# Next POI Recommendation with Dynamic Graph and Explicit Dependency

**Feiyu Yin[1*], Yong Liu[2*], Zhiqi Shen[2], Lisi Chen[1], Shuo Shang[1,3†], Peng Han[1†]**

[1] University of Electronic Science and Technology of China
[2] Nanyang Technological University, Singapore
[3] Sichuan Artificial Intelligence Research Institute, Yibin, 644000, China
shirleyyfy@std.uestc.edu.cn, stephenliu@ntu.edu.sg, zqshen@ntu.edu.sg, lchen012@e.ntu.edu.sg, jedi.shang@gmail.com, penghan_study@foxmail.com

## Abstract

Next Point-Of-Interest (POI) recommendation plays an important role in various location-based services. Its main objective is to predict the user's next interested POI based on her previous check-in information. Most existing methods directly use users' historical check-in trajectories to construct various graphs to assist sequential models to complete this task. However, as users' check-in data is extremely sparse, it is difficult to capture the potential relations between POIs by directly using these check-in data. To this end, we propose the Sequence-based Neighbour search and Prediction Model (SNPM) for next POI recommendation. In SNPM, the RotatE knowledge graph embedding and Eigenmap methods are used to extract POI relationships implied in check-in data, and build the POI similarity graph. Then, we enhance the model's generalized representations of POIs' general features by aggregating similar POIs. As the context is typically rich and valuable when making Next POI predictions, the sequence model selects which POIs to aggregate not only depends on the current state, but also needs to consider the previous POI sequence. Therefore, we construct a Sequence-based, Dynamic Neighbor Graph (SDNG) to find the similarity neighbourhood and develop a Multi-Step Dependency Prediction model (MSDP) inspired by RotatE, which explicitly leverage information from previous states. We evaluate the proposed model on two real-world datasets, and the experimental results show that the proposed method significantly outperforms existing state-of-the-art POI recommendation methods.

## Introduction

The development of mobile technologies greatly motivates the popularity of location-based services. To experience more advanced features on these services, users are more willing to share their historical check-in data (Wang et al. 2019). The location-based services usually employ next Point-Of-Interest (POI) recommendation systems that are developed based on users' check-in behaviors to enhance the user experiences (Han et al. 2019; Yu et al. 2019). The objective of next POI recommendation is to predict a user's next destination based on her historical check-in records and current location. Actually, a considerable number of location-based services rely on the prediction of the next POI, such as assisting trip planning, precise advertising, and analyzing user behavior patterns (Shang et al. 2011; Han et al. 2020). Improving the prediction accuracy of next POI will help significantly improve the user experience.

Most existing methods treat the next POI recommendation problem as a sequential prediction task (Kong and Wu 2018; Liu et al. 2016). All these methods aim to explore the potential regularity of users' check-ins and reasonably infer where they will go next by summarizing their personalized preferences. Moreover, the spatio-temporal context is also usually used to assist in obtaining the features of users and POIs, and participating in the prediction process. The early works employ Recurrent Neural Networks (RNN) and attention mechanisms to capture the regularity of users' POI sequences (Cho, Myers, and Leskovec 2011; Yu et al. 2019). Recently, some graph-based models build the global POI transition graph and user similarity graph to enrich the learned user and POI representations (Li et al. 2021; Chen et al. 2021).

Although these methods have achieved some success in the next POI recommendation task, especially the models based on Graph Neural Networks (GNN), they may still suffer from the following deficiencies.

- **Construction of Graphs.** Most existing graph-based methods (Yuan, Cong, and Sun 2014; Rao et al. 2022; Yang, Liu, and Zhao 2022) directly build graphs (*e.g.*, POI transition graph and POI similarity graph) based on the POI sequences of users. However, it is difficult for them to mine the potential edges in the graphs, due to the extreme sparsity of check-in data. Moreover, some POI graphs studied in previous studies (Wang et al. 2022; Han et al. 2022) only pay attention to the connectivity between POIs, neglecting the weight of edges (either 1 or 0), thus increasing the difficulty in understanding POI graphs for the sequence modeling.
- **Aggregation of Neighbourhood Nodes.** The POI graphs constructed by previous methods (Feng et al. 2021; Wang et al. 2022) are almost static. To obtain more general representations of users and POIs, they usually use specific algorithms (*e.g.*, graph convolutional networks) to aggregate the neighbourhood information of POIs. Taking the POI transition graph as an example, when aggregating the POI information, the selected similar neighboring

*These authors contributed equally.
†Shuo Shang and Peng Han are corresponding authors.

nodes (*i.e.*, POIs) are fixed. They do not dynamically adjust the neighbour selection strategy based on the user's previous trajectory.

- **Prediction Model.** The recent methods (Sun et al. 2020) that extend RNN with gate mechanisms to overcome the disadvantage of RNN in capturing long-term dependencies are still sub-optimal. The current state representation in RNN is only derived from the current input and the hidden state from the previous timestamp. It leads to severe information oblivion in modeling the long-range temporal dependency and may also introduce noises into historical information due to the error accumulation by steps.

To remedy above challenges, we propose the Sequence-based Neighbour search and Prediction Model (SNPM) for next POI recommendation. Specifically, we first build a RotatE-Transition Graph (RTG) to represent the POI transition relationships based on the check-in data of all users. Then, the Spectral Clustering algorithm (Azran and Ghahramani 2006) is used to partition POIs into multiple non-overlapping clusters, where POIs with similar attributes are grouped into the same cluster. As RTG has the ability to mine potential transition relationships from historical trajectories, it can greatly improve the clustering effect. Moreover, our clustering algorithm places POIs with the same attributes into the same cluster, and obtains preliminary vector representations of these POIs.

