclidean structured data. However, we have many non-Euclidean structured data in real life, such as social networks and information networks. Due to the translational variance in these non-Euclidean structured data, we cannot use a convolution kernel of the same size to perform convolution operations. In other words, CNN cannot perform well on obtaining spatial features in our location graph. Therefore, we use the GCN to capture spatial dependence in this paper.

The GCN constructs a filter Z in the Fourier transform, Z acts on the location graph and its first-order neighborhood to capture the spatial dependence between locations. Specifically, taking a user u as an example, we input the feature matrix X_u into the GCN $Z(X_u, A_u)$ to learn the spatial features from the location graph, $Z(X_u, A_u)$ is expressed as:

$$Z(X_u, A_u) = \sigma(D_u^{-1}\tilde{A}_u X_u W) \tag{7}$$

where $\tilde{A}_u = A_u + I_1$ is a self-connection structure, and σ is the activation function, such as Tanh and ReLU. A_u is the adjacency matrix of the location graph, W represents the weight matrix in this layer, and D_u is the out-degree diagonal matrix of A_u. Additionally, we describe the architecture for capturing the spatial dependence in **Fig. 1**. As defined in Subsection 3.1,

$G_u = (set(S_u^1 + S_u^2 + \cdots + S_u^{t-1}), edgeset(S_u^1 + S_u^2 + \cdots + S_u^{t-1}))$. The period corresponds to the value of t in the training process. Similarly, assume that S_u^t is the current trajectory. Currently, the inputs to the GCN are a feature matrix X_u that contains all locations in $set(S_u^1 + S_u^2 + \cdots + S_u^{t-1})$ and its adjacency matrix A_u. Since we only consider one feature, that is, the location, we first take the X_u into an embedding layer to generate a dense representation matrix, and then the embedded $X_u \in \mathrm{R}^{N*F}$ is fed into the GCN. Second, location embedding is the convolved graph, $L_1, L_2, \ldots, L_N$ and $x_1, x_2, \ldots, x_F$ denote all locations in G_u and the features, respectively. Next, we feed the location spatial feature matrix from the graph convolution process into a fully connected (FC) neural network and generate graph embedding. Here, we use the pooling strategy method on the matrix element from the output of the FC layer to generate graph embedding as in [43]. Hence, in this paper, we adopt the mean-pooling strategy to generate graph embedding.

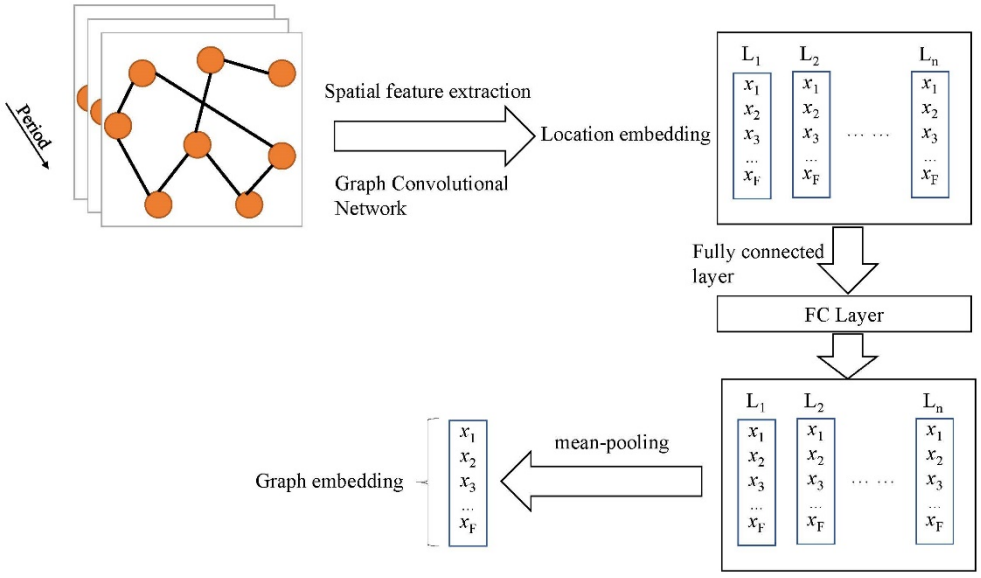


Fig. 1. Capturing spatial dependence with the GCN

3.4 Temporal dependence modeling

Seq2seq framework

The seq2seq framework is an important variant of RNN. Due to the limit of the traditional RNN model, the RNN needs the inputs and outputs to have the same length. However, many natural language processing tasks usually have different lengths of inputs and outputs. Many researchers solve this problem by padding the inputs or outputs of short lengths to train the data, but this method causes a problem of low accuracy. Therefore, the seq2seq framework was proposed and first applied to the machine translation task. Bahdanau et al. [44] proved its effectiveness and obtained significant results on this task. Specifically, it includes three parts, i.e., encoder, cell state, and decoder. The encoder receives the input and generates the hidden state at the last time step (cell state), then the decoder receives the cell state and feeds its unique input to generate outputs. In a word, the bridge of the encoder and decoder (cell state) records the information of the encoder's input and is used as reference information during the training of the decoder.

Capturing temporal dependence

RNN is a neural network that connects nodes and forms a directed graph along a temporal sequence. Due to its special architecture, it is applicable for capturing the temporal dependence of time-series data. However, it also leads to the problem of vanishing gradient and exploding gradient. These two problems lead to the difficulty of long input memory storage and model training. To solve these problems, LSTM and Gated Recurrent Unit (GRU) were proposed. They are all variants of RNN, and the LSTM and GRU solve two problems by constructing a gated mechanism. We introduce only LSTM here due to the similar architecture of the two models. As shown in **Fig. 2**, LSTM has a very important module named cell state, and it updates by a small number of linear interactions over time. Furthermore, it has three gates to remove or add information to the cell state, which includes the forget gate, input gate, and output gate. h_{t-1} is the hidden state at $t-1$, x_t is the input at time t. f_t is the forget gate and controls forget information selectivity. i_t is the input gate and controls which part of the information needs to be updated, c_t' is candidate vector, then c_t is updated by f_t, i_t and c_t'. o_t is the output gate and controls which part of cell state information will output. The calculation formulas of LSTM are as follows:

