



# Attentive sequential model based on graph neural network for next poi recommendation

Dongjing Wang<sup>1</sup> · Xingliang Wang<sup>1</sup> · Zhengzhe Xiang<sup>2</sup> · Dongjin Yu<sup>1</sup> · Shuiguang Deng<sup>3</sup> · Guandong Xu<sup>4</sup>

Received: 15 January 2021 / Revised: 8 August 2021 / Accepted: 27 September 2021 /  
Published online: 20 October 2021

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2021

## Abstract

With the rapid development of Information Technology, there exist massive amounts of data available on the Internet, which result in a severe information overload problem. Especially, it becomes more and more challenging but necessary to help users find the contents or services that they really need. To address the problem mentioned above, recommender systems have been developed to exploit user's historical behavior data and provide personalized services for promoting customer experiences in many fields, such as Point of Interest (POI) applications, multimedia services, and e-commerce websites. Specifically, in POI recommendation, user's next check-in behaviors depend on both long- and short-term preferences. However, traditional recommendation methods often ignore the dynamic changes of user's short-term preferences over time, which limits their performance. Besides, many existing methods cannot fully exploit the complex correlations and transitions between POI in check-ins sequences. In this paper, we propose an Attentive Sequential model based on Graph Neural Network (ASGNN) for accurate next POI recommendation. Specifically, ASGNN firstly models user's check-in sequences as graphs and then use Graph Neural Networks (GNN) to learn the informative low-dimension latent feature vectors (embeddings) of POIs. Secondly, a personalized hierarchical attention network is adopted to exploit complex correlations between users and POIs in check-in sequences and capture user's long- and short-term preferences. Finally, we perform the next POI recommendation via leveraging user's long- and short-term preferences obtained from their behavior sequences with ASGNN. Extensive experiments are conducted on three real-world check-in datasets, and the results demonstrate that the proposed model ASGNN outperforms baselines, including some state-of-the-art methods.

**Keywords** Recommender system · Sequential recommendation · POI recommendation · Graph neural network · Attention

---

This research was supported by Natural Science Foundation of Zhejiang Province under No.LQ20F020015, the Fundamental Research Funds for the Provincial University of Zhejiang by Hangzhou Dianzi University under No.GK199900299012-017, and Zhejiang Provincial Key Science and Technology Program Foundation under No.2020C01165.

---

✉ Dongjin Yu  
yudj@hdu.edu.cn

Extended author information available on the last page of the article.

# 1 Introduction

Nowadays, with the rapid development of cloud computing, Internet of Things, large amounts of web applications and services have emerged, which trigger the explosive growth of online data and bring us into the big data era. In 2019, for example, the scale of transacting users of Meituan Dianping (a China's leading shopping platform for locally found consumer products and retail services) increased by 12.5%, and the number of service providers in Meituan Dianping, including hotels, restaurants, travel agencies, reached over millions<sup>1</sup>.

Then, it becomes more and more difficult for users to find contents or services that they are interested in, which results in a severe “information overload” problem.

Recommender systems [23, 29] have been proposed as an effective way to address the “information overload” problem in many fields, such as POI applications [41], online music services [37] job suggestions [16], business process management [7] and so on.

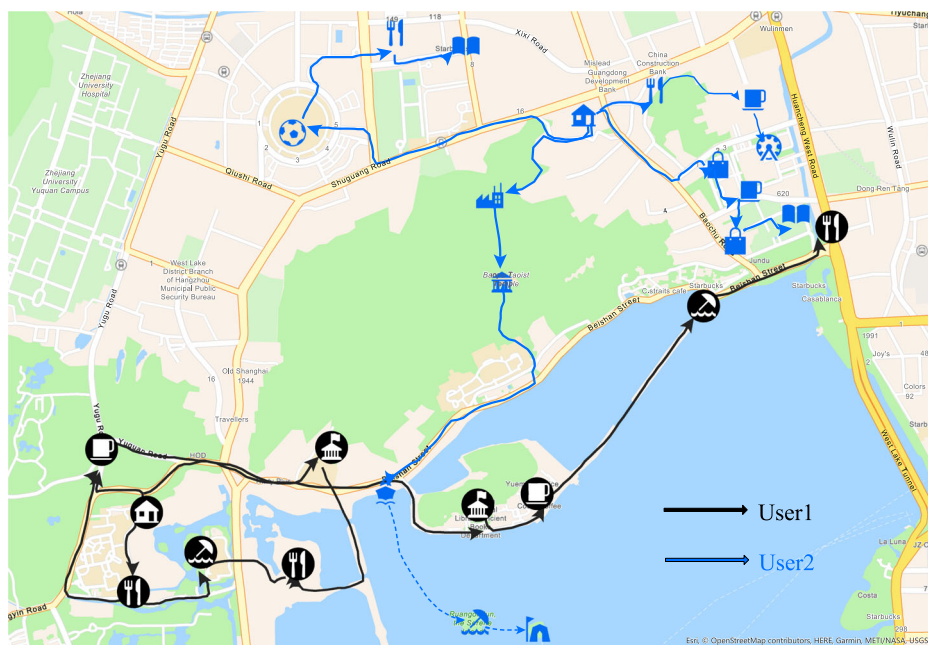
However, as for existing recommendation approaches, it is still a challenging task to fully exploit such a large amount of sequential behavior data for improving recommendation performance in an effective and efficient way. Especially, traditional recommendation methods based on machine learning require tedious manual feature engineering. Around 2015, a wave of deep learning [19] swept across both industrial and academic area, quickly causing an important technological transformation. Specifically, deep learning can utilize low-level data and information to form denser higher-level features and discover the distributed features of data. Therefore, with deep learning, the massive amounts of online data can be utilized in a more effective way. Especially, a lot of implicit information can be effectively used to improve the performance of personalized recommendations and alleviate the cold start problem.

As for POI recommendation, existing methods still face the following two challenges.

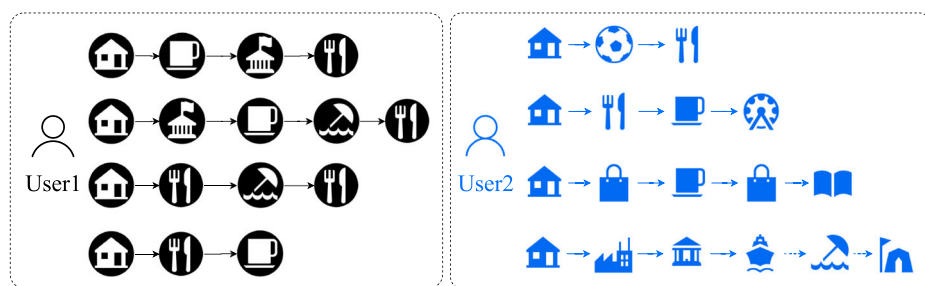
- (1) *How to fully exploit all user's check-in behaviors?* In many existing methods, user's behavior data are often modeled as sequences for recommendations. However, user's check-in behaviors generally contain many useful features that haven't been fully exploited. For example, as shown in Figure 1, the users may visit different POIs, such as stadium, museum, restaurant, and so on, which depends on their specific purpose or preferences. Besides, user's check-in sequences may become quite complex especially when a large number of POIs are available, which makes it difficult to directly capture user's sequential behavior patterns and preferences from their check-in sequences.
- (2) *How to capture and leverage user's long- and short-term preferences accurately?* user's check-in behaviors generally depend on both long- and short-term preferences, which may play different roles in POI recommendation. Specifically, user's long-term preferences are relatively stable over time. For example, in the scenario shown in Figure 1, the user1 has stable long-term preferences. Specifically, user1 likes visiting tourist attractions, coffee shops, and restaurants. Besides, user's short-term interests tend to change dynamically over time. For example, user2 (blue curve in Figure 1) enjoy various kinds of POIs, and she/he may visit mall or bookstore for shopping, or take part in outdoor activities, such as climbing or boating.

Especially, the prediction/recommendation for user2's next behavior mainly depends more on her/his short-term preferences.

<sup>1</sup><https://meituan.todayir.com/attachment/202003301717261783547356.en.pdf>



(a) check-in trajectories



(b) check-in sequences

**Figure 1** An example of two user's check-in records for POIs around West Lake in Hangzhou.

In the example mentioned above, the long- and short-term preferences of user1 and user2 have different effects on the recommendation tasks, so it is necessary to capture both preferences and leverage them in an adaptive and effective way.

To address the problems mentioned above, we propose the Attentive Sequential model based on Graph Neural Network (ASGNN) for next POI recommendation, which consists of the following four steps. Firstly, we construct user's historical check-in data as directed graphs. Specifically, each check-in sequence can be modeled as a directed graph, where the nodes include user nodes and POI nodes, and the edges in the graph are divided into edges from user nodes to POI nodes and edges between POI nodes. user's check-in behaviors can be very complex especially when there exist a lot of POIs. Most traditional methods model

user's behaviors as sequences and use a sliding window to capture sequential information, which may miss some important features. Compared with those strategies, the graph structure adopted in ASGNN can effectively model user's behavior patterns in a graph unified and localized way. Secondly, we use GNN to model user's preferences and POIs' features by exploring rich interactions/transitions between users and POIs. Specifically, a Gated Graph Neural Network (GGNN) is adopted to embed nodes in the graph into low-dimensional space and represent them as latent vectors (embeddings). Especially, GGNN can elegantly obtain user's check-in behavior patterns by learning to propagate and borrow information from other nodes/edges in the graph. Thirdly, we design a two-layer personalized hierarchical attention network to obtain user's personalized long- and short-term preferences for POIs and unify them in an adaptive way. Especially, the two-tier attention structure explores information from each record in the check-in sequences to capture user preferences and leverage long- and short-term preferences via assigning personalized weights. Finally, ASGNN uses the softmax layer to predict user's next behaviors and perform personalized recommendations for the target users.

