Information Sciences 629 (2023) 1–14

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

GNN-based long and short term preference modeling for next-location prediction

Jinbo Liu [a], Yunliang Chen [a,*], Xiaohui Huang [a], Jianxin Li [b], Geyong Min [c]

[a] School of Computer Science, China University of Geosciences, Wuhan, 430078, China
[b] School of Information Technology, Deakin University, Geelong, VIC3220, Australia
[c] Department of Computer Science, College of Engineering, Mathematics and Physical Sciences, University of Exeter, Exeter, EX4 4QF, UK

ARTICLE INFO

Keywords:
Location prediction
Spatial-temporal dependence
User preference
Graph convolution network

ABSTRACT

Next-location prediction is a special task of the next POIs recommendation. Different from general recommendation tasks, next-location prediction is highly context-dependent: (1) sequential dependency, i.e., the sequential locations checked in by a user have high correlation; (2) temporal dependency, i.e. check-in preferences are identified as different days or nights; (3) spatial dependency, i.e. users prefer to visit closer locations. Recent studies have been very successful in predicting users' next location by comprehensively considering user preferences. Nonetheless, these methods not only fail to capture temporal dependencies but also fail to capture location topology information. To fill this gap, we propose a GNN-based model which converts POIs into a low-dimensional metric and integrates users' long-term and short-term preferences to comprehensively represent dynamic preferences. The model consists of graph neural networks for long-term preference modeling and LSTM for short-term preference modeling. Comprehensive experiments are conducted on two real-world datasets, and results demonstrate the effectiveness of our approach over state-of-the-art methods for next-location prediction.

1. Introduction

With the development of smart mobile devices and social networks, it is convenient for people to check in at physical location and reveal their positions with others. Abundant check-in data makes it possible to predict human movement and explore user preferences for POI (Point of Interest) recommendations. In this background, the human mobility prediction is proposed to recommend the next location that people might visit according to a check-in history. Location prediction can provide a strong support for subsequent multiple types of intelligent location services, such as crowd flow prediction for risk control, user behavior intention prediction for the business recommendation, etc. Therefore, accurate and effective location prediction technology has significant application value.

Different from traditional recommendation tasks, the interactions between users and POIs are spare due to privacy protection [32, 6]. Therefore, location prediction becomes more challenging due to the sparseness of user activity. Three essential properties of user check-in should be considered to improve the prediction performance. First, numerous studies have shown that people are more likely to visit points close to their geographic space or points they have already visited. Spatial dependency is a significant factor for user next location recommendation. Second, User behavior shows strong time-cycle patterns in weeks or days. For example, users

* Corresponding author.
E-mail addresses: jinbo.liu@cug.edu.cn (J. Liu), chenyunliang@cug.edu.cn (Y. Chen), xhhuang@cug.edu.cn (X. Huang), jianxin.li@deakin.edu.au (J. Li), g.min@exeter.ac.uk (G. Min).

https://doi.org/10.1016/j.ins.2023.01.131
Received 13 July 2022; Received in revised form 19 November 2022; Accepted 29 January 2023

Available online 1 February 2023
0020-0255/© 2023 Elsevier Inc. All rights reserved.

typically visit POIs around the workplace on weekdays, while they prefer to visit some POIs near home on weekends. Temporal dependency is also an indispensable factor in predicting the user's next location. Finally, the transition probabilities of users from one check-in location to other POIs are non-uniformly distributed due to personal preference. Therefore, we aim to design a location prediction framework by jointly considering temporal-spatial dependency and sequential influence.

Human mobility prediction has been extensively studied. In the early studies, Matrix Factorization and Markov Chains become the mainstream method based on sequential prediction approaches. Markov chain models such as the one proposed by Rendle et al. [20] often fail to obtain the desired results because they consider only a small amount of features and ignore the complex behavioral patterns. These Markov-based models cannot capture the temporal dependence, which leads to poor performance. In recent years, deep learning technology has shown extraordinary dominance in various research fields. Recurrent neural networks and their variants have become a common solution for sequential data analysis tasks [7,24]. For instance, LSTPM [22] is proposed to design a special network for long-term user preferences and a variant of LSTM (Long Short-Term Memory) for short-term user preferences respectively.

Despite existing researches have achieved some encouraging results, there are still some critical issues that need to be focused on. Some researchers have only considered the temporal dependence, ignoring the spatial dependence of consecutive locations in check-in activities. Recently, GCN (Graph Convolutional Networks) becomes popular because of its successful application in many fields [14,21,39]. GCN is a deep learning method that draws on the ideas of the convolutional network, recurrent network, and deep autoencoder. Its powerful processing capability for graph-structured data makes it rapidly emerging in the fields of computer vision, recommendation systems, traffic flow prediction, chemical analysis, etc.

In computer vision, people have explored methods of using graph-structured data in multiple tasks such as point cloud classification and segmentation, scene graph generation, and action recognition [17,23]; in recommender systems, it is obvious that recommendation task can be transformed into link prediction task [27,36]. The prediction goal is converted to predict the lost connections between users and items; in traffic flow prediction, people use graph-based spatiotemporal neural network methods to accurately predict data such as traffic speed and traffic volume in the traffic network, effectively helping the intelligent transportation system save resource [11]. Regarding the location prediction task, many non-Euclidean graph data in location social networks bring new possibilities for introducing graph neural networks into the field of human mobility research [37].

Thus, a deep learning method should be proposed to predict the next location considering the topology between locations. Furthermore, embedding techniques are used to solve the problem of sparsity. The user's historical trajectory and current trajectory should be considered separately, mining the user's long-term life patterns from the historical trajectory and mining the user's short-term behavioral preferences from the current trajectory. The long-term preference and short-term preference are aggregated to perform the prediction task. In summary, the contributions of this article are summarized as follows:

- A new unified model GLSP (GNN-based long and short term preference model) is proposed to predict next location. Moreover, a linear combination unit is introduced to obtain the different weight of the long-term and short-term modules. A method for calculating the geographically reachable probability of the candidate locations is designed to evaluate the influence of geographic factors.
- Inspired by the application of graph neural networks in recommender systems, a long-term and short-term preference fusion mechanism is proposed. Specifically, we use the check-in information in historical trajectories to construct location graphs, use graph neural networks to obtain corresponding location embeddings, and then obtain graph embeddings to represent the spatial dependencies in users' historical trajectories. We process the history trajectory by LSTM to obtain the temporal dependencies in the historical trajectory. Finally, the spatial and temporal dependencies are combined to learn the long-term preference representation. For convenience, LSTM is used to model short-term user preferences.
- We conduct extensive experiments on two real-world datasets to validate the GLSP and illustrate that the new approach outperforms prior work.

The remainder of this article is organized as follows. Section 2 introduces the related work and gives an overview of the state-of-the-art. Section 3 proposes the GLSP algorithm for the next location prediction. Section 4 illustrate the detailed experimental evaluation. And Section 5 concludes this work and provides the future direction.

2. Literature review

This section summarizes the closely related works from two aspects: pattern-based methods and model-based methods. In addition, the development of graph neural networks is also worthy of attention, so we summarize its development in the field of recommendation.

2.1. Next-POI recommendation

Pattern-based approaches can only obtain explicit predefined patterns on existing trajectories, but cannot capture comprehensive movement patterns about the user's movement. Therefore, pattern-based approaches are less suitable for LBSN scenarios where trajectory data is extremely sparse. In contrast, model-based methods have excelled in the field of location prediction due to their ability to capture more complex motion laws and aggregate diverse information. Therefore, we mainly introduce the difference between the two methods.