$$i_t = \sigma(W_{ii} * x_t + b_{ii} + W_{hi} * h_{t-1} + b_{hi}) \tag{8}$$

$$f_t = \sigma(W_{if} * x_t + b_{if} + W_{hf} * h_{t-1} + b_{hf}) \tag{9}$$

$$c_t' = tanh(W_{ic'} * x_t + b_{ic'} + W_{hc'} * h_{t-1} + b_{ho}) \tag{10}$$

$$o_t = \sigma(W_{io} * x_t + b_{io} + W_{ho} * h_{t-1} + b_{ho}) \tag{11}$$

$$c_t = f_t * c_{t-1} + i_t * c_t' \tag{12}$$

$$h_t = o_t \odot \tanh(c_t) \tag{13}$$

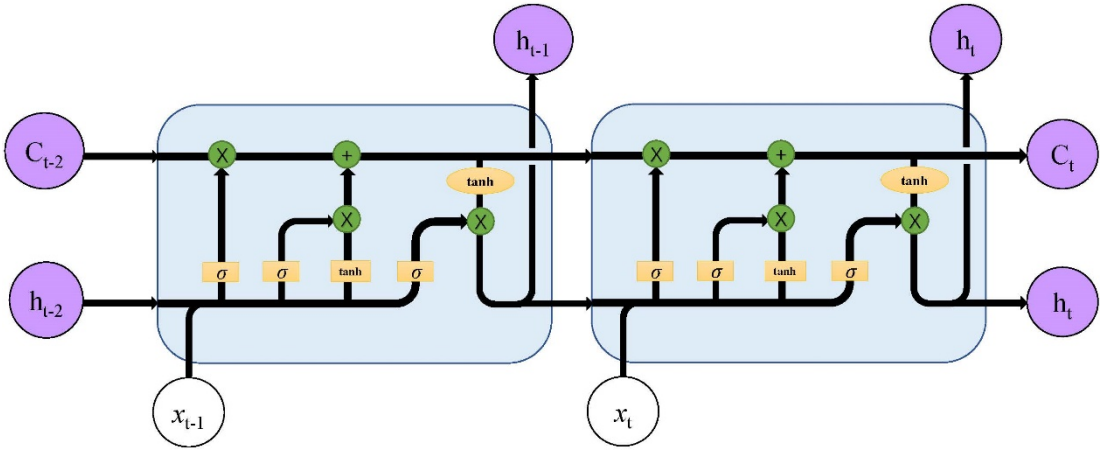


Fig. 2. The LSTM architecture

We capture the temporal dependence based on LSTM architecture. The feature (location) matrix of the historical and current trajectory is first fed into the embedding layer to generate the dense representation. Then, the embedded historical trajectory ($S_u^1 + S_u^2 + \cdots + S_u^{t-1}$) is fed into the encoder to obtain the cell state that subsequently contains the temporal dependence of the historical trajectory. Finally, we take the cell state and embedded current trajectory $l_u^{t_1}, l_u^{t_2}, l_u^{t_3}, \ldots, l_u^{t_{T-1}}$ as the input of the decoder to obtain the next point $l_u^{t_T}$ of the current trajectory.

3.5 Our model

We introduced two important modules above, but it is a key point to aggregate them. To predict the next location of a person with spatial-temporal dependence at the same time, we propose a model with a GCN based on the seq2seq framework. As shown in **Fig. 3**, our model contains three parts: encoder, GCN and decoder. For the current trajectory S_u^t, we take the user u's historical trajectory $S_u^1 + S_u^2 + \cdots + S_u^{t-1}$ and location graph G_u as the inputs of encoder and GCN model, respectively. Once the encoder captures the temporal dependence of the historical trajectory, the GCN model captures the spatial dependence of G_u at the same time. Subsequently, we aggregate the two dependences into the cell state vector. Finally, we feed the current trajectory $l_u^{t_1}, l_u^{t_2}, l_u^{t_3}, \ldots, l_u^{t_{T-1}}$ of S_u^t into the decoder and the predicted trajectory $l_u^{t_2}, l_u^{t_3}, l_u^{t_4}, \ldots, l_u^{t_T}$ is obtained, where $l_u^{t_T}$ is the predicted next location. Our model learns a mapping function f with historical trajectories S_u and location graph G_u, which is described as:

$$\left[l_u^{t_2}, l_u^{t_3}, l_u^{t_4}, \ldots, l_u^{t_T}\right] = f\{(l_u^{t_1}, l_u^{t_2}, l_u^{t_3}, \ldots, l_u^{t_{T-1}}) | S_u^{t-1}, S_u^{t-2}, \ldots, S_u^1; G_u\} \tag{14}$$