The main contributions of this work are summarized as follows:

- We propose a new model named ASGNN that represents user's check-in behaviors as graphs and use GNN to learn user's behavior patterns and their preferences in a graph localized way for next POI recommendation.
- We design a personalized hierarchical attention mechanism to capture user's long- and short-term preferences and leveraged them adaptively for sequential recommendations.
- Extensive experiments are conducted on three real-world POI datasets, and the results show that the proposed model ASGNN outperforms baselines, including some state-of-the-art methods.

The remainder of this paper is organized as follows. We review existing related works in Section 2. Section 3 illustrates the key concepts and notations used in this paper. Section 4 describes the proposed sequential POI recommendation model. Detailed experimental results and analysis are presented in Section 5. Finally, we conclude our work in Section 6.

## 2 Related work

In this section, we describe the existing works on traditional and deep learning based recommender systems, and also present related works on GNN and attention mechanism that inspire this work.

### 2.1 Traditional recommendation methods

Traditional recommendation methods mainly include the following three categories: content-based methods, collaborative filtering methods, and hybrid methods.

The main idea of content-based (CB) recommendation methods [26] is to discover similar items based on the items users have already rated or interacted with and then recommend them. CB methods mainly rely on content information, such as user's profile and items' feature, instead of rating/interacting records, so there is no data sparsity problem. However, it usually encounters tedious feature engineering problems.

Collaborative filtering (CF) [9] recommendation methods are generally divided into user-based CF, item-based CF, and model-based CF. Specifically, user-based CF algorithms try to find a set of users (neighbors) who have similar interests with the target user, and then recommend neighbors' items that are not interacted by the target user. For example, Jia et al. [15] propose a user-based CF recommender system for tourist attraction recommendation, which has three steps: representation of user (tourist) information, generation of neighbor users (tourists) and the generation of attraction recommendations. Item-based collaborative filtering algorithm [31] measures the similarity between items based on rating/interacting records, and generate a recommendation list based on the similarity between candidate items and the target user's historical behaviors. For example, Linden et al. [23] use item-to-item CF method on Amazon for product recommendations. This method measures the similarity between products based on user's purchasing or rating behaviors, and recommends the products that are similar with the target user's historical records to them. Besides, there are many kinds of model-based CF algorithms, and matrix factorization (MF) [18] technique is one of the most widely used approaches. Specifically, MF is a latent factor model that transforms both items and users into low-dimension latent space that characterizes items and users with factors automatically inferred from user's feedback. Lian et al. [22] present a POI recommendation method named GeoMF, which exploits user's mobility records on location-based social networks (LBSNs) and models those implicit feedback data with weighted matrix factorization for POI recommendation.

Hybrid recommendation is proposed to address the shortcomings of each single recommendation method. Specifically, different recommendation strategies can be combined in complementary ways to achieve better performance. For example, Lekakos et al. [20] propose a hybrid movie recommendation approach based on content-based and collaborative filtering. Albadvi et al. [1] propose an online retail store recommendation technique, which extracts user's preferences in each product category separately and provides personalized recommendations based on various kinds of features, such as product taxonomy, attributes of product categories, and so on.

## 2.2 Deep learning based recommendation methods

Deep learning techniques can solve the problem of tedious manual feature problems in traditional machine learning by automatically discovering distributed high-level features, and they have widely applied in image recognition, natural language processing, recommender system and so on.

Specifically, deep learning can be used to capture the features of users or items. For example, Covington et al. [6] propose a content-based video recommendation algorithm based on deep learning. The method's main idea is to find the top- $n$  videos that are similar with the user preference vector with deep neural network. Elkahky et al. [8] propose a Multi-View Deep Neural Network (MVDN) model, which maps user/item entities to the same hidden space by deep neural network to generates recommendations based on the matching degree between users and items in hidden space.

In the area of collaborative filtering based recommender systems, Ma et al. [24] propose stacked autoencoders (SAE) POI recommendation approach, which combines collaborative filtering strategy with autoencoders to capture the complex features in check-in data. Hidasi et al. [11] adopt recurrent neural networks (RNN) to mine user's dynamic preferences in their historical behavior sequences for session-based recommendations. Jannach et al. [14] combine the K-nearest neighbor algorithm with RNN to enhances the scalability of recommendation model. Tang et al. [34] propose a Convolutional Sequence Embedding

Recommendation Model (Caser), which embeds a sequence of recent items into an “image” in time and latent space and uses a convolutional filter to learn the sequence patterns as local image features.

Deep learning is also widely applied in hybrid recommender systems. For example, Zhao et al. [46] design a spatio-temporal gated network (STGN) to incorporate both spatio and temporal information and capture both the long- and short-term preferences for accurate sequential POIs recommendation. Hsieh et al. [12] use collaborative metric learning to model the similarity between entities in data and then perform recommendation task. Bansal et al. [3] propose a recurrent neural network based method that uses GRU to learn a vector representation of text content for text recommendation.

### 2.3 Graph neural network

With the proliferation of graph-structured data, such as social network graphs, many research works on GNN have emerged. Especially, GNN and its variants, such as Graph Convolutional Network (GCN) and Graph Recurrent Networks (GRN), have been widely used in many tasks, including prediction and recommendation. For example, Scarselli et al. [32] propose a GNN based model with information propagation mechanism. Specifically, each node in the graph updates its state by exchanging information until all nodes reach some stable value. Wang et al. [39] represent the interaction data between users and POIs as higher-order connection graphs and learn user’s personalized preferences for POIs from the complex and nonlinear interactions in graphs.

Berg et al. [4] propose a graph self-encoder framework based on the user-item bipartite graph, which solves the problem of score prediction in recommendation systems from the perspective of link prediction. Ying et al. [43] develop a data-efficient GCN algorithm named PinSage for web-scale recommender systems. Specifically, PinSage combines random walks and graph convolutions to generate embeddings of nodes that incorporate both graph structure and node feature information in an efficient way. Li et al. [21] propose the GGNN, which uses the Gate Recurrent Units (GRU) [5] in the propagation step. It unrolls the recurrent neural network for a fixed number of  $T$  steps and backpropagates through time to compute gradients. GGNN is also applied in the session recommendation field. For example, Wu et al. [40] adopt GGNN for session-based recommendations, which uses an attention network to model user’s global preference and current interests.

Besides, graph attention networks, which combines GNN with attention networks, are proposed and applied in recommender systems. For example, Song et al. [33] propose a dynamic graph attention neural network based recommender system for online communities, which model user’s dynamic behaviors and context-dependent social influence with recurrent neural network and graph-attention neural network separately for social recommendation. Wang et al. [38] propose a Knowledge Graph Attention Network (KGAT) to link user-item instances together by attributes and fuse user-item and knowledge graph to form a unified network structure for recommendation.

### 2.4 Attention mechanism

Attention mechanism allows models to focus on important information in a large amount of data, significantly improving their efficiency and effectiveness, and it has been applied in many fields.

For example, Mnih et al. [25] combine the attention mechanism with RNN model for image classification. Besides, Bahdanau et al. [2] apply the attention mechanism to machine

translation in the NLP field. Attention mechanism is also applied in recommendation tasks to improve the performance. Ying et al. [42] propose a recommendation model based on hierarchical attention network. Specifically, the first attention layer is used to capture the user's long-term preferences, and the second attention layer is used to couple the user's long- and short-term preferences and output user's preference representation. Besides, Huang et al. [13] propose an attention-based spatio-temporal long and short-term memory (ATST-LSTM) network for next POI recommendation. Especially, the attention mechanism in ATST-LSTM enables it to focus on the relevant historical check-in records in a check-in sequence for better performance.

### 3 Preliminary

In this section, we introduce the sequential POI recommendation problem studied in this paper, and also present the definitions of key concepts. The main symbols and their explanation are given in Table 1.

In our sequential recommender system, the main goal is to predict which POI a user will visit next based on her/his historical check-in data.