2.1.1. Markov chain-based methods

Markov-based models mainly use the transfer matrix to predict the probability of the next behavior. Markov chain-based temporal recommendation model is representative of the traditional trajectory sequence prediction algorithm. Gambs [5] models the likelihood of future moves by recording the transfer of historical trajectories between multiple locations and building a transfer matrix. Ishikawa et al. [9] use a Spatio-temporal database to store the historical trajectories of mobile objects and divide the map area into grids of different sizes. Markov chains are used to describe the probability of moving objects in the map grid and R-Tree is used as a tutor to predict the next location of the objects based on the grid location of the historical trajectories of mobile objects. The trajectory prediction algorithm of the implicit Markov model proposed by Qiao et al. [18] also divides the space in the form of a grid, so that the location of the trajectory points appearing in the grid corresponds to the observed states in the implicit Markov model. And the prediction model is established by the results of segmentation and clustering of the trajectory sequence corresponds to the implicit states in the model. However, these Markov-based methods can only perform well in certain scenarios as they cannot capture the temporal dependencies.

Markov models can capture the regularity between state transitions and predict the next state based on one or more states before the current state. By increasing the Markov model's order, the prediction effect will gradually improve. At the same time, due to the excessive sparseness of the transition matrix, the requirements for data storage increase sharply, resulting in a sharp drop in model efficiency. Markov models do not pay enough attention to historical information, have limited predictive ability, and often fail to obtain ideal results.

2.1.2. Neural network-based methods

With deep learning developing, RNN has received increasingly attention due to the ability to model sequence data [1]. It has been highly successful in many research efforts, including the tasks such as natural language processing, temporal prediction, etc. To date, there are also many studies that have successfully predicted human mobility using recurrent neural networks. Since the temporal and spatial intervals between adjacent locations can be captured by the RNN, ST-RNN [15] achieves predictions by linearly interpolating and learning specific transformation matrices for time and distance. ST-LSTM [10] is a variant model that observes that time intervals and spatial intervals can be combined based on linear interpolation, and inputs the combined input to the multiplication gate of the LSTM for prediction. Time-LSTM [38] changes the LSTM structure and adds a time gate to improve the prediction accuracy. STGN [35] significantly improves prediction performance by adding spatio-temporal gates to the LSTM. ATST-LSTM [8] introduces an attention mechanism to help assign different weights to each check-in location, but it is limited to only consider locations that users visit consecutively. DeepMove [4] combines the attention layer that learns the long-term periodicity with the recurrent neural network layer that learns the short-term sequence transition law to improve the prediction effect. LSTPM [22] is an optimal model that comprehensively considers the user's preference and captures the user's behavioral intention through the historical trajectory and current trajectory. LSTPM also improves the RNN structure and introduces geo-dilated RNN to aggregate user short-term preferences.

Therefore, RNN and its various variant models are the basis of neural network-based methods. Although these models use contextual features such as time and space to train RNN modules and effectively model the changing laws of mobile user preferences, there are still some problems to be solved. On the one hand, the historical trajectories of users are usually very long, and the entire trajectory information is difficult to be effectively utilized by existing methods. On the other hand, RNN-based methods tend to ignore the spatial dependencies between user check-in points. Time dependency focuses on the order of check-in points in the trajectory, while spatial dependency focuses on the distribution characteristics of check-in in the trajectory in the spatial dimension. Compared with the above models, we use LSTM to capture temporal and sequential dependencies and graph neural networks are also used to capture spatial correlations between locations in historical trajectories.

2.2. GNN-based recommendation

Due to the successful application of GNN in the field of representation learning, there has been a recent increase in research related to GNN-based recommendation models. Several researchers have introduced graph neural networks into session sequence-based recommendations. SRGNN [28] converts the sequence into a directed graph without weights and uses the edges of graph to express the transition relationships of sequence, and finally uses gated graph neural networks to propagate information between nodes connected by edges. Xu et al. [31] uses GGNN to extract local information based on SRGNN and captures global dependencies before remote items using self-attentive networks. With the successful application of random walk algorithm in the field of node representation, PinSage [33] tries to use random walk algorithm to generate node representation on the graph of recommendation field to complete the prediction. NGCF [25] uses a combination of higher-order graph neural networks and collaborative filtering to jointly learn node representations in the graph.

Most of the existing studies use GNN to capture the transition patterns from user sequential behavior sequence graphs [12]. For example, A-PGNN [29] enriches the representation of connections between nodes in a sequence by combining the user's historical sequence with the current sequence. GCE-GNN [26] uses local context and global context to help predict the transition pattern in the current sequence. MA-GNN [16] changes the composition in a way that assumes that one check-in has an effect on all subsequent multiple check-ins, so it considers the influence of non-adjacent nodes by treating four check-in points as a group and adding edges between them. Fi-GNN [13] constructs a feature graph and captures higher-order interactions between features using GNN. All the above studies provide a reference for graph-based sequence prediction.

Summarizing the existing research, we can find that the research on human mobility in the current location social network [2,3,19] ignores the rich contextual information in the location social network when modeling user preferences, and the user preferences

Table 1
Key mathematical notation.

Notation	Description
u, U	A user and the user collection
l, L	A location and the location collection
T_u^n, T_u^N	The n-th historical trajectory of u; the current trajectory of u
e_l	The embedding of location l
G_u, A_u	The location graph; the adjacency matrix of the location graph
P_L, P_S	The long-term preferences; the short-term preferences

contained in the user check-in trajectories are not comprehensive enough, resulting in low prediction accuracy. Therefore, introducing a graph neural network to model the historical trajectory to capture the spatial dependencies not considered by most models, while separately modeling the long-term and short-term preferences to obtain a more accurate preference representation, is still an issue that needs further research.

3. Proposed method

This section is as follows: Subsection 3.1 first defines several key concepts and presents the problem definition. Subsection 3.2 gives a solution to the problem and provides an overall overview of model. Then, Subsection 3.3 describes the important modules of our model in detail.

3.1. Problem definition

To make the presentation clear, we firstly define the data concepts and next location prediction task. Table 1 summarizes the notations related to this article.

Definition 1 *(POI)*. A POI is defined as the unique identifier of the user's check-in location, which contains two attributes: the identifier ID and the latitude and longitude coordinates.

Definition 2 *(Check-in)*. $<u, l, t>$ describes that a user u visiting POI l at time t.

Definition 3 *(Check-in Sequence)*. A check-in sequence is depicted as check-in collection. It is defined as $T^i = \{\langle l_1, t_1\rangle, ..., \langle l_n, t_n\rangle\}$, the times in the sequence that belong to the same day. For convenience, the sequence of check-ins for a day is represented as $T^i = \{l_1, l_2, l_3..., l_n\}$.

Definition 4 *(Historical Trajectories Set S)*. The historical trajectory is a collection of many check-in sequences that have occurred. We denote $U = \{u_1, u_2, ..., u_n\}$ to be a set of LBSN users. The historical trajectory of the i-th user is $T_{u_i} = \left(T_{u_i}^1, T_{u_i}^2, ..., T_{u_i}^{t-1}\right)$, where $T_{u_i}^j = \left(l_{u_i}^{t_1}, l_{u_i}^{t_2}, ...l_{u_i}^{t_T}\right)$ and $l_{u_i}^{t_{th}}$ indicates the location where u_i went at time t_{th}. Hence, after processing the entire dataset, we can describe all user's historical trajectories $T = \left(T_{u_1}, T_{u_2,...,T_{u_n}}\right)$.