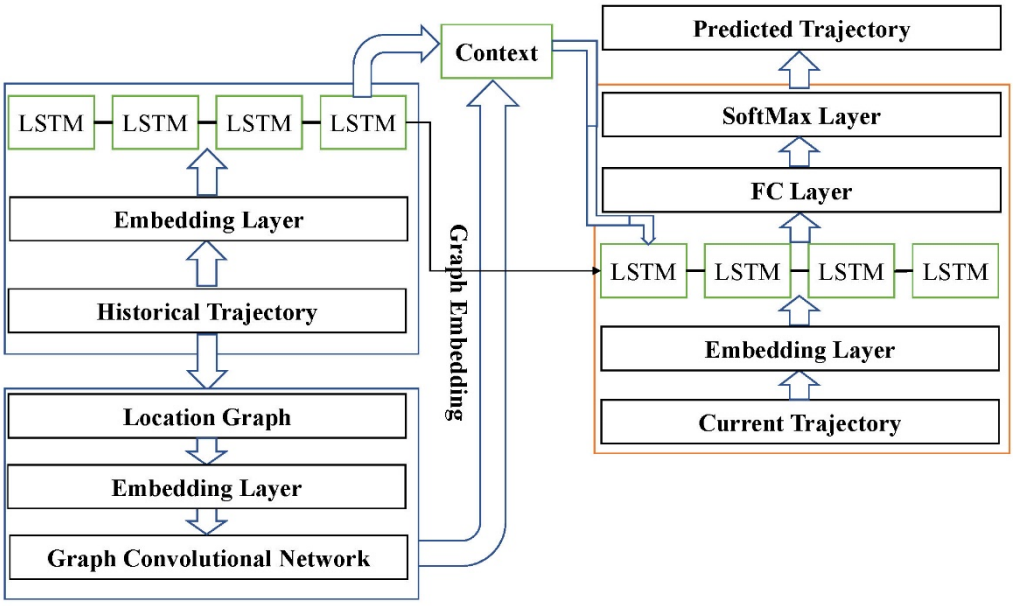


Fig. 3. Our prediction model

When we generate the graph embedding of G_u, the encoder generates the cell state and hidden state at the last time step. We aggregate the graph embedding and cell state vector with an add function and generate an updated cell state vector. Therefore, the inputs of the decoder contain three parts: hidden state, updated cell state vector and the current trajectory. Then, we add a fully connected network to generate the probability distribution of predicted locations. Finally, for the probability distributions of predicted locations, we add a log SoftMax layer to predict the next location.

When we feed the historical trajectory $S_u^1 + S_u^2 + \cdots + S_u^{t-1}$ with the length of η into our model, the calculation formulas of the model are as follows:

$$h_{t_1} = h_{t-1_\eta} = f(h_{t-1_{\eta-1}}, l_u^{t-1_{\eta-1}}) \tag{15}$$

$$c_{t_1} = add(c_{t-1_\eta}, Z(X_u, A_u)) \tag{16}$$

$$p\left(l_u^{t_T} \middle| l_u^{t_{T-1}}, \ldots, l_u^{t_2}, l_u^{t_1}\right) = g(h_{t_{T-1}}, l_u^{t_{T-1}}; c_{t_{T-1}}) \tag{17}$$

where $h_{t_{T-1}}$ and $c_{t_{T-1}}$ are the hidden state and the cell state of the decoder, respectively, the initial value h_{t_1} is the hidden state at the last time step from the encoder, and $Z(X_u, A_u)$ is the graph convolution process of G_u. c_{t_1} is the initial updated cell state vector, and Eq. (17) denotes the prediction process in the decoder. f and g denote the activate function. The next location $l_u^{t_T}$ corresponds to the historical information, and recent information.

In our model, we consider the prediction process as a classification problem. The output of the decoder classifies the predicted locations, and the number of categories is the number of locations that all users visited. We take a negative log-likelihood loss as the loss function of the model in the training process. Furthermore, we add an $L2$ regularization term L_{reg} to the loss function, which can effectively avoid overfitting problems. The calculation of the loss function is given below:

$$L = -\frac{1}{M}\sum_{i=1}^{M}\sum_{k=1}^{K_i}\frac{1}{K_i} l[y_k] + \lambda L_{reg} \tag{18}$$

where M is the sample of the training dataset, K_i is the length of the current trajectory of the i^{th} sample, y_k is the index of the target location in the training dataset, and $L[y_k]$ denotes the value of y_k in the output vector l of the decoder. L_{reg} contains all the parameters to be learned.

4. Experiments

In this section, we evaluate our model on two real-world datasets. We first describe the two datasets and the method of data processing, coupled with the detail of the parameter settings and evaluate the metrics of our experiment. Finally, our experimental results are discussed.

4.1 Data Description

We use Foursquare and Gowalla as two real-world datasets to evaluate our model. These two datasets are all check-in datasets collected from users. These two datasets contain the user id, location id, latitude and longitude, and check-in time. We take their location id as the prediction task. The following shows the information of the two datasets:

(1) Foursquare [45] is a location technology company that provides many services, such as city guides based on user location information. This dataset was collected in New York from April 3, 2012, to March 16, 2013, which includes approximately one thousand users and forty thousand locations.

(2) Gowalla [46] is a location-based social network similar to Foursquare; users can share places, events and travel routes between friends. This dataset contains ten thousand users and one million locations from all around the world, and it was collected mainly from January 2009 to October 2010.

The basic information of the two datasets is illustrated in **Table 1**, and more details of the two datasets are shown in **Fig. 4** and **Fig. 5**. Two figures show the cumulative distribution function (CDF) and probability mass function (PMF) of the two datasets. The CDF curve of the Foursquare dataset denotes the probability distribution of location visits, which means that 90% of location visits are fewer than 20, and the PMF curve denotes that the locations with more than 20 visits are rare. Our data processing corresponds to the data information from two

curves.

Table 1. Data information

Dataset	**Foursquare**	**Gowalla**
City	New York	Null
Duration	11 months	About 2 years
Users	1083	107092
Locations	38333	1280969
Records	227427	6442892