Typically, previous methods (Yang, Liu, and Zhao 2022; Wang et al. 2022) directly apply Graph Neural Networks (GCN) on the constructed graph to update the representation of each POI. However, as mentioned above, they ignore the context of the current state. To solve this problem, we propose to use the obtained vector representations of POIs to construct a Sequence-based, Dynamic Neighbor Graph (SDNG). Finally, SDNG is used to as a part of the prediction model (*i.e.*, MSDP) to complete our prediction task. In MSDP, we explicitly compute the current hidden state representation based on multiple previous time steps by MSRNN, emphasizing the importance of prior states in the sequence prediction task to avoid oblivion of information and reduce the noise introduced.

In summary, the main contributions made in this work are as follows.

- We propose a new method for building the POI transition graph and obtaining POIs' vector representations, by adapting the idea of RotatE (Sun et al. 2019) to describe users' check-in data. This method can help alleviate the impacts caused by the extreme sparsity of users' POI trajectories.
- We propose SDNG to dynamically select the neighbouring nodes that should be aggregated for the current POI, based on the user's previous POI sequences. It achieves better performance than the static and single-point-based approach.
- We integrate the idea of RotatE into the RNN structure to build the prediction model (*i.e.*, MSDP). When computing the current state representation, MSDP explicitly exploits the hidden representations of multiple previous states by MSRNN, reducing the oblivion of information and the noise introduced.
- We conduct extensive experiments on two real-world datasets to demonstrate the performance of the proposed model. The experimental results show that the proposed SNPM model consistently outperforms the state-of-the-art POI recommendation methods.

## Related Work

The pioneering methods for next POI recommendation are mainly developed based on matrix factorization techniques. For example, to solve the problems caused by the high sparsity of check-in data, FPMC (Cheng et al. 2013) applies matrix decomposition to learn a personalized transition matrix for each individual user. PRME (Feng et al. 2015) uses metric embedding method to map POIs into low dimensional Euclidean space, and then uses the Euclidean distance to measure the similarity between POIs.

Recently, RNN structures and attention mechanisms have been extensively employed in next POI recommendation tasks. For instance, STRNN (Liu et al. 2016) constructs spatio-temporal matrices by calculating the spatio-temporal distances between consecutive check-ins to extend RNN for next POI recommendation. DeepMove (Feng et al. 2018) combines Gated Recurrent Unit (GRU) networks with an attention layer to learn the periodicity and sequential patterns of users' check-in data. Moreover, STAN (Luo, Liu, and Liu 2021) explicitly exploits relative spatio-temporal distances of all the check-ins, and considers personalized item frequency with a bi-layer attention architecture. Flashback (Yang et al. 2020) introduces periodicity and design a similarity function between previous states and current state. However, few of these models explicitly consider the impact of multiple previous time steps, leaving some key messages missing from their models and amplifying the noise introduced when computing current states of long trajectories.

As the check-in data for next POI recommendation can be simply described by sparse graphs, this property motivates the application of GNN for POI recommendation. For example, ARNN (Guo et al. 2020) uses meta-path based method to find related neighbor POIs in their knowledge graph. SGRec (Li et al. 2021) converts POI sequences into graphs by adding new nodes and edges relying on the POI transition relations. GETNext (Yang, Liu, and Zhao 2022) uses graph convolutional networks on the trajectory flow map to obtain a POI-to-POI probability map and reinforce the representations of POIs. DRAN (Wang et al. 2022) leverages the disentangled representations to explicitly model different aspects of a POI, by applying graph neural networks on the transition-based and distance-based POI relation graphs. All these methods do not use users' previous trajectories to adjust their neighbour selection strategy, in other words, their neighbor selection strategy is static.

## Preliminaries

This work focuses on the next POI recommendation task with a set of users $\mathcal{U} = \{u_1, u_2, \cdots, u_M\}$ and a set of POIs $\mathcal{L} = \{\ell_1, \ell_2, \cdots, \ell_N\}$, where $M$ and $N$ denote the

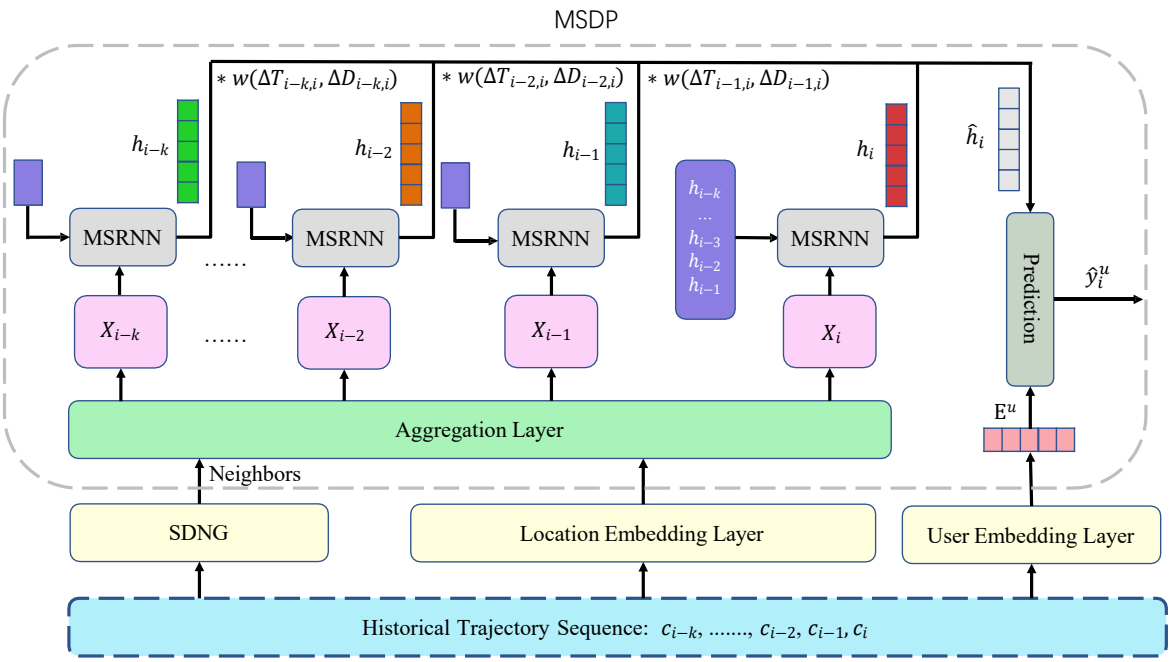



Figure 1: The overall architecture of the proposed next POI recommendation model, *i.e.*, SNPM.

number of users and items, respectively. Each POI $\ell_i$ owns a geographical coordinate $(lat, lon)$, denoting its latitude and longitude. Next, we introduce the definitions of some preliminary concepts.