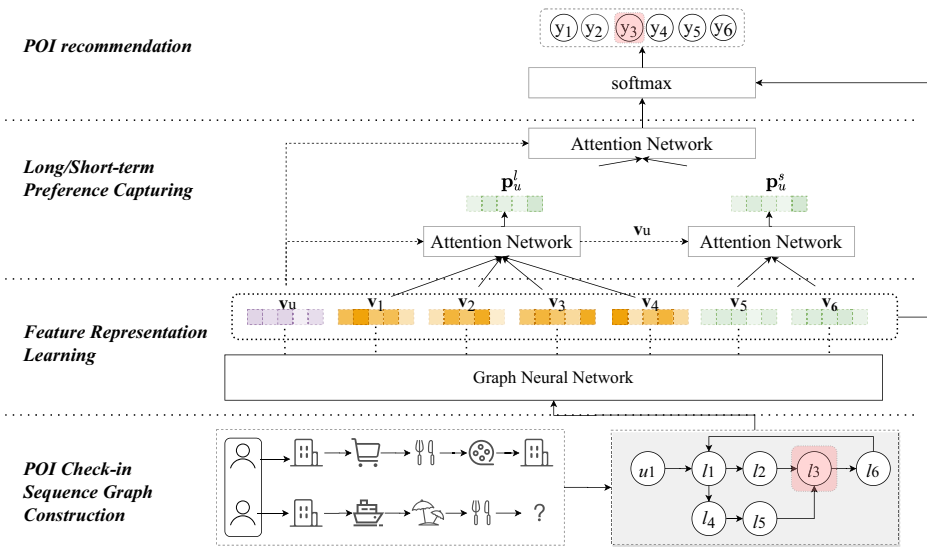
Formally, let  $L = \{l_1, l_2, \dots, l_m\}$  denotes the set of POIs in all check-ins sequences. Each user  $u$ 's check-in records can be represented by a time-stamped sequence  $S_u = [l_{u,1}, l_{u,2}, \dots, l_{u,n}]$ , where  $l_{u,i} \in L$  denotes the  $u$ 's  $i$ -th check-in record in  $S_u$ . The task of the sequential POI recommender system is to predict the  $u$ 's next check-in behavior  $l_{u,n+1}$  given her/his historical check-in sequence  $S_u$ . Specifically, each POI in the candidate set is ranked according to its score  $\hat{y}$ , and top ranked POIs will be recommended to the target user. The definitions of key concepts are given as follows:

**Definition 1 POI Check-in Sequence Graph.** POI check-in sequence Graph is defined as  $G = (V, E)$ , where  $V = (U, L)$  represents the vertex set, and  $U$  and  $L$  are user set and POI

**Table 1** Symbols used in this work

Symbol	Description
$G = (V, E)$	POI check-in sequence graph
$V = (U, L)$	vertex set
$E$	edge set
$e \in E$	an edge
$U$	user set
$u \in U$	a user
$L$	POI set
$l \in L$	a POI
$\mathbf{v} \in \mathcal{R}^d$	A $d$ -dimension embedding vector
$S_u$	Check-in sequence for user $u$ .
$\mathbf{A}$	adjacency matrix
$\mathcal{R}^m$	$m$ -dimensional Euclidean space
$\odot$	matrix corresponding element point multiplication
$\sigma$	sigmoid function
$\phi$	ReLU activation function





**Figure 2** The framework of ASGNN consists of four main components: 1) POI check-in sequence graph construction, 2) feature representation learning, 3) long/short-term preference capturing, and 4) POI recommendation

set, respectively.  $E$  represents the set of all edges, including user-POI edges and POI-POI edges.

**Definition 2 User-POI Interaction Edge.** User-POI interaction edge  $E(u, l_i)$  indicates that user  $u$  has interacted with POI  $l_i$ . Specifically, user-POI edges encode user's preferences and POIs' latent features. For example, two users share similar preferences for POIs if they have checked in the same/similar POIs.

**Definition 3 POI-POI Transition Edge.** POI-POI transition edges are defined as  $E(l_i, l_j)$ , which represents that user  $u$  visited POI  $l_j$  after visiting POI  $l_i$ . The edges between POIs indicate user's check-in behavior patterns that may be important for POI prediction/recommendation. For example, the edges with higher weights mean that there are frequent transitions between the corresponding POIs.

**Definition 4 Sequential POI Recommendation.** Given a user  $u \in U$  and her/his sequence  $S_u$  before time  $t$ , we need to predict the POI  $l \in L$  for which that user will check in at  $t$ .

## 4 Proposed model

As shown in Figure 2, the proposed model ASGNN consists of four main steps: 1) POI check-in sequence graph construction, 2) feature representation learning, 3) long/short-term preference capturing, and 4) POI recommendation.



#### 4.1 POI check-in sequence graph construction

Instead of modeling check-in behaviors as sequences directly like many existing methods, ASGNN firstly constructs POI check-in sequence graph to represent user's behavior data, and the reason is two-fold. Firstly, we do not need to truncate user's behavior sequences into fixed length since graph structure can represent the whole behavior sequence. Therefore, ASGNN can reserve more sequential information that is important for tasks of prediction and recommendation. Secondly, user's check-in behaviors are represented in a graph localized way for effective preferences/features propagation. Besides, the weights of edges in graph structure indicate the check-in counts of users on POIs. In this way, we can capture user's behavior patterns in a more effective way for better recommendation.

Specifically, we model each check-in sequence  $S$  as a directed graph  $G(V, E)$ .  $V = (U, L)$  represents the vertex set, where  $U$  is user set, and  $L$  is POI set. Besides,  $E$  represents the set of edges, including user-POI edges and POI-POI edges. Each user-POI edge  $e = (u, l)$  indicates the check-ins between user  $u$  and POI  $l$ , and every POI-POI edge  $e_{l_{u,i-1}, l_{u,i}}$  indicates that the user  $u$  has visited the POI  $l_i$  after  $l_{i-1}$  in sequence  $S_u$ . Since POIs may be checked in by users repeatedly, we assign a normalized weighted value to each edge. Specifically, the check-in count of POI/user node is divided by the out-degree of its starting node.

#### 4.2 Feature representation learning

After the construction of POI check-in sequence graph, we can learn the low dimensional vector representation (embedding) of POI and user through GNN. Each POI  $l_{u,i} \in L$  and user  $u \in U$  are embedded in a latent feature space, and the node vector  $\mathbf{v} \in \mathcal{R}^d$  represents a latent feature/preference vector (embedding) of a POI or user, where  $d$  is the dimension. The learned embeddings can capture the intrinsic features of POIs, and represent user's preferences for POIs. Especially, it avoids the problem that Markov's decision process requires a vast number of states.

In vanilla GNN, it is computationally inefficient to update the hidden states of all nodes iteratively. We use a GNN's variant inspired by RNN [5, 30], namely GGNN, with gate mechanism to diminish the restrictions of the vanilla GNN model and improve the effectiveness of the long-term information propagation across the graph. In this way, user's check-in data can be exploited in a more effective way to capture user's long-term preferences more accurately.

Formally, the update function for the embedding  $\mathbf{v}_i$  of each node on graph  $G_{V,E}$  is defined as follows.

$$\mathbf{a}_{s,i}^t = \mathbf{A}_i^u \cdot [\mathbf{v}_1^{t-1}, \dots, \mathbf{v}_n^{t-1}]^T \mathbf{H} + \mathbf{b}, \quad (1)$$

$$\mathbf{z}_{s,i}^t = \sigma(\mathbf{W}_z \mathbf{a}_{s,i}^t + \mathbf{U}_z \mathbf{v}_i^{t-1}), \quad (2)$$

$$\mathbf{r}_{s,i}^t = \sigma(\mathbf{W}_r \mathbf{a}_{s,i}^t + \mathbf{U}_r \mathbf{v}_i^{t-1}) \quad (3)$$

$$\tilde{\mathbf{v}}_i^t = \tanh(\mathbf{W}_o \mathbf{a}_{s,i}^t + \mathbf{U}_o (\mathbf{r}_{s,i}^t \odot \mathbf{v}_i^{t-1})), \quad (4)$$

$$\mathbf{v}_i^t = (1 - \mathbf{z}_{s,i}^t) \odot \mathbf{v}_i^{t-1} + \mathbf{z}_{s,i}^t \odot \tilde{\mathbf{v}}_i^t. \quad (5)$$

As shown in the Figure 3, the connection matrix  $\mathbf{A}^u \in \mathcal{R}^{n \times 2n}$  is defined as the concatenation of two adjacency matrices  $\mathbf{A}^{u,in}$  and  $\mathbf{A}^{u,out}$ , which determines how the nodes in the graph communicate with each other.

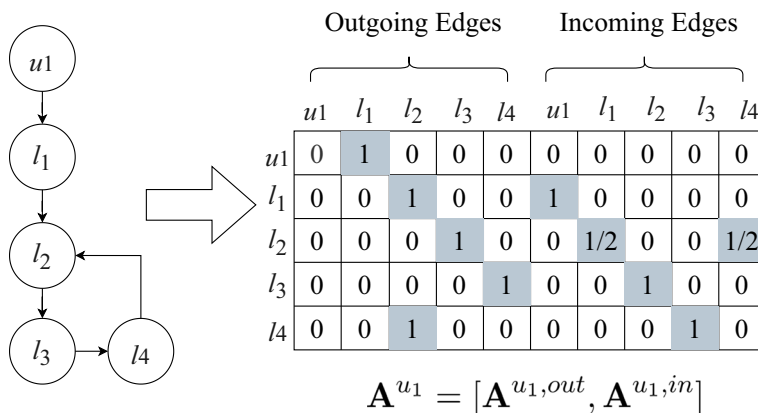
Specifically, Equation (1) defines the propagation process of information, such as user's behavior patterns or preferences, on incoming and outgoing edges between nodes in a graph localized way. Especially,  $\mathbf{a}_{s,i}^t \in \mathcal{R}^{d \times 2d}$  contains the activation information of the incoming and outgoing edges, and the adjacency matrix  $\mathbf{A}^u \in \mathcal{R}^{n \times 2n}$  defines the degree to which the nodes in the graph are connected to each other. Moreover, each element  $\mathbf{A}_{i,j}^u$  represents the weighted value of the edge between node  $l_i$  and  $l_j$ . An example of graph and the connection matrix  $\mathbf{A}^u$  are shown in Figure 3.  $\mathbf{A}_i^u \in \mathcal{R}^{1 \times 2n}$  are two columns of blocks in  $\mathbf{A}^u$  corresponding to node  $l_{s,i}$ .  $[\mathbf{v}_1^{t-1}, \dots, \mathbf{v}_n^{t-1}]$  are embeddings of POIs before  $t$  in the sequence  $S$ . Besides,  $\mathbf{H} \in \mathcal{R}^{d \times 2d}$  is the weighted matrix, and  $b$  is the bias term.