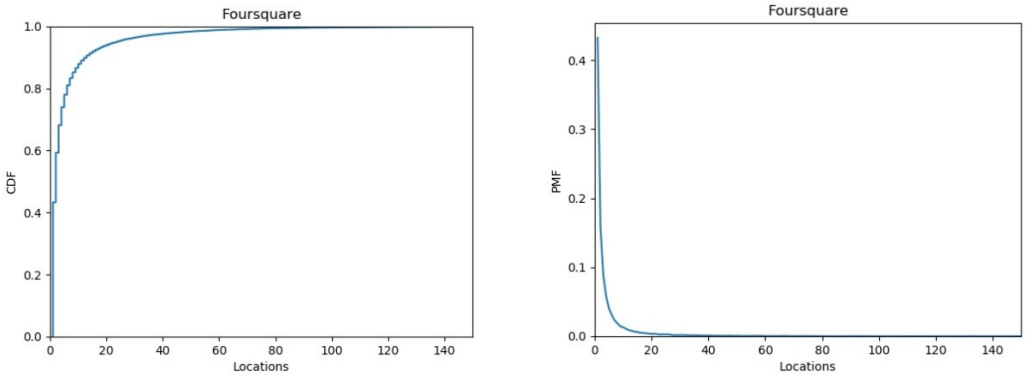


Fig. 4. The CDF and PMF of Foursquare dataset

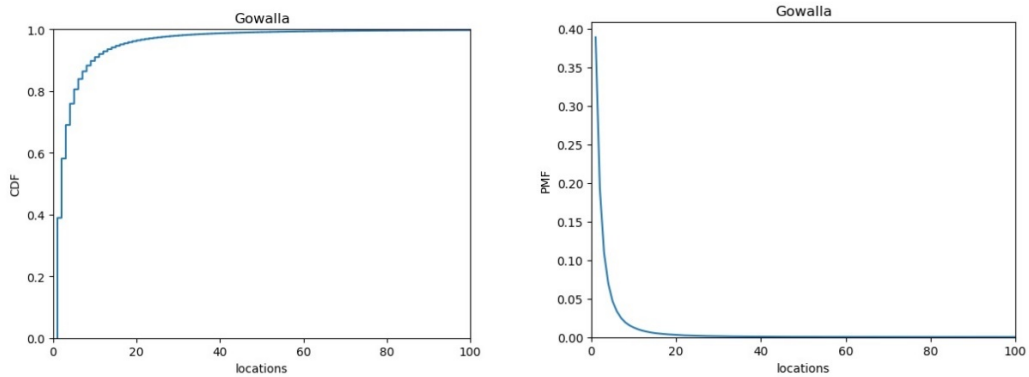


Fig. 5. The CDF and PMF of Gowalla dataset

4.2 Data processing and making training sets

We clean the two datasets, to include the following: the minimum length of each user's trajectory is 20; the minimum value of a location's visits is 10 and is set by the analysis of two figures, which can obtain a good training dataset; the minimum and maximum number of trajectory sequences is 10 and 20, respectively, which limits the length of S_u^t; and the minimum number of the sub-trajectories is 5, which limits the number of t.

Our training dataset corresponds to our model, and our model is based on the seq2seq framework. With reference to the machine translation problem, we transform our training dataset into data pairs. We consider u again, and its data pairs are $\{(S_u^1, S_u^2),(S_u^1 + S_u^2, S_u^3),\ldots\}$, which show the historical trajectory and the current trajectory. Every data pair is a sample, and we repeat the same operation on all users to generate our training dataset. Furthermore, we take 80% of the data pairs of every user as the training set and the remaining 20% of data pairs as the test set.

4.3 Parameter setting and evaluation metrics

In this section, we introduce the parameter settings and evaluation metrics of our model. To evaluate the prediction performance of our model, we use two metrics to evaluate the difference between the predicted location and the actual location. Specifically, we set the prediction accuracy Av_{ac} and log perplexity score as our evaluation metrics. However, because our dataset contains many different users, we choose the average value of the metrics for these users to evaluate our model. The calculation formulas are as follows:

$$Av_{ac} = \frac{\sum_{i=1}^{N_{user}} acc_i}{N_{user}} \tag{19}$$

$$acc_i = \frac{T_{loc}^i}{L_{tar}^i} \tag{20}$$

where T_{loc} is the number of corrected locations that the model predicted, and L_{tar} is the length of the target trajectories. N_{user} denotes the number of users, and acc_i represents the prediction accuracy of u_i. Av_{ac} is used to measure our model's prediction precision; the higher the value is, the better the performance.

$$Av_{ppl} = \frac{\sum_{i=1}^{N_{user}} ppl_i}{N_{user}} \tag{21}$$

$$ppl_i = lge^{-\frac{1}{\sum_{E \epsilon \varepsilon_{test}^i} lnp(E)}} \tag{22}$$

$$p(E) = \prod_{j=1}^{T} p(l_{u_i}^{t_T} | l_{u_i}^{t_1}, l_{u_i}^{t_2} l_{u_i}^{t_3}, \ldots, l_{u_i}^{t_{T-1}}) \tag{23}$$

where Av_{ppl} is the average perplexity of users, $p(E)$ is our prediction process of the decoder in the test dataset, and ε_{test}^i is the test dataset's current trajectory of u_i. Here, we use the base-10 logarithm as the default logarithm to calculate the perplexity, which is used to measure the quality of a sample predicted by our model. The smaller the value is, the better the prediction performance.

In this paper, we train three models at the same time. Specifically, we set the Bi-LSTM as the architecture of the encoder, and we then set two LSTM layers on the decoder. Finally, we only set one graph convolutional network layer to capture spatial dependence. To avoid overfitting, we set a regularization parameter λ. The value of λ is different for the two datasets, and we provide a reference value, as shown in **Table 2**. To avoid the training difficulty of the neural network, we set a small number of hidden units. Therefore, we set the size of hidden units and location embedding to 400 and 300, respectively, and the value of every parameter is not unique. **Table 2** shows the experimental parameters in detail.

Table 2. Experiment parameters

Parameters	Foursquare	Gowalla
Hidden layers	300	400
L2 regularization	1e-7	1e-6
Learning rate	5e-5	1e-4
Location embedding	400	300
Graph embedding	300	400
Gradient clip	3.0	2.0

4.4 Experiment results

In this section, we compare our model with baseline models and several updated neural network-based methods. (1) The Markov model is widely used to predict human trajectories for a long time, and it builds a transition matrix according to the historical trajectory to generate future location probabilities. (2) The general RNN model is a neural network-based

model, and it performs well on time-series data. Here, we use the LSTM as the basic architecture of the RNN. (3) The variant of the RNN model is one module of our model without the GCN model. We use the seq2seq framework to compare the performance with our model, and we change the architecture of the encoder. In the experiment, we take the Bi-LSTM and LSTM architecture as the comparison object. (4) SERM is an RNN model that models a semantic-aware spatial-temporal context. (5) ST-RNN is an RNN model that incorporates the spatial and temporal contexts. (6) DeepMove is an RNN model that models the heterogeneous periodicity with an attentional mechanism. In this paper, we set the Bi-LSTM-LSTM as a basic architecture of the seq2seq framework in our model.