**Definition 1: (Check-in)** A check-in is represented by $c = (u, \ell, t)$, which denotes that the user $u$ visits the location $\ell$ at the timestamp $t$.

**Definition 2: (User Trajectory)** A user trajectory consists of an ordered set of check-in records. Let $\mathcal{T}_{u_i} = \{c_1, c_2, \cdots, c_m\}$ represent the trajectory of user $u_i$, where $c_k$ denotes the $k$-th check-in in the trajectory, and $m$ denotes the length of the trajectory.

**Definition 3: (Next POI Recommendation)** Given the trajectory $\mathcal{T}_{u_i}$ of the user $u_i$, the objective of next POI recommendation is to recommend $u_i$ a list of top-ranked POIs that she may have interests at the next timestamp.

## Overall Framework

Figure 1 shows the overall architecture of SNPM. Observe that SNPM consists of two main components: 1) a novel sequence-based dynamic graph model, and 2) a mult-steps dependency prediction model that explicitly exploits multiple historical hidden states to update the current state.

## Dynamic Graph

Graph-based models are becoming popular recently, and they have been applied in the next POI recommendation task (Yang, Liu, and Zhao 2022; Wang et al. 2022). The graph itself plays an important role in determining performance of the graph-based model. However, how to construct the graph has not been well studied in POI recommendation task. In this section, we will propose a strategy to construct a dynamic POI graph for next POI recommendation, which can learn the relationships between different POIs for different time slots dynamically.

### Region Generation

To construct a graph describing the POI similarity relationships, we need to compute the similarity scores between all pairs of POIs. We first cluster POIs into different regions based on a static global graph that contains the similarity of transition patterns between POIs based on the historical trajectories. Then, we only compute the similarity scores between POIs belonging to the same region and set other similarity scores to 0.

How to construct the static global graph $G^g$ is the foundation of our clustering strategy. There are many ways to construct a graph for all POIs, and here we utilize an embedding method to learn the interactive relationship between them. For all contextual information and user check-in history, we could use a triplet to denote the sample of them. Then we could apply the embedding method from knowledge graph to learn the embeddings of all entities to construct the POI graph, where RotatE is the embedding method applied in our work. After that, for every pair POI $\ell_i$ and $\ell_j$, we could have a distance function $d(\ell_i, \ell_j)$ to reflect the interactive distance between them. The similarity score $s(\ell_1, \ell_2)$ is defined as

$$s(\ell_1, \ell_2) = e^{-d(\ell_i, \ell_j)}. \tag{1}$$

Now, the entry $G^g_{i,j}$ is defined by the similarity score as

$$G^g_{i,j} = \begin{cases} s(\ell_i, \ell_j) & \text{if } \ell_j \in \mathcal{N}_k(\ell_i) \text{ or } \ell_i \in \mathcal{N}_k(\ell_j),, \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

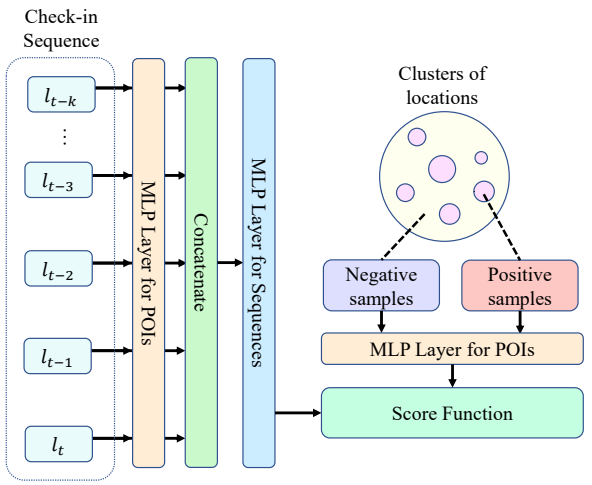



Figure 2: The architecture of SDNG.

where $\mathcal{N}_k(\ell_i)$ is the set of k-nearest neighbors for POI $\ell_i$ based on the similarity score.

To cluster the POIs into different regions based on $G^g_{i,j}$, we apply the Eigenmap method (Belkin and Niyogi 2003) to generate the embeddings for all POIs, which could also be used as the feature to construct the dynamic graph later. And our goal is to make POIs within one region to be highly relevant to each other while POIs from different regions are irrelevant. Suppose the dimension of node embedding is $d$, we can obtain the Eigenmap embedding in the following way: we first sort the eigenvectors $\{\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_n\}$ in the increasing order of eigenvalues by eigenvalue decomposition of $G^g_{i,j}$. Then for the $i$-th POI, we could get its embedding $\mathbf{e}_i \in \mathbb{R}^d$ as Equation (3):

$$\mathbf{e}_i = \{f_1(i), f_2(i), \ldots, f_d(i)\}, \tag{3}$$

where $f_j(i)$ is the $i$-th element in $j$-th eigenvector $\mathbf{f}_j$. In this way, the Eigenmap embedding could capture the topology information of the graph $G^g_{i,j}$

Once we have the POI embeddings, we then use the K-means method to cluster all POIs into $n_g$ regions $\{\mathfrak{G}_1, \mathfrak{G}_2, \ldots, \mathfrak{G}_{n_g}\}$. The distance metric for the K-means method is the l2-norm of our features between the sample and the cluster center. And the number of clusters $n_g$ is tuned due to the performance.