Equations (2) and (3) describe the calculation process of update gate  $\mathbf{z}_{s,i}^t$  and reset gate  $\mathbf{r}_{s,i}^t$ , respectively. Specifically,  $\sigma$  is the logistic sigmoid function, and  $\mathbf{W}_z$ ,  $\mathbf{U}_z$ ,  $\mathbf{W}_r$  and  $\mathbf{U}_r$  are trainable weight matrices that are learned in the optimization process. According to Equation (4), the candidate state is constructed based on the previous state, the current state, and the reset gate. Specifically,  $\tanh$  is the hyperbolic tangent activation function,  $\mathbf{W}_o$  and  $\mathbf{U}_o$  are weight matrices which are optimized in the training process.

Moreover, (5) defines how to update the state of each node based on the previous hidden state and the candidate state with updating gate.

### 4.3 Long/Short-term preference capturing

Many existing recommendation methods [10, 40] use attention mechanism to capture both short- and long-term preferences and combine them together. However, they often ignore user's personalized requirements and behavior patterns, and generally add up different preferences in a linear way. Especially, user's long- and short-term preferences may play different roles in the prediction/recommendation tasks, but they are generally assigned with the same or fixed weights in those methods.



**Figure 3** An example of graph and the connection matrix  $\mathbf{A}^{u_1}$

In ASGNN, we design a personalized hierarchical attention network to obtain user's long- and short-term preferences and combine them adaptively. Specifically, the two-layer attention mechanism can capture important information, such as user's preferences accurately by assigning different weights to the check-in records in behavior sequences. Similarly, user's long- and short-term preferences are leveraged in an adaptive way for better recommendation. For user  $u$ , her/his check-in sequence is denoted as  $S_u = [l_{u,1}, l_{u,2}, \dots, l_{u,n}]$ , where  $n$  is the number of records in sequence. Specifically,  $u$ 's long-term behaviors  $l_{u,1}, \dots, l_{u,n-1}$  indicate her/his long-term preferences. Formally,  $u$ 's long-term preference is defined as:

$$\mathbf{h}_i^l = \phi(\mathbf{W}_l \mathbf{v}_i + \mathbf{b}_l), \quad (6)$$

$$\alpha_{u,i}^l = \frac{\exp(\mathbf{v}_u^\top \mathbf{h}_i^l)}{\sum_{i \in S_u} \exp(\mathbf{v}_u^\top \mathbf{h}_i^l)}, \quad (7)$$

$$\mathbf{p}_u^l = \sum_{i \in S_u} \alpha_{u,i}^l \mathbf{v}_i, \quad (8)$$

where  $\mathbf{W}_l$  and  $\mathbf{b}_l$  are model parameters, and  $\phi$  is the ReLU activation function. Specifically, we first feed the dense low-dimensional embedding of each item  $i \in S_u$  through a multi-layer perceptron (MLP) to get the hidden representation  $\mathbf{h}_i^l$  based on (6).

Then we put the embedding  $\mathbf{v}_u$  of user  $u$  as the context vector and calculate the attention score  $\alpha_{u,i}^l$  as the normalized similarity between  $\mathbf{h}_i^l$  and  $u$  with the softmax function according to (7), which characterizes the importance of item  $l_i$  for user  $u$ . Finally,  $u$ 's long-term preference  $\mathbf{p}_u^l$  is calculated as the weighted sum of the POI embedding according to the attention scores in (8).

Besides,  $u$ 's short-term preference can be inferred from her/his last (most recent) check-in behavior  $l_{u,n} \in S_u$ , and it can be formally defined as:

$$\mathbf{h}_i^s = \phi(\mathbf{W}_s \mathbf{v}_i + \mathbf{b}_s), \quad (9)$$

$$\alpha_{u,i}^s = \frac{\exp(\mathbf{v}_u^\top \mathbf{h}_i^s)}{\sum_{i \in S_u} \exp(\mathbf{v}_u^\top \mathbf{h}_i^s)}, \quad (10)$$

$$\mathbf{p}_u^s = \sum_{i \in S_u} \alpha_{u,i}^s \mathbf{v}_i, \quad (11)$$

where  $\mathbf{W}_s$  and  $\mathbf{b}_s$  are model parameters,  $\alpha_{u,i}^s$  is the attention score, and  $\mathbf{p}_u^s$  is the representation of  $u$ 's short-term preference.

Then, we can combine the user's long-term preferences with short-term preferences in an adaptive way to obtain her/his personalized preference  $\mathbf{p}_u$ , which is formally defined as:

$$\mathbf{p}_u = \beta_l \mathbf{p}_u^l + \beta_s \mathbf{p}_u^s, \quad (12)$$

where  $\beta_l$  is the weight of long-term user preference, and  $\beta_s$  is the weight of short-term user preference.

#### 4.4 POI recommendation

Finally, we can use the softmax layer to calculate the possibility (score) that the target user  $u$  interacts with the candidate POI  $l_i$  based on  $u$ 's preference  $\mathbf{p}_u$  and  $l_i$ 's feature  $\mathbf{v}_i$ , and

perform next POI recommendation. Formally, the score  $\hat{\mathbf{z}}_{u,i}$  for each candidate POI  $l_i \in L$  is calculated as follows:

$$\mathbf{z}_{u,i}^{\wedge} = \mathbf{p}_u^T \mathbf{v}_i. \quad (13)$$

Then we can apply a softmax function to calculate the normalized output vector  $\hat{\mathbf{y}}$ :

$$\hat{\mathbf{y}} = \text{softmax}(\hat{\mathbf{z}}), \quad (14)$$

where  $\hat{\mathbf{z}}$  denotes the recommendation scores over all candidate POIs and  $\hat{\mathbf{y}}$  denotes the probability of POIs being interacted by the target user next.

For each POI Check-in Sequence Graph, the loss function can be written as follows by using cross-entropy:

$$\mathcal{L}(\hat{\mathbf{y}}) = - \sum_{i=1}^m y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i), \quad (15)$$

where  $\mathbf{y}$  denotes the one-hot encoding vector of the ground truth item.

## 5 Experiments

In this section, we evaluate the performance of the proposed model ASGNN on three real-world datasets. Specifically, the experiments are designed to answer the following five research questions:

- **RQ1:** Does the proposed model ASGNN outperform state-of-the-art baselines in sequential POI recommendation tasks?
- **RQ2:** What are the effects of the key components in the ASGNN architecture, and can they effectively capture key features for improving the performance of recommendation?
- **RQ3:** How does the dimension of embedding in ASGNN influence the recommendation results?
- **RQ4:** How do ASGNN and baselines perform on datasets with different sparsity?
- **RQ5:** Does ASGNN learned the POI embedding effectively?

### 5.1 Experimental designs

#### 5.1.1 Datasets

We evaluate the proposed model ASGNN on three real-world datasets as follows:

- **Gowalla**<sup>2</sup> is a LBSN website that allows users to share locations by checking-ins. Specifically, if user  $u$  visits POI  $l$  at time  $t$ , there will be a tuple record  $(u, l, t) \in S_u$ , where  $S_u$  is  $u$ 's sequential behavior sequence. Specifically, we filtered out POIs with less than five occurrences and remove users who only have one interaction record. The final Gowalla dataset contains 307,376 check-ins by 6,533 users to 23,329 POIs.
- **FourSquare**<sup>3</sup> is a location platform based on user's geographical information. It encourages mobile phone users to share their current location with others. Specifi-

<sup>2</sup><https://snap.stanford.edu/data/loc-gowalla.html>

<sup>3</sup><https://sites.google.com/site/yangdingqi/home/foursquare-dataset>

**Table 2** Complete statistics of the dataset

Dataset	Gowalla	Foursquare	Brightkite
user	6,533	1,809	1,932
POI	23,329	5,514	6,056
check-in record	307,376	128,147	40,040
time range	04/2012-09/2013	02/2009-10/2010	04/2008-10/2010

cally, we remove POIs whose frequency less than five and retained user's behavioral sequences with more than one POI. The final Foursquare dataset contains 128,147 records between 1,809 users and 5,514 POIs.

- **Brightkite**<sup>4</sup> was once a location-based social networking service provider where users shared their locations by checking-in. Specifically, we remove POIs whose frequency less than two and retained user's behavioral sequences with more than one POI. The final Foursquare dataset contains 40,040 records between 1,932 users and 6,056 POIs.

In each dataset, we randomly select 80% of all check-in records as the training set and use the remaining 20% as the test set. The statistical information for all datasets is shown in Table 2.

### 5.1.2 Baseline methods

In this section, we compare the proposed model ASGNN with seven baselines, including some traditional recommendation algorithms and several state-of-the-art methods based on deep learning techniques, such as GNN and attention mechanism.