We compare our model with these traditional and updated methods. Here, we choose two common metrics including Top@1 and Top@5 accuracy, which means that whether the actual location equals the location with the highest probability or appears in the candidate's first five locations ranked by probabilities. Additionally, we calculate the average perplexity of such models. All compared methods have been conducted three times, we choose the max value of results, and the accuracy results are presented in **Fig. 6** and **Fig. 7**.

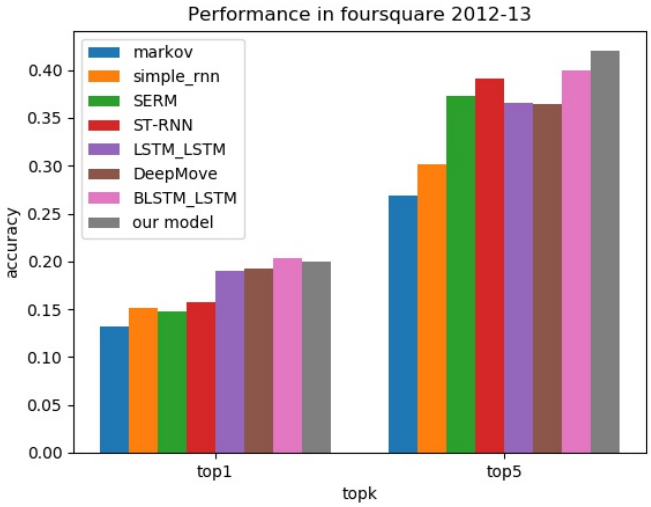


Fig. 6. Top@1 and Top@5 results on Foursquare dataset

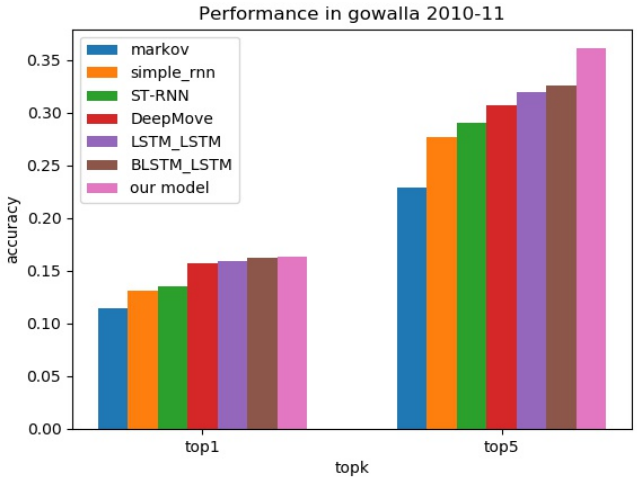


Fig. 7. Top@1 and Top@5 results on Gowalla dataset

Due to the similarity of two datasets, we mainly analyze the Foursquare dataset in this paper. As shown in **Fig. 6** and **Fig. 7**, we can observe that the traditional Markov model has the worst performance because it only considers the last location of the user, which results in low accuracy with little information on the transition matrix. Then, the general RNN model has a better performance compared with the Markov model, and it has a strong sequence model ability with its unique gate architecture. Subsequently, we evaluated two different variants of the RNN seq2seq framework on this dataset. The prediction accuracy is greater than 30% better than the simple RNN. The seq2seq framework has a better sequence model ability due to the LSTM with limited length. In addition, we can see that Bi-LSTM architecture helps us improve prediction accuracy compared with a general seq2seq framework. The Bi-LSTM combines the historical trajectory and future trajectory together to influence the present trajectory and generate the cell state. The information of the cell state contains more context of the historical trajectory, which helps improve the prediction accuracy. Finally, our model has the best performance compared with the above models.

These updated neural network methods (ST-RNN, SERM, and DeepMove) all exhibit higher accuracy than the simple RNN models and the Markov-based methods. SERM and ST-RNN cannot handle the sequential transition regularities well with a long trajectory, which leads to low accuracy of the Top@1 result. DeepMove has the best accuracy among the three methods due to its ability to exploit historical trajectories with attentional mechanisms. However, it cannot capture the spatial dependence of the historical trajectory. With the use of BLSTM, we can capture the long-term dependency of the historical trajectory. Then, with the use of the GCN, our model captures the topological architecture between locations. Compared with DeepMove, our model shows an increase of 7% and 15% in the Top@1 and Top@5 results, respectively.

We also calculate the perplexity of every model, as shown in **Fig. 8**. We introduced the perplexity above, and the low-perplexity probability model can better predict future locations. We can see that our model has the lowest perplexity from the results. The perplexity results can better prove the efficiency of our prediction model.