## Graph Construction

The visiting preference of users will be influenced by the POIs they have visited before. However, most of previous works construct a stable POI transition graph, which only contain the influence of current POI to next POI. To overcome this, we propose a dynamic, sequence-based neighbour Graph model (*i.e.*, SDNG) by learning method, where the connection is expected as the transition from visited POI sequence to next POI. The architecture of SDNG is shown in Figure 2.

Given one visited POI sequence $\mathbf{S}(\ell_t) = \{\ell_t, \ell_{t-1}, \ldots, \ell_{t-k}\}$ at $t$-th timestamp with the corresponding Eigenmap embeddings $\{\mathbf{e}_t, \mathbf{e}_{t-1}, \ldots, \mathbf{e}_{t-k}\}$, we firstly project their embeddings into a new space as

$$\tilde{\mathbf{e}}_t = \sigma\left(\mathbf{e}_t \mathbf{W}^p + \mathbf{b}^p\right), \tag{4}$$

where $\mathbf{W}^p \in \mathbb{R}^{d \times d_p}$ and $\mathbf{b}^p \in \mathbb{R}^{d_p}$ are learnable parameters, and $\sigma$ is the activation function such as ReLU. Then we could get the sequence embedding $\xi_t$ as

$$\xi_t = \sigma\left(\left(\tilde{\mathbf{e}}_t \| \tilde{\mathbf{e}}_{t-1} \| \ldots \| \tilde{\mathbf{e}}_{t-k}\right) \mathbf{W}^s + \mathbf{b}^s\right), \tag{5}$$

where $\|$ is the operation of concatenation, $\mathbf{W}^s \in \mathbb{R}^{kd_p \times d_p}$ and $\mathbf{b}^s \in \mathbb{R}^{d_p}$ are learnable parameters.

Given all visited POI sequences in the training dataset and the corresponding next POI $l_{t+1}$, we could construct the loss function for the dynamic graph as

$$\mathcal{L}_{seq} = -\sum_t \psi(||\xi_t - \tilde{\mathbf{e}}_t^-||_2^2 - ||\xi_t - \tilde{\mathbf{e}}_{t+1}||_2^2), \tag{6}$$

where $\psi$ is the logsigmoid function, and $\tilde{\mathbf{e}}_t^-$ is the learned embedding of negative POI $\ell_t^-$ which is randomly selected from the regions that don't contain POI $\ell_t$.

Once we have trained all parameters of SDNG, we only need to compute the connection weight between POIs in the same region. Then give any POI $\ell_i$ and its current sequence $\mathbf{S}(\ell_i)$, for the target POI $\ell_j$, we have the weight for the dynamic graph $G^d(\ell_i, \mathbf{S}(\ell_i), \ell_j)$ as

$$G^d(\ell_i, \mathbf{S}(\ell_i), \ell_j) = \begin{cases} s^d_{i,j} & \text{if } \ell_i \text{ and } \ell_j \text{ in the same region,} \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

where $s^d_{i,j} = e^{(-\rho||\xi_i - \tilde{\mathbf{e}}_j||_2^2)}$ and $\rho$ is hyperparameter that controls the influence proportion of neighboring POIs.

# Next POI Prediction Model

In this section, we introduce the overall framework of MSDP, which is used to complete the next POI recommendation task. Previous works (Yang et al. 2020) utilize existing sequence models, such as RNN, to encode the relationship from previous timestamp to current timestamp. However, these methods model this relationship implicitly with complex gate operation. To explicitly take into account multiple previous records, we propose a new recurrent neural network, named MSRNN, to improve the performance of next POI recommendation. The structure of MSRNN is illustrated in Figure 3.

## MSRNN

To explicitly exploit the dependency between different timestamps, we model the computation between latest $K$ visited POIs and current visiting POI explicitly. Inspired by the computation of RotatE, we use an explicit dependency relation operation $\mathbf{h}_{i-k} \circ \mathbf{r}_k$ to denote the influence caused by the $k$-th previous check-in to current check-in, where $\mathbf{r}_k$ is the $k$-th dependency relationship vector with $\|\mathbf{r}_k\|_2^2 = 1$, $\mathbf{h}_{i-k}$ is the hidden state of $k$-th previous check-in record, and $\circ$ is the Hadamard product which denotes the rotation operation. Note that $\mathbf{r}_k$ is a global trainable vector. For the

$i$-th timestamp, given the hidden state of $K$ previous check-ins, the influence $\mathbf{g}_i$ of previous sequence information to current check-in is calculated as follows,

$$\mathbf{g}_i = \sum_{k=1}^{K} \alpha_k (\mathbf{h}_{i-k} \circ \mathbf{r}_k), \tag{8}$$

where $\alpha_k$ is the weight to control the contribution of the $k$-th latest timestamp. Inspired by the success of attention mechanism, we exploit it to assign the value of $\alpha_k$ as follows,

$$\alpha_k = \frac{\exp\left((\mathbf{h}_{i-k} \circ \mathbf{r}_k)\,\mathbf{W}_a + \mathbf{b}_a\right)}{\sum_{j=1}^{K} \exp\left((\mathbf{h}_{i-j} \circ \mathbf{r}_j)\,\mathbf{W}_a + \mathbf{b}_a\right)}, \tag{9}$$

where $\mathbf{W}_a$ and $\mathbf{b}_a$ are trainable parameters of the attention layer. The hidden state of $i$-th timestamp is also influenced by its check-in, which is encoded in $\mathbf{q}_i$ as follows:

$$\mathbf{q}_i = \mathbf{X}_i \mathbf{W}_c + \mathbf{b}_c, \tag{10}$$

the $\mathbf{W}_c$ and $\mathbf{b}_c$ are trainable parameters, and $\mathbf{X}_i$ is the graph-based embedding of $i$-th timestamp which will be illustrated later. Finally, we obtain the hidden state $\mathbf{h}_i$ of $i$-th timestamp as follows,

$$\mathbf{h}_i = \sigma(\kappa \mathbf{q}_i + \mathbf{g}_i), \tag{11}$$

where the $\kappa$ is a hyperparameter that controls the weight of the current location information in the hidden state, and $\sigma$ is the activation function.