- **POP** method ranks candidate items based on their popularity and then recommends the top ranked items to the target users.
- **BPR** [28] is a classic recommendation algorithm that optimizes the pairwise rank loss on implicit feedback data for top- $n$  recommendation.
- **FPMC** [27] combines matrix factorization with Markov chains to model user's sequential behaviors for recommendation.
- **HRM** [36] represents each user or item as a feature vector in continuous space, and employs a two-layer model to learn the hybrid preference/feature representations of users and items from transaction data for next-basket recommendation.
- **CPAM** [44] combines skip-gram based POI embedding model with logistic matrix factorization to incorporate both context influence and user preference for POI recommendation.
- **SHAN** [42] uses a hierarchical attention network for behavior sequence modeling and recommendations
- **SRGNN** [40] is a session-based recommendation method that uses the GNN to model user's behavior data and learn vector representations of users and items for recommendation.

<sup>4</sup><https://snap.stanford.edu/data/loc-brightkite.html>

### 5.1.3 Evaluation metrics

The proposed model ASGNN was evaluated with two metrics, i.e., recall and Mean Reciprocal Rank (MRR).

- **Recall** is defined as

$$Recall@n = \frac{\#(hit)}{\#(testcase)} \quad (16)$$

where  $\#(hit)$  is the number of ground truth items that appear in the recommended list and  $\#(testcase)$  is the number of all testcases. Especially, recall is widely used to evaluate the quality of the recommendation results. To be precise, it is the ratio of the number of user-interacted POIs to the total number of POIs in the recommendation results. A larger recall value indicates better recommendation result.

- **MRR** is a ranking evaluation metric which is the average of the reciprocal ranks of the target POIs in a recommendation list. Formally, MRR is defined as

$$MRR@n = \frac{1}{\#(all)} \times \sum \frac{1}{rank_i} \quad (17)$$

where  $rank_i$  denotes the rank position of the  $i$ -th test target POI in the recommendation list. If  $rank_i > n$ ,  $\frac{1}{rank_i} = 0$ . Specifically, the MRR is the average of the reciprocal degree of the target POIs, and it mainly measures the ranking order of items in recommendation list. Especially, a larger MRR value means better result.

### 5.1.4 Implementation details

All trainable parameters are initialized using a Gaussian distribution with a mean of 0 and a standard deviation of 0.1, and they are optimized in small batches using Adam optimizer [17]. Besides, we set the learning rate to 0.001, the batch size to 100, number of epochs to 30. All experiments are performed on a server with Intel(R) Xeon(R) Silver 4108 CPU, GeForce RTX 2080Ti, 128GB memory, and running Ubuntu 18.04, python 3.6, TensorFlow 1.15.

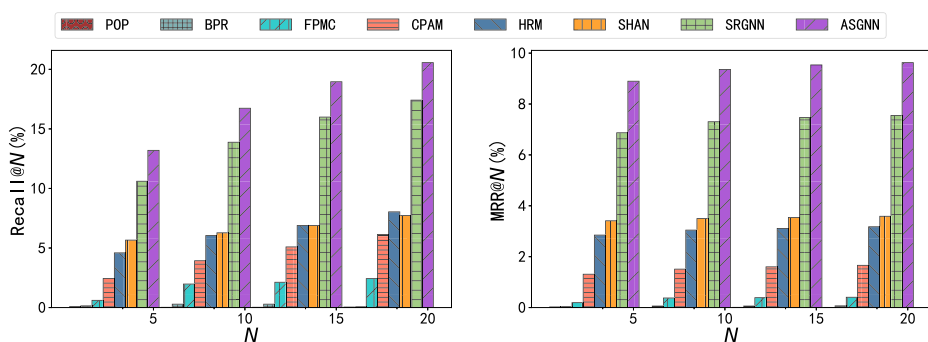
## 5.2 Comparison with baselines (RQ1)

We firstly compare the proposed model ASGNN with the baselines mentioned above to evaluate ASGNN's performance in detail. The experimental results on Gowalla and Foursquare and Brightkite datasets are shown in Figure 4a and b, and c, separately.

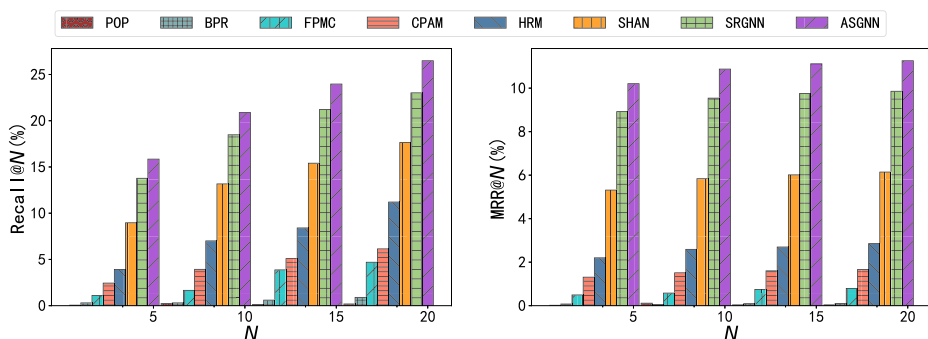
As shown in the experimental results, POP and BPR do not perform well on the all three datasets, with their recall@20 values under 1% in most cases. Specifically, the performance gap between those two methods and other baselines is quite large. The reason is that the BPR method is based on the interaction data between users and POIs and it ignores the sequential information in the check-in data. The POP method only considers the popularity of the POI, so it has the lowest performance on all three datasets.

Especially, user's check-in sequences contain information and features that are important for improving POI recommendation, such as user's behavior patterns as well as the correlations between POIs in check-in sequences.

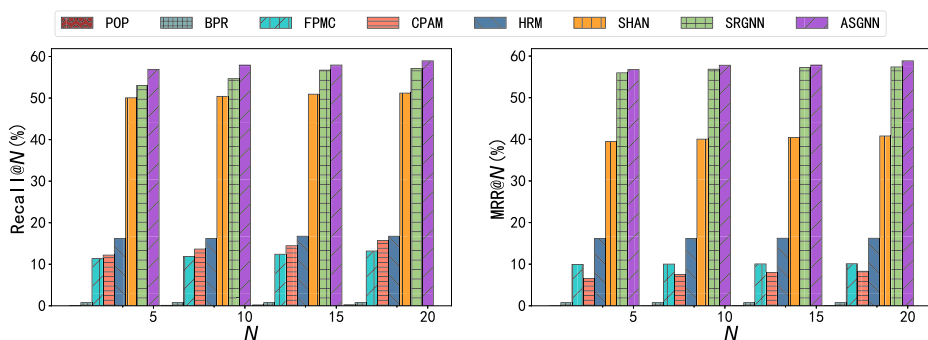
FPMC and CPAM incorporate the sequential information in user's check-in sequences, and they are more effective than POP and BPR algorithms. For example, as shown in Figure 4a and b, and c, the recall@20 for FPMC and CPAM algorithms range from 2%



(a) Gowalla dataset



(b) Foursquare dataset



(c) Brightkite dataset

**Figure 4** Performance comparison between ASGNN and baselines

to 6% on Gowalla Dataset. Specifically, FPMC algorithm combines matrix factorization with Markov chains to capture user's long-term preference and sequential information in POI check-in data. Besides, CPAM can capture the contextual influence between POIs in check-in sequences and combine it with user's preferences for POI recommendation. However, FPMC and CPAM do not explicitly learn user's short-term preferences or combine them with user's long-term preferences, so their performance is not as good as the proposed



model ASGNN, which shows both long- and short-term preferences captured by ASGNN are important for accurate recommendation.

Compared with FPMC which combines user's general tastes and sequential behavior in a linear way, HRM, SHAN, and SRGNN capture and combine user's long- and short-term preferences in a more effective way, so they achieve better results. For example, the recall@20 of HRM and SHAN algorithms are between 7% and 8% on Gowalla Dataset, respectively, and the recall@20 for the SRGNN algorithms reached 17.41% on Gowalla dataset. Specifically, HRM uses a two-layer model to construct hybrid preference/feature representations (embeddings) of users and items from check-in sequences with average or maximum pooling strategies, so it performs well on POI recommendation task. Besides, SHAN algorithm adopts a hierarchical attention network to capture user's long- and short-term preferences from behavior sequences and combines them together for better results. Similarly, the SRGNN algorithm can learn and combine user's long- and short-term preferences with GNN.

Moreover, the proposed model ASGNN still outperform those three baselines, and the reason is that the long- and short-term preferences may play different roles for each user in POI recommendation. Especially, ASGNN can accurately model and capture both preferences with GNN and leverage them via a two-layer personalized hierarchical attention network in a more effective and adaptive way for better recommendation performance.