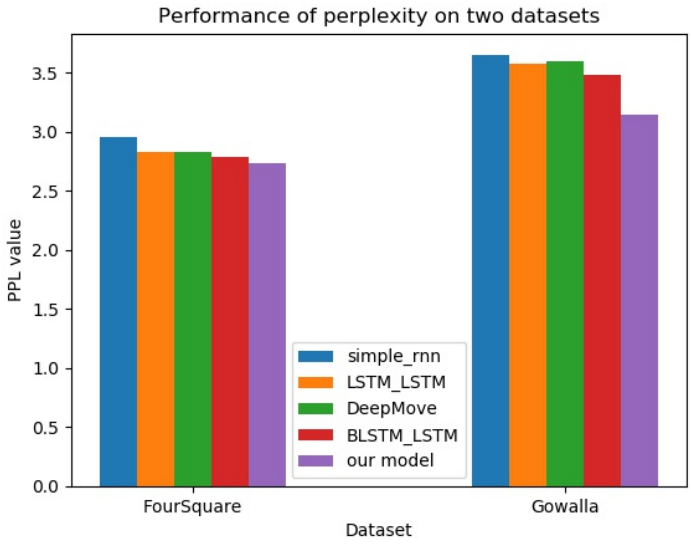


Fig. 8. PPL results on two datasets

Here, we note that the Top@1 result does not have large changes. However, the Top@5 result has better performance. We determined that this is because the GCN captured the spatial dependence of the historical trajectory, and our model deeply explored the correlation between the historical trajectory. Thus, our model has more precision on the prediction of candidate locations, and it has less influence on locations with the highest probability. In other words, our aggregating method with spatial and temporal dependence is effective, but the structural learning and temporal dependence are not learned well. In addition, we build our model based on the seq2seq framework, which increases the difficulty of training. Therefore, we obtained the same results on the Top@1 result compared with the seq2seq framework on the Top@1 result. Furthermore, we created a change chart of the loss curve and test accuracy of two datasets in **Fig. 9** and **Fig. 10**. From the two figures, we can see that our model has better performance. The BLSTM_LSTM and LSTM_LSTM in figures are variants of our model without GCN. It can be observed that the GCN architecture has a certain promotion of accuracy for the prediction task. In addition, our model quickly obtains the optimal value, which denotes that our model is feasible. However, the loss and accuracy curves tend to be smooth early, which means that our model cannot learn mobility patterns deeply in the following epoches. In other words, our model is hard to train. The reason for that includes our inefficient aggregating method, complex model architecture and more parameters. The adjustment of model structures, such as integrating GCN with LSTM, may be efficient.

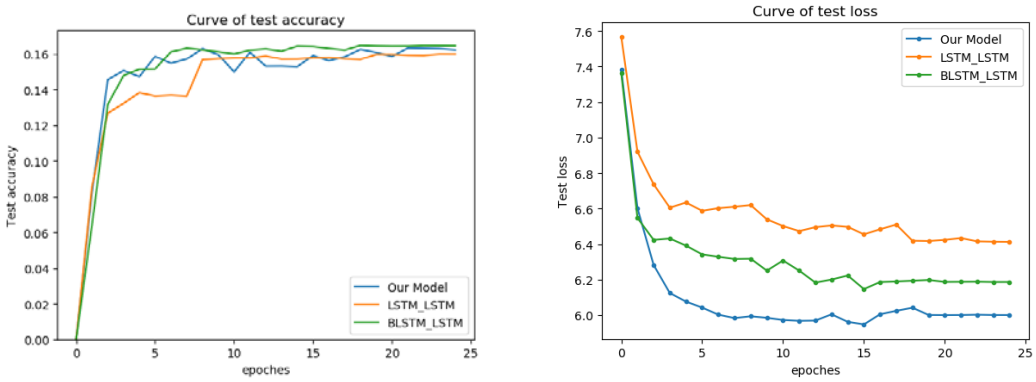


Fig. 9. The curves of test accuracy and test loss in Gowalla

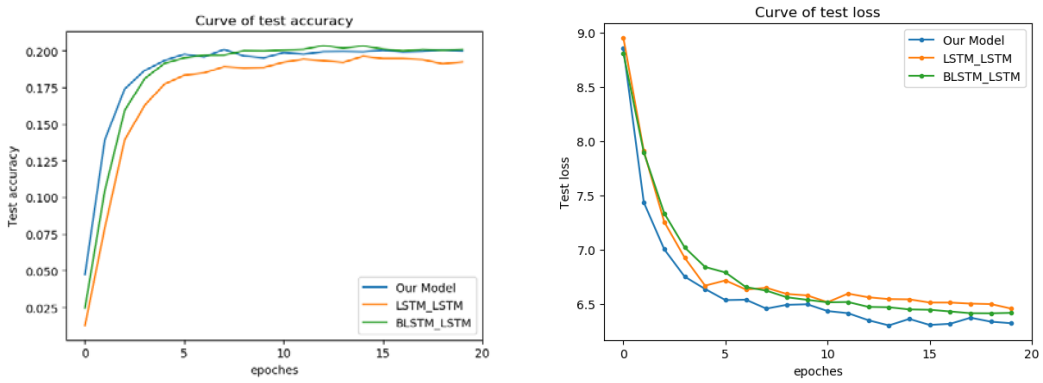


Fig. 10. The curves of test accuracy and test loss in Foursquare

5. Conclusion

In this paper, we proposed a neural network-based deep learning method to predict the next location that a person will go, which can capture the spatial dependence and time dynamics with GCN based on the seq2seq framework. We evaluated our model on two real-world datasets and then compared our model with the traditional model and updated deep learning methods. The experimental results indicate that our model has a better performance compared with other models. There are some disadvantages and future directions for our work. First, although our model has the best performance in this paper, the method of aggregating spatial-temporal dependence needs to be improved. Second, the low accuracy of our model on two datasets mainly ascribes to the sparsity of data. In the future, we will consider more features (such as the semantic context) to augment the richness of the data, and improve our model architecture to learn mobility patterns deeply. In this way, we can not only predict the next location where humans will go but also explore the deeper meaning of the human movement.

References

[1] Z. Xia, Z. Hu and J. Luo, "UPTP Vehicle Trajectory Prediction Based on User Preference Under Complexity Environment," *Wireless Personal Communications*, vol. 97 no. 3, pp. 4651-4665, 2017. Article (CrossRef Link)

[2] C. Feng, S. Arshad, S. Zhou, D. Cao and Y. Liu, "Wi-multi: A Three-phase System for Multiple Human Activity Recognition with Commercial WiFi Devices," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 7293-7304, 2019. Article (CrossRef Link)

[3] D. Ashbrook and T. Starner, "Learning significant locations and predicting user movement with GPS," in *Proc. of Sixth International Symposium on Wearable Computers*, pp. 101-108, October 7-10, 2002. Article (CrossRef Link)

[4] G. Yavaş, Dimitrios K., Özgür Ulusoy, Y.Manolopoulos, "A data mining approach for location prediction in mobile environments," *Data & Knowledge Engineering*, Vol. 54, no. 2, pp. 121-146, 2005. Article (CrossRef Link)

[5] A. Noulas, S. Scellato, N. Lathia and C. Mascolo, "Mining User Mobility Features for Next Place Prediction in Location-Based Services," in *Proc. of IEEE 12th International Conference on Data Mining*, pp. 1038-1043, December 10-13, 2012. Article (CrossRef Link)