## Graph-based Embedding

Graph neural networks (GNN) (Scarselli et al. 2009) have been applied in many applications due to its high performance. The main computation of GNN is to revise the embedding of the entity by adding the embeddings of its neighbors. Given the dynamic graph in Equ (7), we could construct our graph-based embedding $\mathbf{X}_i$ as

$$\mathbf{X}_i = \mathbf{E}_{n_i} + \sum_j G^d(\ell_{n_i}, \mathbf{S}(\ell_{n_i}), \ell_j)\mathbf{E}_j, \tag{12}$$

where $\mathbf{E}_i$ is the learnable embedding for POI $\ell_i$, and $n_i$ is the index of POI for i-th timestamp. To reduce the computation time, we only utilize the embeddings of top-K neighbors based on the graph weight.

## Next POI Prediction

After we obtain the hidden state of all timestamps with MSRNN, we use the following normalized vector $\hat{\mathbf{h}}_i$ for the final prediction as

$$\hat{\mathbf{h}}_i = \frac{\sum_{j=1}^{i} \hat{w}_j \mathbf{h}_j}{\sum_{j=1}^{i} w_j}, \tag{13}$$

where $\hat{w}_j$ denotes similarity function $w(\Delta T_{i,j}, \Delta D_{i,j})$, which is the same as Flashback(Yang et al. 2020).

To model the interaction between the user and the POI, the concatenation of their embeddings is utilized, which is then fed into a fully connected layer to generate the probability $\hat{y}_i^u$ for all candidate POIs. This process is described as follows

$$\hat{y}_i^u = \Gamma(\mathbf{W}_f[\hat{\mathbf{h}}_i || \mathbf{E}_u]), \tag{14}$$

where $\mathbf{W}_f$ is a trainable weight matrix used to predict the probability, $\mathbf{E}_u$ is the embedding for user $u$, $\Gamma$ is the softmax function, and $||$ denotes the concatenation operation.

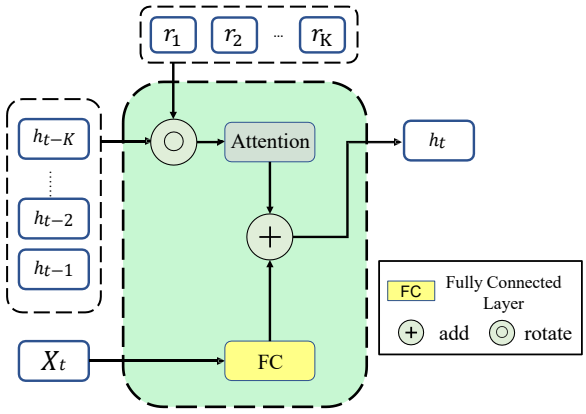



Figure 3: The structure of MSRNN

## Loss Function

We use the cross-entropy function to define the loss function used for learning the model parameters as follows,

$$Loss = -\sum_{u=1}^{|\mathcal{U}|} \sum_{i=1}^{m} \sum_{k=1}^{|\mathcal{L}|} f_{i,k}^u\,(\ell_k) \log\left(\hat{y}_{i,k}^u\right) + \beta ||\Theta||_2, \tag{15}$$

where $m$ denotes the length of the check-in sequence of each user, $f_{i,k}^u$ is an indicator function that is equal to 1 if $\ell_k$ is the ground truth and 0 otherwise, $||\Theta||_2$ represents the $L2$ regularization of all trainable parameters in prediction model, and $\beta$ is the hyperparameter to control the influence of regularization.

# Experiments

In this section, we perform experiments on real datasets to demonstrate the effectiveness of the proposed model.

## Datasets

The experiments are conducted on two widely used real world datasets: Gowalla[1] and Foursquare[2]. Gowalla dataset includes users' check-in data generated from February 2009 to October 2010, and the data in Foursquare is collected from April 2012 to January 2014. In these two datasets, each check-in consists of the user id, POI id, latitude, longitude, and timestamp. Following (Yang et al. 2020), we remove inactive users that have less than 100 check-in records. Then, each user's check-ins are sorted in chronological order based on the check-in timestamps. The first $80\%$ check-ins of each user are split into multiple length-equally (*e.g.*, 20) sequences, which are used as training set, Likewise, the remaining $20\%$ check-ins are used as testing set. Meanwhile, the transition graphs and similarity clusters used in the proposed model are built based on the training set. Table 1 summarizes the statistics of the experimental datasets.

[1]http://snap.stanford.edu/data/loc-gowalla.html
[2]https://sites.google.com/site/yangdingqi/home

| Dataset | #Users | #POIs | #Check-ins |
|---|---|---|---|
| Gowalla | 7,768 | 106,994 | 1,823,598 |
| Foursquare | 45,343 | 68,879 | 9,361,228 |

Table 1: Statistics of experimental datasets.

## Evaluation Metrics

Following (Yang et al. 2020), we use Accuracy@$k$ (Acc@$k$) and Mean Reciprocal Rank (MRR) as the evaluation metrics to measure the performance of next POI recommendation methods. Acc@$k$ estimates the ratio of true POIs occurring in the top-$k$ recommended POIs. In the experiments, we empirically set $k$ to 1, 5, and 10. As opposed to Acc@$k$, MRR measures the index of the correct POI in the ordered list of recommendations. The definitions of these metrics are as follows,

$$Acc@k = \frac{1}{m}\sum_{i=1}^{m} \dot{a}(rank_i \leq k),$$
$$MRR = \frac{1}{m}\sum_{i=1}^{m} \frac{1}{rank_i}, \tag{16}$$

where $\dot{a}$ is a discrimination function that returns 1 if the condition is true and 0, otherwise. $rank_i$ represents the index of the correct POI in the ordered list of recommendations, $m$ is the total number of POIs that need to be predict in testset.