Goal (Location Prediction). Given one user's full historical trajectory $T_{u_i} = \left(T_{u_i}^1, T_{u_i}^2, ..., T_{u_i}^{t-1}\right)$ and current trajectories $T_{u_i}^t = \left(l_{u_i}^{t_1}, l_{u_i}^{t_2}, ..., l_{u_i}^{t_{T-1}}\right)$, the location prediction problem is to learn a model to predict u_i's next location $l_{u_i}^{t_T}$. The purpose of the task is to get a list of the most likely locations that the user will visit at the next moment to complete the prediction of the user's behavioral intent. The higher the location in the list, the more likely the user will visit that location.

3.2. Framework overview

This section mainly outlines the solution for the user's next location prediction. Our core idea is to learn the comprehensive behavioral intentions by aggregating user preference. And a list of the next locations to be visited by users will be generated. As shown in Fig. 1, we first present the framework overview of GLSP. The GLSP model consists of three components: the long-term preference modeling, the short-term preference modeling, and the prediction module. Our main contributions lie in all three modules, where we model the long-term preferences using Graph Neural Network and model the short-term using LSTM. We also propose a module to obtain the personalized weights of different user preferences.

3.3. GNN-based long and short preference model

This section first introduces how to obtain the user's long-term habit representation and short-term intention representation. Then the process of aggregating the user's overall interests to complete the prediction of the user's next location will be described. Finally, we introduce a module used for calculating geographically reachable probabilities to correct the results.

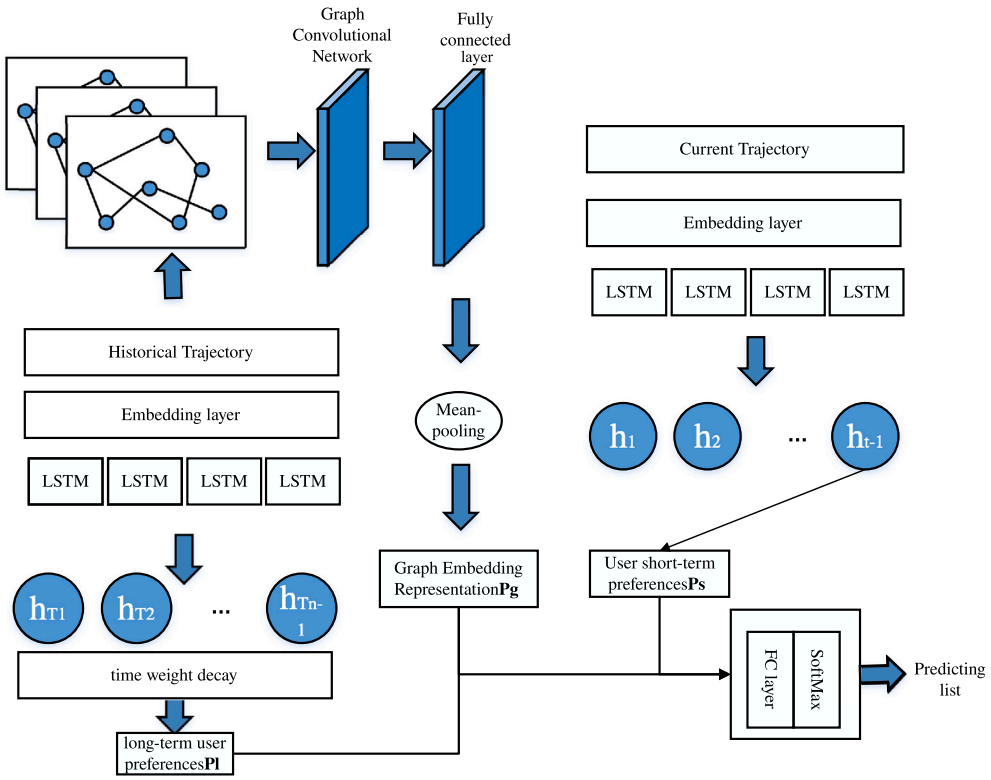

Fig. 1. The architecture of our GLSP model.

3.3.1. *Long-term module*

This section mainly introduces our proposed method to learn users' long-term preferences based on user historical trajectories. Although users' long-term sequences reflect their overall preferences to some extent, many potential patterns between check-in locations are not considered, so we learn long-term preferences from users' location maps and historical trajectories together.

First, we consider only one feature, the location id, and put its one-hot encoding into the embedding layer to get its dense representation. After representing all locations in the same vector space, graph neural networks will be introduced to learn the vector representation of each location $v \in \mathbb{R}^d$. Then we illustrate how the location graph is generated as shown in Algorithm 1, from which we obtain the adjacency matrix and feature matrix of the user's location graph. Finally, we input the location graph into the graph convolution layer and the fully connected network layer successively to generate the node embedding in the graph and then obtain the graph embedding by mean-pooling operation. In this process, GCN is used to obtain the spatial correlation between check-in locations in the historical trajectory. GCN first constructs a filter Z, and then applies it to the nodes of the location graph and their first-order neighbors to learn the spatial connections between nodes and obtain the embedding representation of the nodes.

Algorithm 1 Location graph construction method.

Input:
Historical trajectory set , $T = (T_{u_1}, T_{u_2}, ..., T_{u_n})$;
Output:
Location graph set, G;
1: Load datasets;
2: For u in U do;
3: For T in T_u^n;
4: $G_u = (V_u, E_u)$, V_u is location collection and E_u is edge collection;
5: $V_u = set\left(T_u^1 + T_u^2 + ... + T_u^{t-1}\right)$;
6: Initialize $A_u \in \mathbb{R}^{|V_u| * |V_u|}$;
7: If node i and node j are connected;
8: $A_{ij} = 1$ else $A_{ij} = 0$;
9: There are only two elements 1, 0 in the adjacency matrix;
10: End for;
11: End for;
12: **return** $G = \{G_{u_1}, G_{u_2}, ..., G_{u_n}\}$;

Algorithm 2 Framework of GCN for Location graph.

Input:
User collection, U; location collection, L; user's check-in sequence collection, T;
Output:
The user's spatial dependency representation, e_g;
Construct the location graph by sampling;
Generate location embedding representation;
Calculate the adjacency matrix $A_u \in \mathbb{R}^{n\times n}$, Node feature matrix $X_u \in \mathbb{R}^{n\times d}$ of user u;
For i =1 to U do (take one user as example);
$H^{l+1} = \sigma\left(D^{-\frac{1}{2}} A D^{-\frac{1}{2}} X W\right)$;
Average pooling;
End for;
return e_g;

An example of how to get the spatial feature representation from the location graph is as follows. The filter Z is defined as:

$$Z\left(X_u, A_u\right) = \sigma\left(D_u^{-1}\widetilde{A_u}X_u W\right) \tag{1}$$

where $\widetilde{A_u} = A_u + I$ denotes a self-connected structure, A_u is the adjacency matrix of the location graph, and D_u is the out-degree diagonal matrix of A_u and W is the weight matrix of the layer. σ is the activation function, such as ReLU and Tanh. With the above equation, we can obtain the embedding representation of all nodes in the location graph. Next, the spatial feature matrix concatenated by all nodes in the graph will be input into a fully connected network and average pooled to obtain the final graph embedding representation. The graph embedding of the location graph is shown in Algorithm 2.