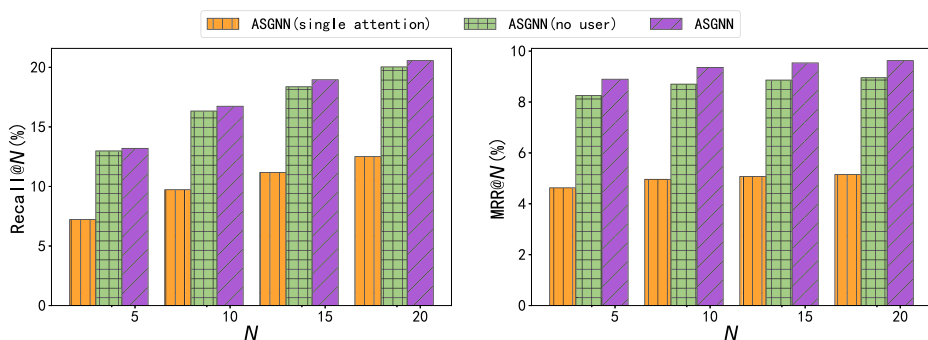
At the same time, we notice that the performance of ASGNN and all baselines on the Foursquare and Gowalla dataset are not as good as the results on the Brightkite dataset. For example, the recall@20 for the FPMC algorithm reached 13.18% on Brightkite dataset, compared with 4.69% on the Foursquare dataset. One reason is the users in Brightkite dataset tend to have repeated check-in behaviors on the same POI, while the users in Foursquare and Gowalla dataset usually like exploring different POIs. Moreover, the proposed model ASGNN still maintains the best results, which shows that ASGNN is more adaptive to datasets with different behavior patterns and size. Furthermore, we design experiments on datasets with different sparsity in Section 5.5.

### 5.3 Ablation experiment (RQ2)

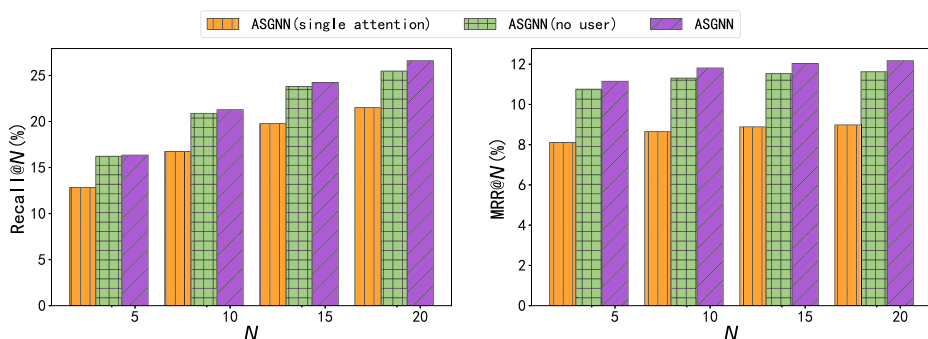
In this section, we evaluate the proposed model ASGNN against its two variants, ASGNN(no user) and ASGNN(single attention). Specifically, ASGNN(no user) does not incorporate user's personalized preferences into attention mechanism. Besides, compared with ASGNN, ASGNN(single attention) uses single layer attention network instead of hierarchical personalized attention network to capture user's preferences. The results are shown in Figure 5, and we have the following two observations.

Firstly, we can see that ASGNN performs better than its two variants on all three datasets, i.e., Gowalla, Foursquare and Brightkite. The results show that the ASGNN(single attention) method has the lowest performance on all three datasets. Specifically, compared with ASGNN, ASGNN(single attention) only uses the single-layer attention network and cannot accurately capture user preferences. Moreover, the linear combination strategy in ASGNN(single attention) ignores that long- and short-term preferences may play different roles in recommendation for each user, which also influences the performance. The results show that the Personalized Hierarchical Attention Network (PHAN) in ASGNN can accurately capture user's preferences and leverage them adaptively for improving the recommendation performance.

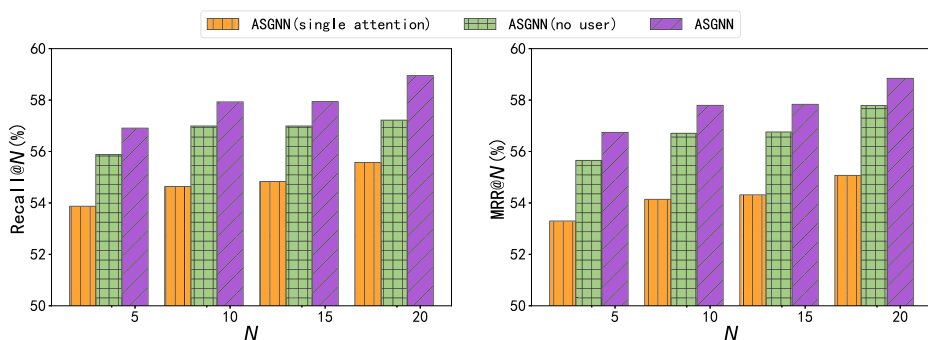
Secondly, we observe that ASGNN(no user) performs better than ASGNN(single attention) on all three datasets, although it is not as effective as ASGNN. The reason is that ASGNN(single attention) does not explicitly incorporate user's preferences, and it cannot



(a) Gowalla dataset



(b) Foursquare dataset



(c) Brightkite dataset

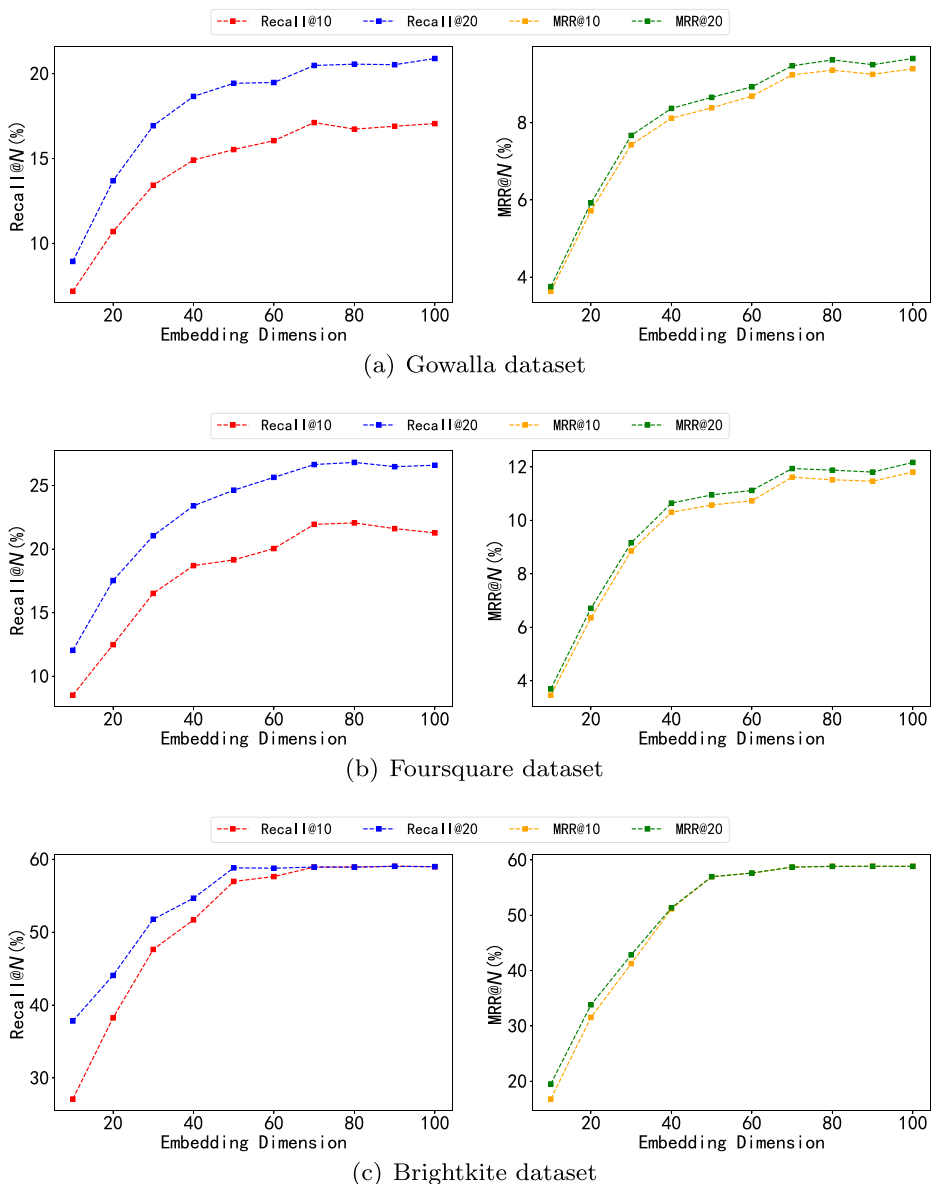
**Figure 5** Results of ablation experiments. ASGNN(no user) refers to removing the personalized recommendation module, and ASGNN(single attention) refers to removing the Hierarchical Attention network

provide personalized recommendation results for different users especially when they have the same or similar check-in sequences. Therefore, the PHAN component of ASGNN plays a very important role in improving the recommendation performance.

In conclusion, the key component of ASGNN, i.e., Personalized Hierarchical Attention Network, enables it effectively capture user's personalized long- and short-term preferences, and leverage them in an adaptive way for accurate recommendation.