## Baseline Methods

We compare the proposed SNPM model with the following baseline methods.

- PRME (Feng et al. 2015): This is a metric embedding method for capturing users' personalized sequential transition patterns.
- STRNN (Liu et al. 2016): This is a RNN-based method using customized spatio-temporal transition matrices to improve the POI recommendation performance.
- DeepMove (Feng et al. 2018): This method combines an attention layer with GRU to learn the periodicity and sequential patterns of users' check-ins.
- LBSN2Vec (Yang et al. 2019): This is a hyper-graph embedding method. In this method, the POIs are ranked based on their similarities to the user and timestamp embeddings.
- STGN (Zhao et al. 2019): This method introduces spatial and temporal gates to extend LSTM to learn the user's long-term and short-term preferences.
- LightGCN (He et al. 2020): This method use lightweight GCN to learns users' preferences on POIs based on a pairwise ranking loss.
- Flashback (Yang et al. 2020): This RNN-based method considers the similarities of multiple previous states to the current state during prediction.
- STAN (Luo, Liu, and Liu 2021) This attention-based method explicitly utilizes the spatio-temporal information of checkins to capture the correlations between POIs.
- GETNext (Yang, Liu, and Zhao 2022): This is an encoder-decoder framework that exploits the user trajectory graph and the POI transition probability graph for next POI recommendation.

## Experiment Settings

The experiments are performed in the environment with the following hardware platform: CPU: AMD Ryzen 5 3600, GPU: NVIDIA GeForce RTX 3090TI. When constructing the POI transition graph through RotatE, we choose 100 neighbors for each POI. When using spectral clustering for POI clustering, we choose the 20 smallest eigenvalues and the corresponding eigenvectors, and set $\omega = 10$ in SDNG. $\omega$ is the number of selected neighboring node. In the prediction Model, $\rho$ is set to $0.05$, $\kappa$ is set to $1/32$. The temporal decay factor $\eta$ and spatial decay factor $\varepsilon$ follow the default setting in Flashback (Yang et al. 2020). Moreover, the $L2$ regularization coefficient $\beta$ is set to $1.5e^{-6}$ and $2e^{-7}$ on Gowalla and Foursquare datasets, respectively. In MSRNN, the number of historical hidden state representations that are explicitly used is 6, Our implementation is available in Pytorch[3].

## Performance Comparison

Table 2 summarizes the POI recommendation performance achieved by different methods on Gowalla and Foursquare datasets. From Table 2, we have following observations.

- The results on both real-world datasets demonstrate that the proposed method SNPM outperforms all other state-of-the-art baseline methods, in terms of all evaluation metrics. Compared with the best baseline method GETnext, the improvement rates achieved by SNPM on all evaluation metrics are more than $5\%$. These results fully demonstrate the superiority of the propose method.
- As shown in Table 1, Gowalla dataset is more sparse than Foursquare dataset. SNPM achieves more improvements on Gowalla dataset. This indicates that SNPM is more effective in discovering implicit relationships between POIs on sparse dataset.

## Ablation Study

To study the effectiveness of the dynamic graph module and MSRNN in the prediction module, we perform ablation study experiments under the following three settings: 1) SNPM uses RNN and discards dynamic graph module for next POI recommendation; 2) SNPM uses RNN to replace MSRNN for POI recommendation; 3) SNPM uses MSRNN without the dynamic POI graph for POI recommendation. Table 3 shows the experimental results of ablation studies. From Table 3, we have the following findings.

- Both the dynamic graph module and multi-prior sequence dependency module (*i.e.*, MSRNN) achieve significant improvements over the original model (*i.e.*, Flashback). The dynamic graph module enhances the model's ability to represent POIs, and MSRNN reinforce the model's ability to handle long sequence information.

[3]https://github.com/Shirley-YFY/SNPM

| Methods | Gowalla | | | | Foursquare | | | |
|---|---|---|---|---|---|---|---|---|
| | Acc@1 | Acc@5 | Acc@10 | MRR | Acc@1 | Acc@5 | Acc@10 | MRR |
| PRME | 0.0740 | 0.2146 | 0.2899 | 0.1503 | 0.0982 | 0.3167 | 0.4064 | 0.2040 |
| STRNN | 0.0900 | 0.2120 | 0.2730 | 0.1508 | 0.2290 | 0.4310 | 0.5050 | 0.3248 |
| DeepMove | 0.0625 | 0.1304 | 0.1594 | 0.0982 | 0.2400 | 0.4319 | 0.4742 | 0.3270 |
| LBSN2Vec | 0.0864 | 0.1186 | 0.1390 | 0.1032 | 0.2190 | 0.3955 | 0.4621 | 0.2781 |
| STGN | 0.0624 | 0.1586 | 0.2104 | 0.1125 | 0.2094 | 0.4734 | 0.5470 | 0.3283 |
| LightGCN | 0.0428 | 0.1439 | 0.2115 | 0.1224 | 0.0540 | 0.1790 | 0.2710 | 0.1574 |
| Flashback | 0.1158 | 0.2754 | 0.3479 | 0.1925 | 0.2496 | 0.5399 | 0.6236 | 0.3805 |
| STAN | 0.0891 | 0.2096 | 0.2763 | 0.1523 | 0.2265 | 0.4515 | 0.5310 | 0.3420 |
| GETNext | 0.1419 | 0.3270 | 0.4081 | 0.2294 | 0.2646 | 0.5640 | 0.6431 | 0.3988 |
| SNPM | **0.1593** | **0.3514** | **0.4346** | **0.2505** | **0.2899** | **0.5967** | **0.6763** | **0.4278** |
| Improvement (%) | 12.26% | 7.46% | 6.50% | 9.19% | 9.56% | 5.80% | 5.16% | 7.27% |

Table 2: The POI recommendation performance achieved by different methods on Gowalla and Foursquare datasets. The best results are in boldface, and the second-best results are underlined.