Next, since we divide the user trajectory by day, we use LSTM to obtain the preference representation on that day. First, LSTM is used to capture the sequential relationship in the trajectory. Specifically, the encoding process for each trajectory is as follows:

$$h_t = LSTM\left(x_t, h_{t-1}\right) \qquad t \in \left\{1, 2, ..., \left|T_u\right|\right\} \tag{2}$$

where h_t is the hidden state of LSTM and $x_t \in \mathbb{R}^d$ is the randomly initialized d-dimensional embedding vector for the t-th check-ins. Through the encoding process, the last hidden state of all trajectories is encoded as $\left\{h_{|T_1|}, h_{|T_2|}, ..., h_{|T_{n-1}|}\right\}$. Considering the time-sensitivity of user preferences, we use exponential decay to quantify the influence weight of user history trajectory on the user's next visit.

$$P_{L1} = \sum e^{-\left(t_n - t_i\right)} \cdot h_{|T_i|} \tag{3}$$

Finally, we synthesize the graph embedding of the historical trajectory and calculate the long-term preference by the time-decay weighted summation to obtain the final user long-term behavioral intention. For convenience, the final long-term preferences are expressed as follows:

$$P_L = \frac{e_g + P_{L1}}{2} \tag{4}$$

3.3.2. Short-term module

Inspired by LSTM for time series prediction, we exploit LSTM to learn users' short-term preferences. LSTM solves the gradient disappearance and gradient explosion problems of general recurrent neural networks by constructing a gated mechanism. Like the LSTM in the long-term preference module, the check-in in the current trajectory will be encoded as a series of hidden states $\left\{h_1, h_2, ..., h_t\right\}$. Considering that all check-ins in the current trajectory may represent the user's behavioral intention, Short-term user preferences are computed and expressed as follows:

$$P_S = h_{t-1} = LSTM\left(x_t, h_{t-2}\right) \tag{5}$$

3.3.3. Prediction framework

With the above steps, we can obtain the long-term behavior representation and short-term intention representation of the users and the embedding representation of the locations. It is well known that users' long-term preferences and temporary intentions have different effects on their next behavior. Inspired by research work [30], a linear combination unit will be introduced to learn the preference weights for different users. In short, we design a module to learn the personalization weights of long-term preference modules and short-term preference modules for different users.

$$P = \alpha P_S + (1 - \alpha) P_L \tag{6}$$

The preference vectors are inputed into the fully connected layer to obtain the preference scores of users visiting each location, and the fully connected neural network is computed as follows:

$$y = Relu(W \cdot P + b) \tag{7}$$

where vector y denotes the preference score of users visiting each location; the length of y is the total number of selectable locations; W and b denote the parameter matrix and bias of the fully connected neural network, respectively, and Relu is the activation function.

Finally, y is mapped to the probability on the interval [0,1] by the softmax function, and the top-k predictions are obtained in term of the probability ranking. The softmax function is calculated as follows.

$$y = softmax\left(y_i\right) = \frac{exp\left(y_i\right)}{\sum_{j=1}^{N} exp\left(y_j\right)} \qquad i \in \{1, 2, ..., N\} \tag{8}$$

where N is the total number of points of interest and the result y represents the probability of a user visiting a location with node identifier i.

To further consider the geographic sensitivity of the target user to the candidate location. The probability of geographic accessibility of the target user to the candidate location can be calculated. It is summed with the obtained cosine similarity as the access probability of user u against location v. Finally, the top-k locations are selected based on the probability magnitude to generate a prediction list.

$$p\left(l|T_u^N\right) = \frac{\sum \phi\left(\frac{dist(l_i,l)}{\sigma}\right)}{\sigma\left|T_u^N\right|} \tag{9}$$

where $dist(x, y)$ denotes the geographical distance between any two points, ϕ is the standard Gaussian distribution function. σ is the kernel density optimal bandwidth parameter and $\sigma \approx 1.06\widetilde{\sigma}\left|T_u^N\right|^{\frac{1}{5}}$ according to the literature [34]. This formula takes into account the influence of geographic location, so that candidate locations that are close from the user check-in center have a higher probability of being visited, while candidate locations that are far from the user check-in center have a lower probability of being visited.

In summary, we have given our solution to the user's next location prediction task. And cross-entropy loss will be used to define the loss function of the model as follows:

$$L_c = \frac{1}{train} \sum_{I=1}^{|U|} \sum_{j=1}^{|L|} y_{ij} \cdot log\left(O_{ij}\right) + \lambda \left\|\Theta\right\|_2 \tag{10}$$

where the cross-entropy loss is denoted as L_c. And $y_{ij} = 1$ when u_i visited l_j otherwise it is 0; $\|\Theta\|_2$ is the regularization term to avoid over-fitting. λ is the regularization parameter and θ indicates all the parameters.

The gradient descent optimization algorithm Adam is used to optimize the model so that the objective function is minimized. And Algorithm 3 describes the learning and training process of the model.

Algorithm 3 User next location prediction algorithm.

Input:
The historical trajectory, T; the current trajectory, T^N;
Output:
Trained Model;
1: Constructing the training set;
2: Initialize the parameters Θ;
3: Repeat;
4: For each user do;
5: Construct location graph, extract graph embedding;
6: Compute long-term preference P_L according to equations (1) (4);
7: Compute short-term preference P_S according to equations (5);
8: Compute the output y according to (6) (8);
9: Obtain the final output p in term of (9);
10: Update with gradient descent in term of (10);
11: Until the model converges;
12: **return** trained model;

4. Comparative experiment

Experiments on two real-world datasets are performed to verify the effectiveness of the GLSP model in location prediction. We intend to seek answers to the several questions: 1) How does our approach perform in comparison with baselines and other models? 2) how each component affects the model performance?

4.1. Dataset description

Foursquare NYC and Foursquare TKY are the two datasets used to train and evaluate the performance of the model. These two datasets have a graph structure where each user's location is considered a vertex, and the connection between two locations is represented with an edge. The statistical results of the dataset are described in Table 2.

Table 2
Statistics of datasets.

Statistical item	Foursquare NYC	Foursquare TKY
Users	741	2032
Locations	12120	22218
check-ins	58361	288847
Average check-ins	78.8	142.1
Density	0.64%	0.63%

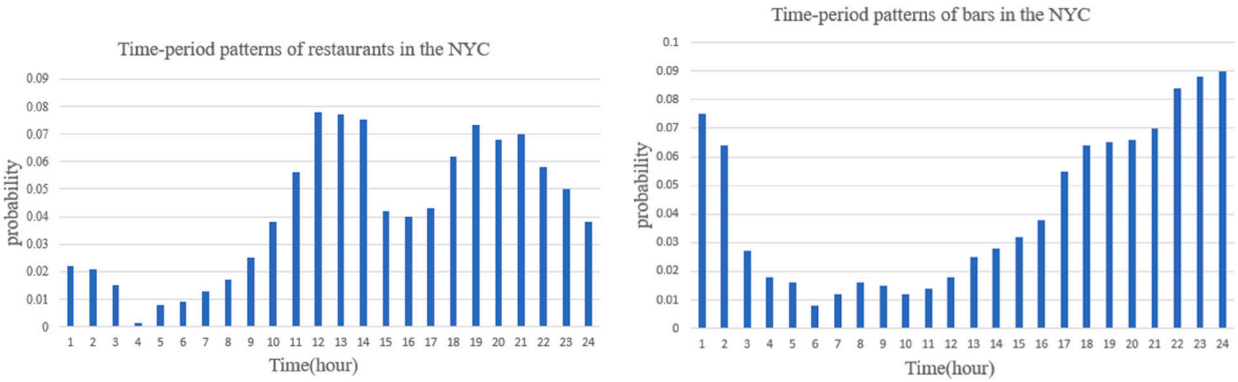

Fig. 2. NYC dataset check-in pattern example graph.