[6] C. Song, Z. Qu, N. Blumm and A.-Laszlo Barabási, "Limits of Predictability in Human Mobility," *Science*, vol. 327, no. 5968, pp. 1018-1021, 2010. Article (CrossRef Link)

[7] C. Cheng, Y. Haiqin and L. Michael & K. Irwin, "Where you like to go next: Successive point-of-interest recommendation," in *Proc. of IJCAI International Joint Conference on Artificial Intelligence*. pp. 2605-2611, August 3-9, 2013. Article (CrossRef Link)

[8] S. Gambs, M.-O. Killijian, and M. N. n. del Prado Cortez, "Next place prediction using mobility markov chains," in *Proc. of the First Workshop on Measurement, Privacy, and Mobility*, ACM, pp. 1–6, 2012. Article (CrossRef Link)

[9] A. Al-Molegi, M. Jabreel, and B. Ghaleb, "Stf-rnn: Space time features based recurrent neural network for predicting people next location," in *Proc. of 2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1–7, December 6-9, 2016. Article (CrossRef Link)

[10] H. Yin and O. Wolfson, "A weight-based map matching method in moving objects databases," in *Proc. of 16th International Conference on Scientific and Statistical Database Management*, pp. 437–438, June 23, 2004. Article (CrossRef Link)

[11] H. Wang, Z. Li, and W. Lee, "Pgt: Measuring mobility relationship using personal, global and temporal factors," in *Proc. of 2014 IEEE International Conference on Data Mining*, pp. 570–579, December 14-17, 2014. Article (CrossRef Link)

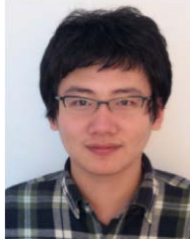
[12] F. Giannotti, M. Nanni, F. Pinelli, and D. Pedreschi, "Trajectory pattern mining," in *Proc. of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 330–339, 2007. Article (CrossRef Link)
[13] T. Kim, S. Taylor, Y. Yue, and I. Matthews, "A decision tree framework for spatiotemporal sequence prediction," in *Proc. of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 577-586, 2015. Article (CrossRef Link)
[14] W. Mathew, R. Raposo, and B. Martins, "Predicting future locations with hidden markov models," in *Proc. of the 2012 ACM Conference on Ubiquitous Computing*, pp. 911–918, September 5-8, 2012. Article (CrossRef Link)
[15] J. Ye, Z. Zhu, and H. Cheng, "What's Your Next Move: User Activity Prediction in Location-based Social Networks," in *Proc. of the 2013 SIAM International Conference on Data Mining*, pp. 171–179, May 2-4, 2013. Article (CrossRef Link)
[16] A. Gellert and L. Vintan, "Person movement prediction using hidden markov models," *Studies in Informatics and Control*, vol. 15, pp. 17–30, 2006. Article (CrossRef Link)
[17] Q. Liu, S. Wu, L. Wang, and T. Tan, "Predicting the next location: A recurrent model with spatial and temporal contexts," in *Proc. of the Thirtieth AAAI Conference on Artificial Intelligence*, pp. 194–200, February 12–17, 2016. Article (CrossRef Link)
[18] D. Yao, C. Zhang, J. Huang, and J. Bi, "Serm: A recurrent model for next location prediction in semantic trajectories," in *Proc. of the 2017 ACM on Conference on Information and Knowledge Management*, pp. 2411–2414, 2017. Article (CrossRef Link)
[19] N. Du, H. Dai, R. Trivedi, U. Upadhyay, M. Gomez-Rodriguez, and L. Song, "Recurrent marked temporal point processes: Embedding event history to vector," in *Proc. of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1555–1564, 2016. Article (CrossRef Link)
[20] A. Farzad and R. N. Asli, "Recognition and classification of human behavior in intelligent surveillance systems using hidden markov model," *IJ Image, Graphics and Signal Processing*, pp. 31-38, 2015. Article (CrossRef Link)
[21] A. Asahara, K. Maruyama, A. Sato, and K. Seto, "Pedestrian movement prediction based on mixed markov-chain model," in *Proc. of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 25–33, November 1-4, 2011. Article (CrossRef Link)
[22] Q. Li and H. C. Lau, "A Layered Hidden Markov Model for Predicting Human Trajectories in a Multi-floor Building," in *Proc. of 2015 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, pp. 344-351, December 6-9, 2015. Article (CrossRef Link)
[23] A. Monreale, F. Pinelli, R. Trasarti, and F. Giannotti, "Wherenext: A location predictor on trajectory pattern mining," in *Proc. of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 637–646, 2009. Article (CrossRef Link)
[24] S. Lee, J. Lim, J. Park and K. Kim, "Next Place Prediction Based on Spatiotemporal Pattern Mining of Mobile Device Logs," *Sensors (Basel)*, 16(2), 145, 2016. Article (CrossRef Link)
[25] A. Karatzoglou, S. C. Lamp and M. Beigl, "Matrix factorization on semantic trajectories for predicting future semantic locations," in *Proc. of 2017 IEEE 13th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pp. 1-7, October 9-11, 2017. Article (CrossRef Link)
[26] C. Cheng, H. Yang, I. King, et al, "Fused matrix factorization with geographical and social influence in location-based social networks," in *Proc. of Twenty-Sixth AAAI Conference on Artificial Intelligence*, July 22–26, 2012. Article (CrossRef Link)
[27] Z. Montazeri, A. Houmansadr and H. Pishro-Nik, "Achieving Perfect Location Privacy in Wireless Devices Using Anonymization," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 11, pp. 2683-2698, 2017. Article (CrossRef Link)
[28] T. Mikolov, M. Karafiát, L. Burget, J. H. Cernocký, and S. Khudanpur, "Recurrent neural network-based language model," *INTERSPEECH*, pp. 1045-1048, 2010.