Both of them help the sequence model capture the underlying POI transition patterns.

- Moreover, we also observe that the improvement in prediction accuracy is reduced, when the two modules work together. The potential reason is that both modules use sequence-based structure, and they may capture part of the same POI transition patterns from users' check-in sequences.

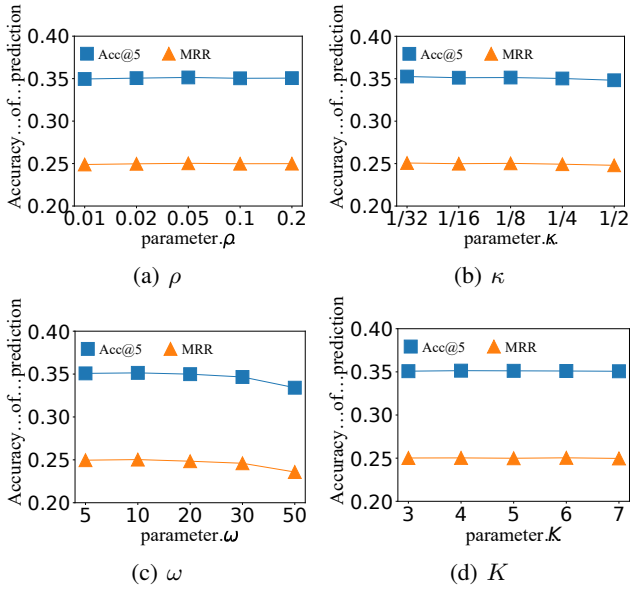
| Methods | Gowalla | | | |
|---|---|---|---|---|
| | Acc@1 | Acc@5 | Acc@10 | MRR |
| Flashback | 0.1158 | 0.2754 | 0.3479 | 0.1925 |
| $w/o$ dynamic graph | 0.1573 | 0.3488 | 0.4321 | 0.2482 |
| $w/o$ MSRNN | 0.1557 | 0.3476 | 0.4307 | 0.2469 |
| SNPM | **0.1593** | **0.3514** | **0.4346** | **0.2505** |

Table 3: Ablation studies on Gowalla dataset. SNPM $w/o$ dynamic graph denotes disabling the SDNG in SNPM, and SNPM $w/o$ MSRNN uses a RNN sequence model to generate the implicit state representation. Flashback denotes disabling both dynamic graph and MSRNN modules in SNPM.

## Hyper-parameter Sensitivity Analysis

We also perform experiments to investigate the impacts of the following four hyper-parameters on the performance of SNPM, including $\rho$, $\kappa$, $\omega$, and $K$ in MSRNN. $\rho$ controls the impact of neighboring nodes in information aggregation. $\kappa$ controls the weight of the current location information in the hidden state. $\omega$ represents the number of selected neighboring nodes. In MSRNN, $K$ denotes the number of hidden states of previous steps that should be considered in the POI recommendation procedure.

Figure 4 shows the performance trends of SNPM with respect to different settings of $\rho$, $\kappa$, $\omega$, and $K$. As shown in Figure 4, the variations in $\rho$, $\kappa$, and $K$ do not have significant impacts on the model performance. Moreover, when the number of aggregated neighbours $\omega$ exceeds 20, the model performance degrades. The reason is that the similarity of the chosen neighbors to current POI is reduced. In summary, the performance of SNPM is not sensitive to the settings of these hyper-parameters.




(a) $\rho$ (b) $\kappa$ (c) $\omega$ (d) $K$

Figure 4: Performance trends of SNPM on Gowalla dataset with respect to different settings of $\rho$, $\kappa$, $\omega$, and $K$.

## Conclusion and Future Work

To the end, we proposes a novel model, named SNPM, for next POI recommendation. In SNPM, we design a sequence-based dynamic neighbor graph model used to search for neighbors of the current POI, and a novel next POI prediction model which explicitly exploit the dependency relationships between different timestamps for better capturing the transition patterns. The experiments on real-world datasets demonstrate that SNPM consistently outperforms state-of-the-art POI recommendation methods. Moreover, we also perform ablation studies to investigate the effectiveness of the dynamic graph module and MSRNN module in the proposed method. For future work, we would like to develop more effective sequence-based neighbour search methods and include the user's personalized preference information into the prediction model.

## Acknowledgments

This work was supported by the NSFC (U2001212, U22B2037, U21B2046, 62032001, and 61932004), and Sichuan Science and Technology Program (2021YFS0007).

## References

Azran, A.; and Ghahramani, Z. 2006. Spectral Methods for Automatic Multiscale Data Clustering. In *CVPR*, 190–197. IEEE Computer Society.

Belkin, M.; and Niyogi, P. 2003. Laplacian Eigenmaps for Dimensionality Reduction and Data Representation. *Neural Comput.*, 15(6): 1373–1396.

Chen, Z.; Yao, B.; Wang, Z.; Gao, X.; Shang, S.; Ma, S.; and Guo, M. 2021. Flexible Aggregate Nearest Neighbor Queries and its Keyword-Aware Variant on Road Networks. *IEEE Trans. Knowl. Data Eng.*, 3701–3715.

Cheng, C.; Yang, H.; Lyu, M. R.; and King, I. 2013. Where You Like to Go Next: Successive Point-of-Interest Recommendation. In *IJCAI*, 2605–2611. IJCAI/AAAI.