(1) Foursquare NYC: This dataset data is a collection of check-in data from April 2012 to February 2013 in New York City, US. For this dataset, we preprocess to ensure that each user has at least 10 check-in records and each valid location is visited by at least 10 different users.
(2) Foursquare TKY: The data in this dataset is also a collection of check-in data from April 2012 to February 2013 in Tokyo, Japan. We process this dataset in the same way as NYC.

Take a dataset as an example, we perform statistics on the dataset to represent the periodicity and geographic aggregation of user check-in behavior. Fig. 2 shows the check-in pattern statistics for restaurants and bars in the NYC dataset. It can be concluded that the peak periods for restaurants to be visited are from 11:00 to 14:00 and from 18:00 to 22:00 and the peak periods for bars to be visited are from 18:00 to 2:00 the next day. It can be seen that restaurants have a higher probability of being visited during the day and bars have a higher probability of being visited at night. Then we count the distribution of distances between adjacent check-in point in the Foursquare dataset and show it in Fig. 3. It can be found that about 90% of consecutive check-ins in the NYC dataset are within 10 kilometers of each other, while this proportion reaches 91.3% in the TKY dataset.

4.2. Evaluation metric

In this article, we choose ACC@k (k=1,5,10) and Mean Average Precision (MAP) as the evaluation metrics, which are generally used to evaluate a location prediction task. ACC@k indicates whether the prediction list of top-k contains the ground truth location and MAP evaluates the quality of the prediction list. Specifically, the average value of these two metrics' overall users is calculated for evaluation. It is obvious that two larger evaluation parameter values represent better prediction performance results. The Precision@k and MAP are defined as follows:

$$ACC@k = \frac{1}{U}\sum_{u\in U}\frac{\left|S^u_{predicted} \cap S^u_{visited}\right|}{\left|S^u_{visited}\right|} \tag{11}$$

$$precision = \frac{TP}{FP+TP} \tag{12}$$

$$MAP = \frac{1}{|U|}\sum_{i=1}^{U} AP_i \tag{13}$$

where $S^u_{predicted}$ indicates the top-k result list predicted for user u. $S^u_{visited}$ indicates the ground truth location visited by user u and the number of $S^u_{visited}$ is 1 in the next prediction task. For each sample to be predicted, if the ground truth location visited by user u appears in the top-k list, then A is the value of precision, otherwise, it takes 0.

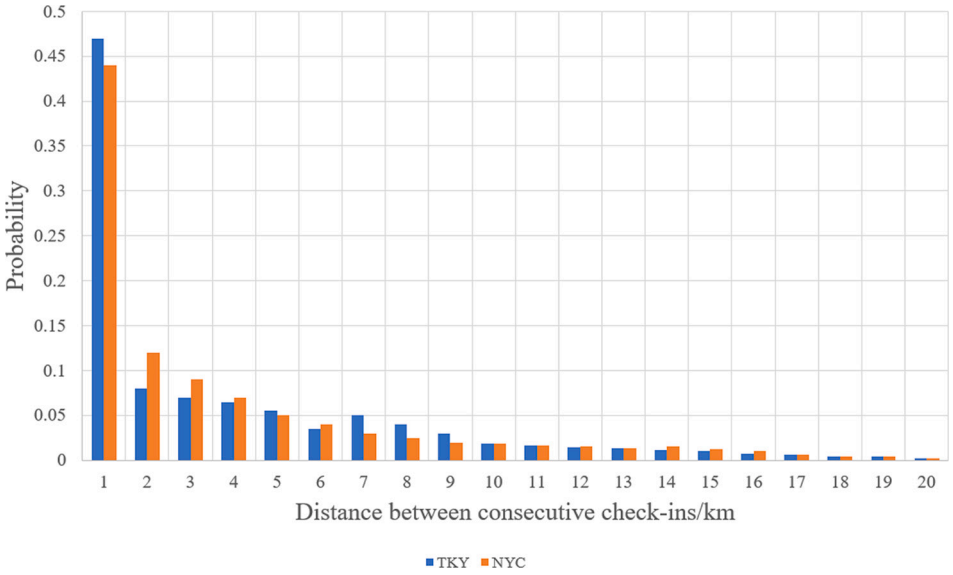

Fig. 3. Distribution of distances between adjacent check-in points.

Table 3
Experiment parameters of baseline.

Baseline	parameters
FMPC	Embedding_size :64
LSTM	Loc_emb_size:500; Time_emb_size:10; Hidden_size:500; Dropout:0.3; Learning_rate:1e-3;Window_size:72; Max_session_len:10
STRNN	Hidden_size:500
DeepMove	Loc_emb_size:500; Time_emb_size:10; u_emb_size:40; Hidden_size:500; Dropout:0.5; Learning_rate:1e3; Window_size:72; Max_session_len:10
LSTPM	Embedding_size:500; Learning_rate:1e-4

4.3. Baseline methods

In order to facilitate the comparison of the effect difference between GLSP and existing models, the five baseline algorithms are used for comprehensive comparison with our model. And the model parameters are from the reference values of the corresponding papers. The detailed parameters of each baseline model are Table 3:

- FPMC [20]: a model for location prediction by inducing human activity patterns.
- LSTM: the common RNN variants. It learns users' sequential behavior using recurrent neural networks.
- STRNN [15]: a variant RNN model for location prediction that considers both temporal and spatial intervals.
- DeepMove [4]: An RNN model that introduces an attention mechanism to model periodicity and uses LSTM to capture the spatiotemporal correlation of trajectories. Use recurrent neural networks to process historical and current trajectories to learn user preferences, and introduce an attention mechanism to calculate the similarity between the current trajectory and the historical trajectory.
- LSTPM [22]: An LSTM-based model for separately modeling user historical activity patterns and short-term behavioral intentions. LSTPM is touted as the state-of-the-art approach for the prediction task.

4.4. Parameter setting

For the division of the training data set, we transform the training dataset into different sets. Take a user u as an example, its data are $\{(T_u^1, T_u^2), (T_u^1, T_u^2, T_u^3, \ldots)\}$, which indicates the historical and current trajectory. Furthermore, eighty percent of the trajectory data of each user's data will be divided into the training set, and the remaining 20 percent of the data will be divided into the test set. For the training details on both datasets, we provide reference values for the parameters of our model. The parameter selection is obtained from the experiment, and the selection basis is explained in the experimental results. As shown in Table 4, our model uses the same size embedding dimension on the same dataset.

Table 4
Experiment parameters.