[29] B. Wang, W. Kong, H. Guan, et al, "Air Quality Forcasting based on Gated Recurrent Long Short-Term Memory Model in Internet of Things," *IEEE Access*, pp. 69524-69534, 2019. Article (CrossRef Link)
[30] M. Wang, S. Niu and Z. Gao, "A Novel Scene Text Recognition Method Based on Deep Learning," *Computers, Materials & Continua*, Vol. 60, No. 2, pp.781-794, 2019. Article (CrossRef Link)
[31] A. Zarezade, S. Jafarzadeh, and H. Rabiee, "Recurrent Spatio-temporal modeling of check-ins in location-based social networks," *PLOS ONE*, vol. 13, 11 2016. Article (CrossRef Link)
[32] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. of the 27th International Conference on Neural Information Processing Systems*, pp. 3104–3112, 2014. Article (CrossRef Link)
[33] S. Park, B. Kim, C. Mook Kang, C. Choo Chung, et al, "Sequence-to-sequence prediction of vehicle trajectory via LSTM encoder-decoder architecture," in *Proc. of IEEE Intelligent Vehicles Symposium (IV)*, pp. 1672-1678, 2018. Article (CrossRef Link)
[34] D. D. Nguyen, C. Le Van, and M. I. Ali, "Vessel trajectory prediction using sequence-to-sequence models over spatial grid," in *Proc. of the 12th ACM International Conference on Distributed and Event-based Systems*, pp. 258–261, June 25-29, 2018. Article (CrossRef Link)
[35] J. Feng, Y. Li, C. Zhang, et al, "Deepmove: Predicting human mobility with attentional recurrent networks," in *Proc. of the 2018 world wide web conference*, pp. 1459–1468, April 23-27, 2018. Article (CrossRef Link)
[36] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016. Article (CrossRef Link)
[37] L. Zhao, Y. Song, M. Deng, and H. Li, "Temporal graph convolutional network for urban traffic flow prediction method," *IEEE Transactions on Intelligent Transportation Systems*, 2018. Article (CrossRef Link)
[38] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," *arXiv preprint arXiv:1707.01926*, 2017. Article (CrossRef Link)
[39] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129–150, 2011. Article (CrossRef Link)
[40] M. Schlichtkrull, T N. Kipf, P. Bloem, et al, "Modeling relational data with graph convolutional networks," in *Proc. of European Semantic Web Conference*. pp. 593-607, June 3-7, 2018. Article (CrossRef Link)
[41] C. Li, Y. Jiang and M. Cheslyar, "Embedding Image Through Generated Intermediate Medium Using Deep Convolutional Generative Adversarial Network," *Computers, Materials & Continua*, Vol.56, No.2, pp.313-324,2018. Article (CrossRef Link)
[42] H. K. Shin, Y. H. Ahn, S. H. Lee and H. Y. Kim, "Digital Vision Based Concrete Compressive Strength Evaluating Model Using Deep ConVolutional Neural Network," *Computers, Materials & Continua*, Vol. 61, No. 3, pp.911-928, 2019. Article (CrossRef Link)
[43] K. Xu, L. Wu, Z. Wang, Y. Feng, M. Witbrock, and V. Sheinin, "Graph2seq: Graph to sequence learning with attention-based neural networks," *arXiv preprint arXiv:1804.00823*, 2018. Article (CrossRef Link)
[44] D. Bahdanau, K. Cho and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," *arXiv preprint arXiv:1409.0473*, 2014. Article (CrossRef Link)
[45] D. Yang, D. Zhang, V. W. Zheng, and Z. Yu, "Modeling user activity preference by leveraging user spatial temporal characteristics in lbsns," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, no. 1, pp. 129–142, 2014. Article (CrossRef Link)
[46] E. Cho, S. A. Myers, and J. Leskovec, "Friendship and mobility: user movement in location-based social networks," in *Proc. of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. pp. 1082–1090, 2011. Article (CrossRef Link)

Jianwei Chen: is currently pursuing the master degree at the School of Computer Science and Technology, Qingdao University. He received the bachelor degree in Digital media technology from the Qingdao University, China in 2017. His research interest is urban travel forecast and mobile social networks.

Jianbo Li: is currently a professor at the Computer Science and Technology College of Qingdao University. He received the bachelor and master degrees in computer science from Qingdao University, China in 2002 and 2005, respectively, and the PhD degree in computer science from the University of Science and Technology of China in 2009. His research interests include urban computing, mobile social networks, and data offloading.

Manzoor Ahmed: received the B.E. and M.S. degrees in electrical engineering and computer science from Balochistan Engineering University, Khuzdar, Pakistan, in 1996 and 2010, respectively, and the Ph.D. degree in communication and information systems from the Beijing University of Posts and Telecommunications, China, in 2015. From 1997 to 2000, he was a Lecturer with Balochistan Engineering University and a Telecomm Engineer in government-owned telecommunication service provider NTC, Pakistan, from 2000 to 2011. He was a Postdoctoral Researcher from the Electrical Engineering Department, Tsinghua University, China, from 2015 to 2018. He is currently a Faculty Member with the Department of Computer Science and Technology, Qingdao University. His research interests include resource allocation and offloading in vehicular communications and networking, fog and edge computing, socially aware D2D communication, and physical layer security. He has several research publications in IEEE top journals and conferences. He received several awards, including the Distinction Award from the President of Pakistan, the Best Employee Award from NTC, and the Best Paper Award from the 2014 GameNets Conference.



Junjie Pang: received the M.S. and Ph.D. degrees in computer science from Jilin University in 2013 and 2017, respectively. He currently holds a post-doctoral position at Qingdao University. His research interests include traffic engineering in data center networking, cloud computing, and SDN.

Minchao Lu: is currently pursuing the master degree at the School of Computer Science and Technology, Qingdao University. He received the bachelor degree in Network engineering from the Liaocheng University, China in 2017. His research interest is urban travel forecast and mobile social networks.

Xiufang Sun: is currently pursuing the master degree at the School of Computer Science and Technology, Qingdao University. She received the bachelor degree in Digital media technology from the Qingdao University, China in 2017. Her research interest is urban traffic forecast and mobile social networks.