### 5.4 Dimension analysis (RQ3)

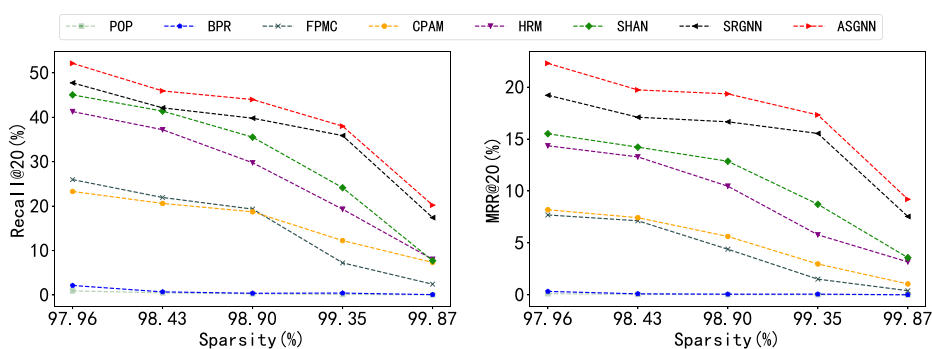
The dimension of the embeddings influences the model's ability of fitting and modeling datasets. Generally, ASGNN with higher dimensional embeddings can depict more useful information of users and POIs, and may have better performance in recommendation task. On the other hand, embeddings with higher dimension may influence the model's efficiency and cause problem of overfitting. Therefore, we set different dimensions in embedding



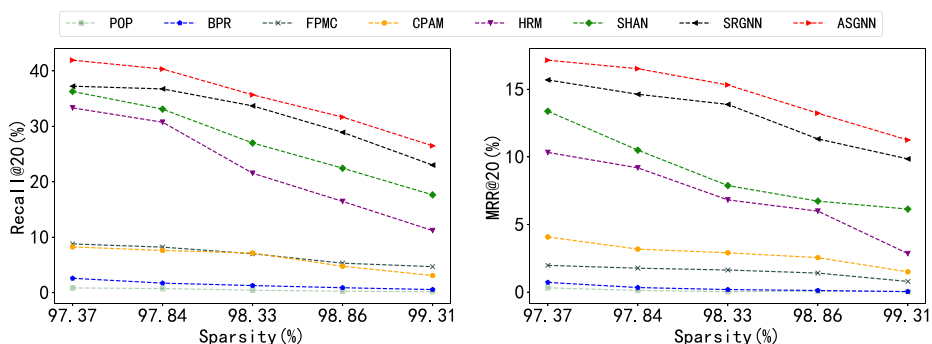
**Figure 6** Effect of different embedding dimensions on next POI recommendation task

layers, which increases from 10 to 100, to investigate how the dimension influences recommendation performance. The experimental results on Gowalla dataset, Foursquare dataset and Brightkite datasets are shown in Figure 6a and b, and c, respectively.

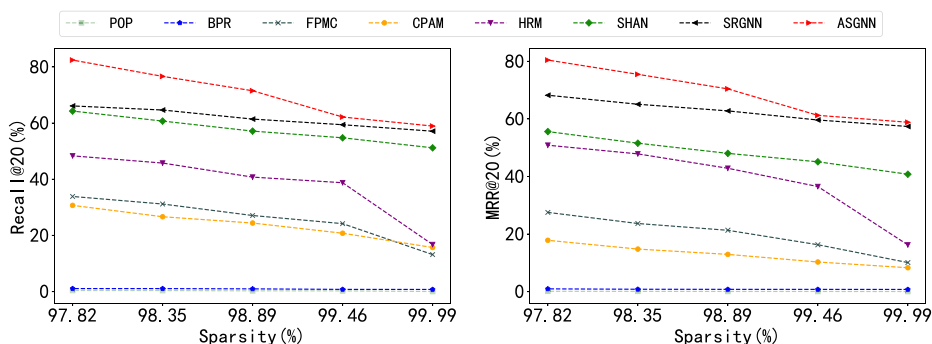
We can observe that the recall@20 and MRR@20 increase for all three datasets as the embedding dimension varies from 10 to 80. The reason is that embeddings with higher dimension can capture more important information for POI recommendation task at the cost of more computation resources and time. Besides, ASGNN with a low dimension (such



(a) Gowalla datasets



(b) Foursquare datasets



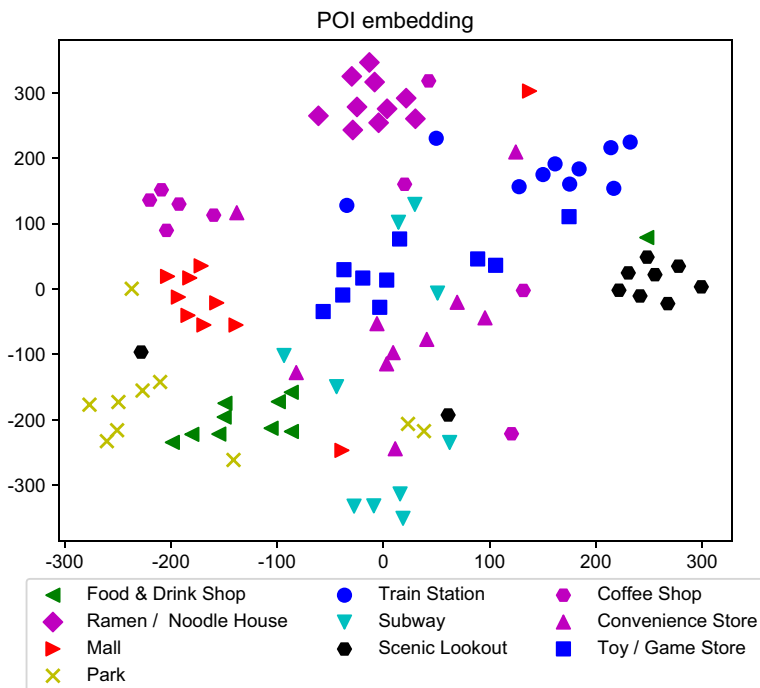
(c) Brightkite datasets

**Figure 7** Influence of data sparsity on next POI recommendation task

as 50) achieves a good performance as well. The reason is that the attention mechanism and GNN improve ASGNN's capacity of capturing relevant features of users and POIs adaptively. ASGNN performs the best on three datasets when the dimension reaches around 80. Moreover, we can see that the recall@10 even decreases slightly on Foursquare dataset when the dimension reaches 100, which may be caused by the over-fitting problem that weakens the generalization ability of ASGNN. Therefore, in subsequent experiments, we set the dimension of the embedding as 80.

### 5.5 Influence of data sparsity (RQ4)

In this section, we use the Gowalla, Foursquare and Brightkite datasets with different sparsity to explore how the sparsity of the data influences the performance of the proposed model ASGNN and baselines. Specifically, these datasets with different sparsity are generated by filtering the POIs with low frequency. In order to make the sparsity gap among the generated datasets change uniformly, we set the frequency thresholds as [5, 20, 30, 40, 50] on the Gowalla dataset, respectively. Specifically, the corresponding sparsity degrees of the Gowalla datasets are [99.87%, 99.35%, 98.90%, 98.43%, 97.96%]. As for the Foursquare dataset, we set the frequency thresholds as [5, 10, 16, 22, 27], and the corresponding sparsity degrees of the Foursquare datasets are [99.31%, 98.86%, 98.33%, 97.84%, 97.37%]. As for the Brightkite dataset, we set the frequency thresholds as [2, 5, 10, 17, 23], and the corresponding sparsity degrees of the Foursquare datasets are [99.99%, 99.46%, 98.89%, 98.35%, 97.82%]. The experimental results on Gowalla, Foursquare and Brightkite datasets are shown in Figure 7a and b and c, respectively.



**Figure 8** Visual illustration of POI embedding (RQ5)

It is intuitive to observe that the recall and MRR of ASGNN and baselines decrease as the sparsity increases. Besides, the proposed model ASGNN outperforms all baselines in terms of recall and MRR on all three datasets with different sparsity. The reason is that ASGNN can fully exploit the check-in sequences to capture and leverage user's preferences and their behavior patterns in a more effective and adaptive way. In conclusion, the results show ASGNN can effectively deal with datasets with different sparsity.

## 5.6 Visual illustration of embedding (RQ5)

In this section, we use the t-SNE [35] method to visualize the POI embedding learned by GNN component of ASGNN. The experimental results are shown in the Figure 8. Note that only Foursquare dataset has the category label of POI, so we illustrate the embeddings learned by ASGNN on Foursquare dataset.

In Figure 8, we can see that POIs with the same label closely cluster in the 2-D visual space. This proves that the GNN component of the proposed approach ASGNN can effectively capture the important characteristics of POIs from the user's check-in data for accurate POI recommendation. Besides, the visualization results show the feasibility of applying the learned embeddings in various tasks, such as data visualization, POI tagging, POI retrieval, and POI clustering.

## 6 Conclusion and future works

In this paper, we propose an Attentive Sequential model based on Graph Neural Network (ASGNN) for next POI recommendation. Specifically, ASGNN consists of four main steps: 1) POI check-in sequence graph construction, 2) feature representation learning, 3) long/short-term preference capturing, and 4) POI recommendation. This work differs from previous work in two main aspects: 1) ASGNN adopts GNN to model user's check-in sequences and their personalized behavior patterns in an effective way; 2) ASGNN uses the PHAN to capture and leverage user's long- and short-term preferences adaptively for improving recommendation performance. Comprehensive experiments are conducted on three real-world POI check-in datasets, and the results show that ASGNN outperforms state-of-the-art baselines in next POI recommendation task. Especially, the PHAN component of ASGNN is shown to be effective in capturing user's behavior patterns and preferences for accurate POI recommendation. In the future, we plan to utilize heterogeneous GNN [45] to fully explore various kinds of auxiliary/side information, such as temporal contexts, metadata, sequential correlations and so on, to further improve the recommendation performance.