Parameters	Foursquare NYC	Foursquare TKY
Location embedding	300	400
Graph embedding	300	400
Learning rate	1e-4	1e-4
L2 regularization	1e-7	1e-6

Table 5
Performance comparation with baselines.

Dataset	NYC				TKY			
	ACC@1	ACC@5	ACC@10	MAP	ACC@1	ACC@5	ACC@10	MAP
FPMC	0.0964	0.2262	0.2943	0.1485	0.0695	0.1725	0.2385	0.1128
LSTM	0.1176	0.2424	0.2916	0.1716	0.1336	0.2738	0.3295	0.1972
STRNN	0.0987	0.2371	0.2580	0.1635	0.1208	0.2435	0.2927	0.1765
DeepMove	0.1408	0.2945	0.3632	0.2103	0.1284	0.2486	0.2925	0.1822
LSTPM	0.1558	0.3256	0.3976	0.2289	0.1577	0.3343	0.4035	0.2215
GLSP	**0.1561**	**0.3372**	**0.4091**	**0.2311**	**0.1584**	**0.3420**	**0.4230**	**0.2348**

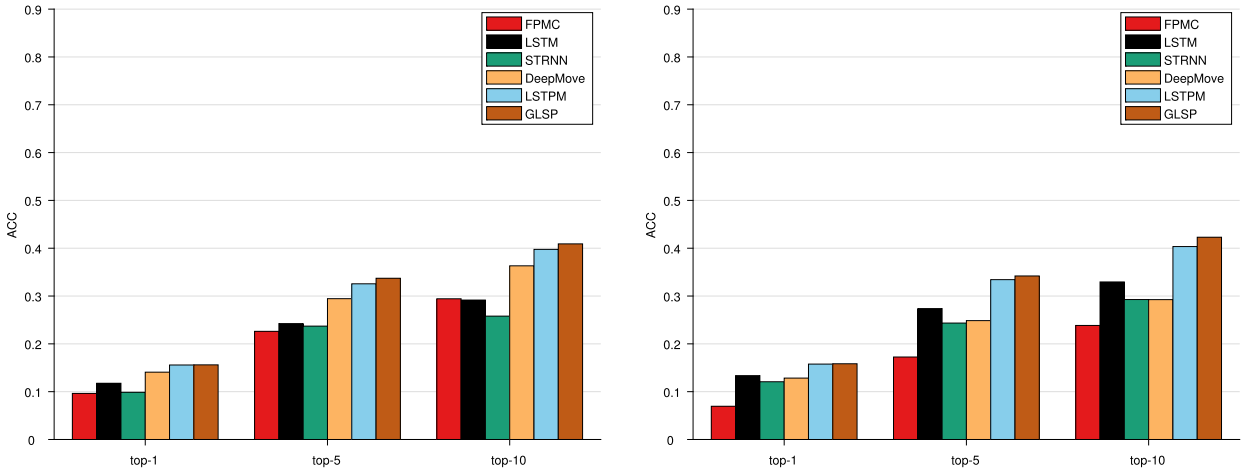

Fig. 4. Top-*k* results on Foursquare NYC and TKY.

4.5. Performance evaluation and discussion

This subsection shows the prediction performance results of all prediction methods under good training parameters. Table 5 demonstrates the experimental results evaluated by ACC@k and MAP for our next location prediction in NYC and TKY datasets.

Considering the more comprehensive performance comparison of the model, the bold data represents the performance of the model proposed in this paper, and the underlined data is the best comparative model performance.

It can be concluded from Table 5 that the evaluation indicators of the GLSP proposed in this article are significantly better than all baseline models on the two datasets. On the dataset Foursquare NYC, the GLSP model improves by 0.2%-61.9%, 3.5%-49.0%, and 2.8%-40.2%, respectively, compared to the comparison models in terms of ACC@1, ACC@5, and ACC@10. In terms of MAP indicators, GLSP is 55.62%, 34.67%, 41.34%, 9.89%, 0.96% higher than FPMC, LSTM, STRNN, DeepMove, and LSTPM, respectively; On the dataset Foursquare TKY, GLSP improves by 0.4%-127.9%, 2.3%-98.2%, and 4.8%-77.3% compared to the baseline model on metrics such as ACC@1, ACC@5 and ACC@10, respectively. These quantitative evaluations clearly reveal that GLSP effectively captures the spatiotemporal dependencies and long-term and short-term preferences in user check-in behavior.

As shown in Fig. 4 and Fig. 5, it can be found that the performance of the FPMC model is the worst, which means that the neural network-based method has better performance on the position prediction task than the matrix factorization-based method. This is because FPMC cannot fully consider the current actual situation and thus cannot well represent the user's preference characteristics. Analyzing the experimental results of a series of neural network methods, it can be found that DeepMove and LSTPM have higher accuracy than simple long-term and short-term memory networks, while STRNN's performance is slightly worse, which may be due to the length of historical trajectories that lead to LSTM and STRNN ability is affected. In addition, among all baseline methods, DeepMove and LSTPM perform well on relevant evaluation metrics, but neither outperforms the GLSP method (Fig. 5).

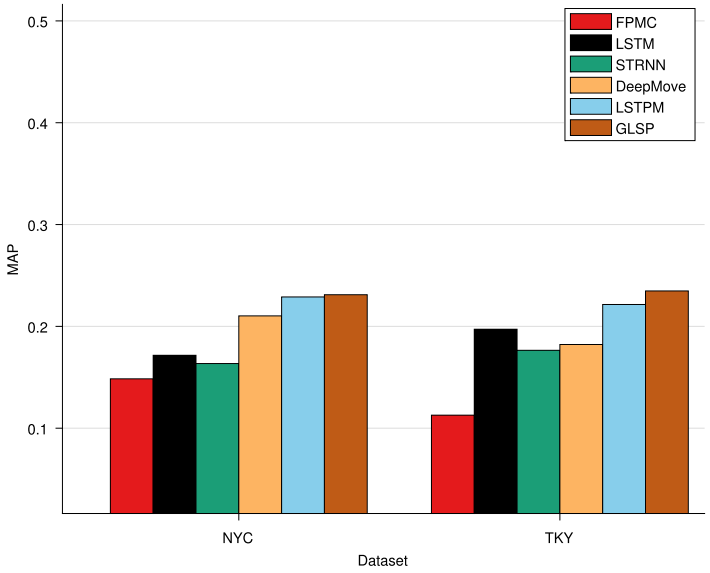

Fig. 5. MAP results on two datasets.

Table 6
Performance comparation with baselines.

Dataset	NYC			TKY		
	ACC@1	ACC@5	ACC@10	ACC@1	ACC@5	ACC@10
GLSP-I	0.1049	0.2089	0.2355	0.1054	0.2146	0.2532
GLSP-II	0.1143	0.2987	0.3754	0.1185	0.3021	0.3854
GLSP-III	0.1378	0.3076	0.3812	0.1451	0.3112	0.4097
Best-baseline	0.1558	0.3256	0.3976	0.1577	0.3343	0.4035
GLSP	**0.1561**	**0.3372**	**0.4091**	**0.1584**	**0.3420**	**0.4230**

The three methods of DeepMove, LSTPM and GLSP all use the user's historical trajectory and current trajectory to model the complex transformation relationship between places or user preferences. Although DeepMove uses LSTM to model long-term and short-term preferences and uses the attention mechanism to learn the attention weight between the current state and the historical state, it does not consider spatial context information such as distance, so it is slightly worse than LSTPM. LSTPM uses a non-local network to model long-term preferences and adds a geographically extended LSTM module when modeling short-term preferences, but it does not use a graph neural network to capture more hidden laws in historical trajectories, so it is difficult to further optimize the prediction accuracy, which is the key for GLSP to outperform the LSTPM method.