Cho, E.; Myers, S. A.; and Leskovec, J. 2011. Friendship and mobility: user movement in location-based social networks. In *SIGKDD*, 1082–1090. ACM.

Feng, J.; Li, Y.; Zhang, C.; Sun, F.; Meng, F.; Guo, A.; and Jin, D. 2018. DeepMove: Predicting Human Mobility with Attentional Recurrent Networks. In *WWW*, 1459–1468. ACM.

Feng, S.; Chen, L.; Zhao, K.; Wei, W.; Li, F.; and Shang, S. 2021. Node2LV: Squared Lorentzian Representations for Node Proximity. In *ICDE*, 2015–2020. IEEE.

Feng, S.; Li, X.; Zeng, Y.; Cong, G.; Chee, Y. M.; and Yuan, Q. 2015. Personalized Ranking Metric Embedding for Next New POI Recommendation. In *IJCAI*, 2069–2075. AAAI.

Guo, Q.; Sun, Z.; Zhang, J.; and Theng, Y. 2020. An Attentional Recurrent Neural Network for Personalized Next Location Recommendation. In *AAAI*, 83–90. AAAI.

Han, P.; Li, Z.; Liu, Y.; Zhao, P.; Li, J.; Wang, H.; and Shang, S. 2020. Contextualized Point-of-Interest Recommendation. In *IJCAI*, 2484–2490. ijcai.org.

Han, P.; Shang, S.; Sun, A.; Zhao, P.; Zheng, K.; and Kalnis, P. 2019. AUC-MF: Point of Interest Recommendation with AUC Maximization. In *ICDE*, 1558–1561. IEEE.

Han, P.; Shang, S.; Sun, A.; Zhao, P.; Zheng, K.; and Zhang, X. 2022. Point-of-Interest Recommendation With Global and Local Context. *IEEE Trans. Knowl. Data Eng.*, 5484–5495.

He, X.; Deng, K.; Wang, X.; Li, Y.; Zhang, Y.; and Wang, M. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *SIGIR*, 639–648. ACM.

Kong, D.; and Wu, F. 2018. HST-LSTM: A Hierarchical Spatial-Temporal Long-Short Term Memory Network for Location Prediction. In *IJCAI*, 2341–2347. ijcai.org.

Li, Y.; Chen, T.; Luo, Y.; Yin, H.; and Huang, Z. 2021. Discovering Collaborative Signals for Next POI Recommendation with Iterative Seq2Graph Augmentation. In *IJCAI*, 1491–1497. ijcai.org.

Liu, Q.; Wu, S.; Wang, L.; and Tan, T. 2016. Predicting the Next Location: A Recurrent Model with Spatial and Temporal Contexts. In *AAAI*, 194–200. AAAI.

Luo, Y.; Liu, Q.; and Liu, Z. 2021. STAN: Spatio-Temporal Attention Network for Next Location Recommendation. In *WWW*, 2177–2185. ACM / IW3C2.

Rao, X.; Chen, L.; Liu, Y.; Shang, S.; Yao, B.; and Han, P. 2022. Graph-Flashback Network for Next Location Recommendation. In *KDD*, 1463–1471. ACM.

Scarselli, F.; Gori, M.; Tsoi, A. C.; Hagenbuchner, M.; and Monfardini, G. 2009. The Graph Neural Network Model. *IEEE*, 20(1): 61–80.

Shang, S.; Yuan, B.; Deng, K.; Xie, K.; and Zhou, X. 2011. Finding the most accessible locations: reverse path nearest neighbor query in road networks. In *SIGSPATIAL*, 181–190. ACM.

Sun, K.; Qian, T.; Chen, T.; Liang, Y.; Nguyen, Q. V. H.; and Yin, H. 2020. Where to Go Next: Modeling Long- and Short-Term User Preferences for Point-of-Interest Recommendation. In *AAAI*, 214–221. AAAI Press.

Sun, Z.; Deng, Z.; Nie, J.; and Tang, J. 2019. RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space. In *ICLR*. OpenReview.net.

Wang, Y.; Li, J.; Zhong, Y.; Zhu, S.; Guo, D.; and Shang, S. 2019. Discovery of accessible locations using region-based geo-social data. *World Wide Web*, 929–944.

Wang, Z.; Zhu, Y.; Liu, H.; and Wang, C. 2022. Learning Graph-based Disentangled Representations for Next POI Recommendation. In *SIGIR*, 1154–1163. ACM.

Yang, D.; Fankhauser, B.; Rosso, P.; and Cudré-Mauroux, P. 2020. Location Prediction over Sparse User Mobility Traces Using RNNs: Flashback in Hidden States! In *IJCAI*, 2184–2190. ijcai.org.

Yang, D.; Qu, B.; Yang, J.; and Cudré-Mauroux, P. 2019. Revisiting User Mobility and Social Relationships in LBSNs: A Hypergraph Embedding Approach. In *WWW*, 2147–2157. ACM.

Yang, S.; Liu, J.; and Zhao, K. 2022. GETNext: Trajectory Flow Map Enhanced Transformer for Next POI Recommendation. In *SIGIR*, 1144–1153. ACM.

Yu, Z.; Lian, J.; Mahmoody, A.; Liu, G.; and Xie, X. 2019. Adaptive User Modeling with Long and Short-Term Preferences for Personalized Recommendation. In *IJCAI*, 4213–4219. ijcai.org.

Yuan, Q.; Cong, G.; and Sun, A. 2014. Graph-based Point-of-interest Recommendation with Geographical and Temporal Influences. In *CIKM*, 659–668. ACM.

Zhao, P.; Zhu, H.; Liu, Y.; Xu, J.; Li, Z.; Zhuang, F.; Sheng, V. S.; and Zhou, X. 2019. Where to Go Next: A Spatio-Temporal Gated Network for Next POI Recommendation. In *AAAI*, 5877–5884. AAAI.