## References

1. Albadvi, A., Shahbazi, M.: A hybrid recommendation technique based on product category attributes. *Expert Syst. Appl.* **36**(9), 11,480–11,488 (2009)
2. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. [arXiv:1409.0473](https://arxiv.org/abs/1409.0473) (2014)
3. Bansal, T., Belanger, D., McCallum, A.: Ask the gru: Multi-task learning for deep text recommendations. In: *Proceedings of the 10th ACM Conference on Recommender Systems*, pp. 107–114 (2016)
4. Berg, R.v.d., Kipf, T.N., Welling, M.: Graph convolutional matrix completion. [arXiv:1706.02263](https://arxiv.org/abs/1706.02263) (2017)


5. Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv:[1406.1078](#) (2014)
6. Covington, P., Adams, J., Sargin, E.: Deep neural networks for youtube recommendations. In: Proceedings of the 10th ACM conference on recommender systems, pp. 191–198 (2016)
7. Deng, S., Wang, D., Li, Y., Cao, B., Yin, J., Wu, Z., Zhou, M.: A recommendation system to facilitate business process modeling. *IEEE Trans. Cybern.* **47**(6), 1380–1394 (2016)
8. Elkahky, A.M., Song, Y., He, X.: A multi-view deep learning approach for cross domain user modeling in recommendation systems. In: Proceedings of the 24th International Conference on World Wide Web, pp. 278–288 (2015)
9. Goldberg, D., Nichols, D., Oki, B.M., Terry, D.: Using collaborative filtering to weave an information tapestry. *Commun. ACM* **35**(12), 61–70 (1992)
10. He, R., McAuley, J.: Fusing Similarity Models with Markov Chains for Sparse Sequential Recommendation. In: 2016 IEEE 16Th International Conference on Data Mining (ICDM), pp. 191–200. IEEE (2016)
11. Hidasi, B., Karatzoglou, A., Baltrunas, L., Tikk, D.: Session-based recommendations with recurrent neural networks. arXiv:[1511.06939](#) (2015)
12. Hsieh, C.K., Yang, L., Cui, Y., Lin, T.Y., Belongie, S., Estrin, D.: Collaborative metric learning. In: Proceedings of the 26th international conference on world wide Web, pp. 193–201 (2017)
13. Huang, L., Ma, Y., Wang, S., Liu, Y.: An attention-based spatiotemporal lstm network for next poi recommendation. *IEEE Transactions on Services Computing* (2019)
14. Jannach, D., Ludewig, M.: When recurrent neural networks meet the neighborhood for session-based recommendation. In: Proceedings of the Eleventh ACM Conference on Recommender Systems, pp. 306–310 (2017)
15. Jia, Z., Yang, Y., Gao, W., Chen, X.: User-Based Collaborative Filtering for Tourist Attraction Recommendations. In: 2015 IEEE International Conference on Computational Intelligence & Communication Technology, pp. 22–25. IEEE (2015)
16. Jiang, M., Fang, Y., Xie, H., Chong, J., Meng, M.: User click prediction for personalized job recommendation. *World Wide Web* **22**(1), 325–345 (2019)
17. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv:[1412.6980](#) (2014)
18. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. *Computer* **42**(8), 30–37 (2009)
19. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. *Nature* **521**(7553), 436–444 (2015)
20. Lekakos, G., Caravelas, P.: A hybrid approach for movie recommendation. *Multimed. Tools Appl.* **36**(1–2), 55–70 (2008)
21. Li, Y., Tarlow, D., Brockschmidt, M., Zemel, R.: Gated graph sequence neural networks. arXiv:[1511.05493](#) (2015)
22. Lian, D., Zhao, C., Xie, X., Sun, G., Chen, E., Rui, Y.: Geomf: joint geographical modeling and matrix factorization for point-of-interest recommendation. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 831–840 (2014)
23. Linden, G., Smith, B., York, J.: Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet Comput.* **7**(1), 76–80 (2003)
24. Ma, C., Zhang, Y., Wang, Q., Liu, X.: Point-of-interest recommendation: Exploiting self-attentive autoencoders with neighbor-aware influence. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pp. 697–706 (2018)
25. Mnih, V., Heess, N., Graves, A., et al.: Recurrent Models of Visual Attention. In: Advances in Neural Information Processing Systems, pp. 2204–2212 (2014)
26. Pazzani, M.J., Billsus, D.: Content-Based Recommendation Systems. In: The Adaptive Web, pp. 325–341. Springer (2007)
27. Rendle, S., Freudenthaler, C., Schmidt-Thieme, L.: Factorizing personalized markov chains for next-basket recommendation. In: Proceedings of the 19th international conference on World wide web, pp. 811–820 (2010)
28. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: Bpr: Bayesian personalized ranking from implicit feedback. arXiv:[1205.2618](#) (2012)
29. Resnick, P., Varian, H.R.: Recommender systems. *Commun. ACM* **40**(3), 56–58 (1997)
30. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. *Nature* **323**(6088), 533–536 (1986)
31. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th international conference on World Wide Web, pp. 285–295 (2001)



32. Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G.: The graph neural network model. *IEEE Trans. Neural Netw.* **20**(1), 61–80 (2008)
33. Song, W., Xiao, Z., Wang, Y., Charlin, L., Zhang, M., Tang, J.: Session-based social recommendation via dynamic graph attention networks. In: *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pp. 555–563 (2019)
34. Tang, J., Wang, K.: Personalized top-n sequential recommendation via convolutional sequence embedding. In: *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pp. 565–573 (2018)
35. Van der Maaten, L., Hinton, G.: Visualizing data using t-sne. *J. Mach. Learn. Res.* **9**(11) (2008)
36. Wang, P., Guo, J., Lan, Y., Xu, J., Wan, S., Cheng, X.: Learning hierarchical representation model for nextbasket recommendation. In: *Proceedings of the 38th International ACM SIGIR conference on Research and Development in Information Retrieval*, pp. 403–412 (2015)
37. Wang, D., Deng, S., Zhang, X., Xu, G.: Learning to embed music and metadata for context-aware music recommendation. *World Wide Web* **21**(5), 1399–1423 (2018)
38. Wang, X., He, X., Cao, Y., Liu, M., Chua, T.S.: Kgat: Knowledge graph attention network for recommendation. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 950–958 (2019)
39. Wang, X., He, X., Wang, M., Feng, F., Chua, T.S.: Neural graph collaborative filtering. In: *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval*, pp. 165–174 (2019)
40. Wu, S., Tang, Y., Zhu, Y., Wang, L., Xie, X., Tan, T.: Session-based recommendation with graph neural networks. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 346–353 (2019)
41. Yin, H., Wang, W., Wang, H., Chen, L., Zhou, X.: Spatial-aware hierarchical collaborative deep learning for poi recommendation. *IEEE Trans. Knowl. Data Eng.* **29**(11), 2537–2551 (2017)
42. Ying, H., Zhuang, F., Zhang, F., Liu, Y., Xu, G., Xie, X., Xiong, H., Wu, J.: Sequential Recommender System Based on Hierarchical Attention Network. In: *IJCAI International Joint Conference on Artificial Intelligence* (2018)
43. Ying, R., He, R., Chen, K., Eksombatchai, P., Hamilton, W.L., Leskovec, J.: Graph convolutional neural networks for web-scale recommender systems. In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 974–983 (2018)
44. Yu, D., Wanyan, W., Wang, D.: Leveraging contextual influence and user preferences for point-of-interest recommendation. *Multimed. Tools Appl.*, 1–15 (2020)
45. Zhang, C., Song, D., Huang, C., Swami, A., Chawla, N.V.: Heterogeneous graph neural network. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 793–803 (2019)
46. Zhao, P., Zhu, H., Liu, Y., Li, Z., Xu, J., Sheng, V.S.: Where to go next: A spatio-temporal lstm model for next poi recommendation. [arXiv:1806.06671](https://arxiv.org/abs/1806.06671) (2018)

**Publisher's note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

## Affiliations

**Dongjing Wang<sup>1</sup> · Xingliang Wang<sup>1</sup> · Zhengzhe Xiang<sup>2</sup> · Dongjin Yu<sup>1</sup>  · Shuiguang Deng<sup>3</sup> · Guandong Xu<sup>4</sup>**

Dongjing Wang  
dongjing.wang@hdu.edu.cn

Xingliang Wang  
wangxingliang@hdu.edu.cn

Zhengzhe Xiang  
xiangzz@zucc.edu.cn

Shuiguang Deng  
dengsg@zju.edu.cn

Guandong Xu  
Guandong.xu@uts.edu.au

- <sup>1</sup> School of Computer Science and Technology, Hangzhou Dianzi University, 310018, Hangzhou, China
- <sup>2</sup> School of Computer & Computing Science, Zhejiang University City College, Zhejiang, China
- <sup>3</sup> College of Computer Science and Technology, Zhejiang University, Zhejiang, China
- <sup>4</sup> Advanced Analytics Institute, University of Technology, Sydney, Australia