4.6. Ablation and effectiveness analyses

An ablation study should be performed to examine the performance of each component of GLSP. In addition, hyperparameter sensitivity experiments are used to check the stability of the model.

4.6.1. Evaluation of important components of GLSP

In order to compare the influence of different factors on the position prediction model and verify the contribution of different modules to the model performance gain, this section conducts ablation experiments, the operation method is similar to the control variable method, and the experimental performance of three GLSP model variants on the same dataset is evaluated, the variant model is set as follows:

- GLSP-I: A variant model that considers short-term behavioral intentions alone
- GLSP-II: A variant model that considers short-term behavioral intentions alone
- GLSP-III: A variant model that omits the graph neural network module
- GLSP: The GLSP model proposed in this paper has a comprehensive preference modeling module and uses a graph neural network to capture spatial dependencies in historical trajectories

Analysis of the data in Table 6 shows that GLSP-II outperforms GLSP-I on all evaluation metrics. This is because the user's long-term preference reflects the user's long-term habits, and the short-term preference reflects the user's current interest, and the

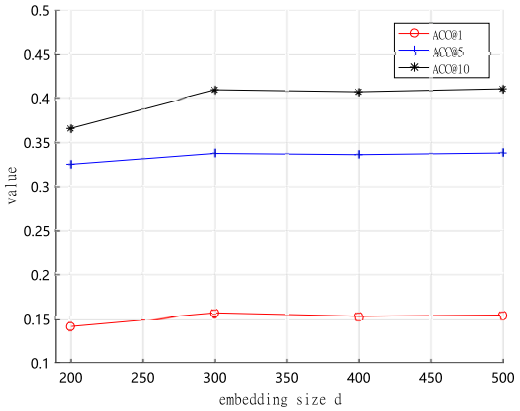

Fig. 6. Parameter sensitivity results of embedding dimension d on NYC dataset.

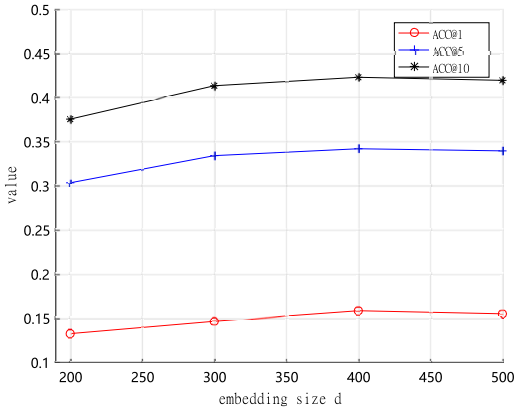

Fig. 7. Parameter sensitivity results of embedding dimension d on TKY dataset.

GLSP model also considers the influence of the user's current behavior and habits when learning the long-term preference. So GLSP-II has better performance than GLSP-I. In addition, the performance of the variant model GLSP-III including long-term and short-term preferences is close to the best baseline model, which indicates that these two preferences are crucial for predicting the user's next behavior. Specifically, compared with GLSP-III, GLSP has improved by 13.2%, 9.6% and 7.3% in ACC@1, ACC@5 and ACC@10 on the NYC data set; GLSP compared with GLSP-III in TKY data. The indicators such as ACC@1, ACC@5 and ACC@10 on the set have increased by 9.1%, 9.8% and 3.2%, which verifies that the spatial dependence modeling of the graph neural network is indeed helpful to improve the prediction accuracy of the model, and the less improvement in the ACC@10 index is due to the fact that with the increase of the length of the candidate position list, the final prediction result has a higher probability of hitting, so the performance improvement is relatively insignificant.

4.6.2. Hyper-parameters sensitivity experiment

Finally, the influence of hyper-parameter should be investigated. The experiment takes into account the sensitivity experimental analysis results of important hyper-parameters in GLSP: the embedding size d. The ACC@k metric is used to evaluate the performance of the model.

Fig. 6 and Fig. 7 depict the experimental results of hyperparameter sensitivity concerning the embedding dimension d on two datasets, respectively. It can be concluded that the validity of the model prediction first increases and then stabilizes as d increases, because higher-dimensional embeddings can represent more complex interaction information and thus capture more latent features. When the performance is optimal, the increase of dimensions will waste system resources. Fig. 6 shows that on the NYC dataset, when the dimension d reaches 300, the model reaches the best performance and tends to be stable; Fig. 7 shows that on the TKY dataset, when the dimension d reaches 400, the model reaches the best performance and tends to be stable. This is also the basis for the parameter reference values given in the previous section.

5. Conclusions

In this article, a location prediction model that integrates user preferences is presented, named GLSP. GLSP uses two methods to capture long-term behavioral patterns of users. GCN captures the spatial dependencies in the location graphs constructed by historical

trajectories and LSTM is used to obtain the temporal dependencies in the historical trajectory. Similarly, the current trajectory is also processed using LSTM to obtain the user's short-term behavioral intentions. Then we combine the user's long-term behavioral rules and short-term behavioral intentions to get the user's next action idea. Experimental results validate that GLSP can efficiently aggregate spatiotemporal information and perform better on location prediction task. For future work, attention mechanism may be introduced into GLSP to improve the prediction accuracy.

CRediT authorship contribution statement

Jinbo Liu: Conceptualization, Data curation, Methodology, Writing – original draft. **Yunliang Chen:** Project administration, Visualization. **Xiaohui Huang:** Writing – review & editing. **Jianxin Li:** Data curation, Validation. **Geyong Min:** Data curation, Validation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

[1] Ahmad Ali, Yanmin Zhu, Muhammad Zakarya, Exploiting dynamic spatio-temporal correlations for citywide traffic flow prediction using attention based neural networks, Inf. Sci. 577 (2021) 852–870.
[2] Emanuele Pio Barracchia, Gianvito Pio, Albert Bifet, Heitor Murilo Gomes, Bernhard Pfahringer, Michelangelo Ceci, LP-Robin: link prediction in dynamic networks exploiting incremental node embedding, Inf. Sci. (2022).
[3] Shuqin Cao, Libing Wu, Jia Wu, Dan Wu, Qingan Li, A spatio-temporal sequence-to-sequence network for traffic flow prediction, Inf. Sci. 610 (2022) 185–203.
[4] Jie Feng, Yong Li, Chao Zhang, Funing Sun, Fanchao Meng, Ang Guo, Depeng Jin, Deepmove: predicting human mobility with attentional recurrent networks, in: Proceedings of the 2018 World Wide Web Conference, 2018, pp. 1459–1468.
[5] Sébastien Gambs, Marc-Olivier Killijian, Miguel Núñez del Prado Cortez, Next place prediction using mobility Markov chains, in: Proceedings of the First Workshop on Measurement, Privacy, and Mobility, 2012, pp. 1–6.
[6] Qiang Gao, Fan Zhou, Ting Zhong, Goce Trajcevski, Xin Yang, Tianrui Li, Contextual spatio-temporal graph representation learning for reinforced human mobility mining, Inf. Sci. (2022).
[7] Yuqiang Han, Qian Li, Yang Xiao, Hucheng Zhou, Zhenglu Yang, Jian Wu, Multiple interleaving interests modeling of sequential user behaviors in e-commerce platform, World Wide Web 24 (4) (2021) 1121–1146.
[8] Liwei Huang, Yutao Ma, Shibo Wang, Yanbo Liu, An attention-based spatiotemporal lstm network for next poi recommendation, IEEE Trans. Serv. Comput. 14 (6) (2019) 1585–1597.
[9] Yoshiharu Ishikawa, Yuichi Tsukamoto, Hiroyuki Kitagawa, Extracting mobility statistics from indexed spatio-temporal datasets, in: STDBM, 2004, pp. 9–16.
[10] Dejiang Kong, Fei Wu, Hst-lstm: a hierarchical spatial-temporal long-short term memory network for location prediction, in: IJCAI, vol. 18, 2018, pp. 2341–2347.
[11] Mengzhang Li, Zhanxing Zhu, Spatial-temporal fusion graph neural networks for traffic flow forecasting, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, 2021, pp. 4189–4196.
[12] Xiaocui Li, Hongzhi Yin, Ke Zhou, Xiaofang Zhou, Semi-supervised clustering with deep metric learning and graph embedding, World Wide Web 23 (2) (2020) 781–798.
[13] Zekun Li, Zeyu Cui, Shu Wu, Xiaoyu Zhang, Liang Wang, Fi-gnn: modeling feature interactions via graph neural networks for ctr prediction, in: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, 2019, pp. 539–548.
[14] Jie Liao, Wei Zhou, Fengji Luo, Junhao Wen, Min Gao, Xiuhua Li, Jun Zeng, Sociallgn: light graph convolution network for social recommendation, Inf. Sci. 589 (2022) 595–607.
[15] Qiang Liu, Shu Wu, Liang Wang, Tieniu Tan, Predicting the next location: a recurrent model with spatial and temporal contexts, in: Thirtieth AAAI Conference on Artificial Intelligence, 2016, pp. 194–200.
[16] Chen Ma, Liheng Ma, Yingxue Zhang, Jianing Sun, Xue Liu, Mark Coates, Memory augmented graph neural networks for sequential recommendation, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, 2020, pp. 5045–5052.
[17] Xiaojuan Qi, Renjie Liao, Jiaya Jia, Sanja Fidler, Raquel Urtasun, 3d graph neural networks for rgbd semantic segmentation, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 5199–5208.
[18] Shaojie Qiao, Dayong Shen, Xiaoteng Wang, Nan Han, William Zhu, A self-adaptive parameter selection trajectory prediction approach via hidden Markov models, IEEE Trans. Intell. Transp. Syst. 16 (1) (2014) 284–296.
[19] Wanting Qin, Jun Tang, Songyang Lao, Deepfr: a trajectory prediction model based on deep feature representation, Inf. Sci. 604 (2022) 226–248.
[20] Steffen Rendle, Christoph Freudenthaler, Lars Schmidt-Thieme, Factorizing personalized Markov chains for next-basket recommendation, in: Proceedings of the 19th International Conference on World Wide Web, 2010, pp. 811–820.
[21] Hani Sami, Jamal Bentahar, Azzam Mourad, Hadi Otrok, Ernesto Damiani, Graph convolutional recurrent networks for reward shaping in reinforcement learning, Inf. Sci. (2022).
[22] Ke Sun, Tieyun Qian, Tong Chen, Yile Liang, Quoc Viet Hung Nguyen, Hongzhi Yin, Where to go next: modeling long- and short-term user preferences for point-of-interest recommendation, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, 2020, pp. 214–221.
[23] Diego Valsesia, Giulia Fracastoro, Enrico Magli, Learning localized generative models for 3d point clouds via graph convolution, in: International Conference on Learning Representations, 2018.
[24] Dongjing Wang, Xingliang Wang, Zhengzhe Xiang, Dongjin Yu, Shuiguang Deng, Guandong Xu, Attentive sequential model based on graph neural network for next poi recommendation, World Wide Web 24 (6) (2021) 2161–2184.
[25] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, Tat-Seng Chua, Neural graph collaborative filtering, in: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2019, pp. 165–174.

[26] Ziyang Wang, Wei Wei, Gao Cong, Xiao-Li Li, Xian-Ling Mao, Minghui Qiu, Global context enhanced graph neural networks for session-based recommendation, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 169–178.
[27] Gang Wu, Leilei Li, Xueyu Li, Yongzheng Chen, Zhiyong Chen, Baiyou Qiao, Donghong Han, Li Xia, Graph embedding based real-time social event matching for ebsns recommendation, World Wide Web 25 (1) (2022) 335–356.
[28] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, Tieniu Tan, Session-based recommendation with graph neural networks, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 2019, pp. 346–353.
[29] Shu Wu, Mengqi Zhang, Xin Jiang, Xu Ke, Liang Wang, Personalizing graph neural networks with attention mechanism for session-based recommendation, in: Thirtieth AAAI Conference on Artificial Intelligence, 2019, pp. 1–12.
[30] Yuxia Wu, Ke Li, Guoshuai Zhao, Qian Xueming, Personalized long- and short-term preference learning for next poi recommendation, IEEE Trans. Knowl. Data Eng. (2020).
[31] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Victor S. Sheng, Jiajie Xu, Fuzhen Zhuang, Junhua Fang, Xiaofang Zhou, Graph contextualized self-attention network for session-based recommendation, in: IJCAI, vol. 19, 2019, pp. 3940–3946.
[32] Xiaofeng Xu, Wenzhi Liu, Lean Yu, Trajectory prediction for heterogeneous traffic-agents using knowledge correction data-driven model, Inf. Sci. 608 (2022) 375–391.
[33] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, Jure Leskovec, Graph convolutional neural networks for web-scale recommender systems, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2018, pp. 974–983.
[34] Jia-Dong Zhang, Chi-Yin Chow, Geosoca: exploiting geographical, social and categorical correlations for point-of-interest recommendations, in: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2015, pp. 443–452.
[35] Pengpeng Zhao, Anjing Luo, Yanchi Liu, Fuzhen Zhuang, Jiajie Xu, Zhixu Li, Victor S. Sheng, Xiaofang Zhou, Where to go next: a spatio-temporal gated network for next poi recommendation, IEEE Trans. Knowl. Data Eng. (2020).
[36] Jianxing Zheng, Zifeng Qin, Suge Wang, Deyu Li, Attention-based explainable friend link prediction with heterogeneous context information, Inf. Sci. 597 (2022) 211–229.
[37] Ting Zhong, Shengming Zhang, Fan Zhou, Kunpeng Zhang, Goce Trajcevski, Jin Wu, Hybrid graph convolutional networks with multi-head attention for location recommendation, World Wide Web 23 (6) (2020) 3125–3151.
[38] Yu Zhu, Hao Li, Yikang Liao, Beidou Wang, Ziyu Guan, Haifeng Liu, Deng Cai, What to do next: modeling user behaviors by time-lstm, in: IJCAI, vol. 17, 2017, pp. 3602–3608.
[39] Wenjie Zi, Wei Xiong, Hao Chen, Luo Chen, Tagcn: station-level demand prediction for bike-sharing system via a temporal attention graph convolution network, Inf. Sci. 561 (2021) 274–285.