POI Recommendation for Occasional Groups Based on Hybrid Graph Neural Networks

Lingqiang Meng[a], Zhizhong Liu[a,*], Dianhui Chu[b], Qaun Z. Sheng[c], Jian Yu[d] and Xiaoyu Song[a]

[a]School of Computer and Control Engineering, Yantai University, Yantai, 264005, China
[b]College of Computer Science and Technology, Harbin Institute of Technology, Weihai, 264209, China
[c]School of Computing, Macquarie University, Sydney, NSW 2109, Australia
[d]Department of Computer Science, Auckland University of Technology, Auckland 1142, New Zealand

ARTICLE INFO

Keywords:
POI recommendation
Occasional group
Social influence
Graph neural networks
POI interaction preferences
POI transfer preferences

ABSTRACT

Recently, POI (Point-of-interest) recommendation for groups has become a critical challenge when helping groups to discover potentially interesting new places, and some effective recommendation models have been proposed to address this issue. However, most existing research focuses on POI recommendation for fixed groups, few studies have been conducted on POI recommendation for occasional groups. To tackle this issue, we propose a POI recommendation model for occasional groups based on Hybrid Graph Neural Networks (termed as PROG-HGNN) which combines excellent graph neural networks models. Firstly, PROG-HGNN generates the fitted representation of the occasional group based on the Node Influence Indicator (INF) method and Graph Attention Networks (GAT) model. Then, PROG-HGNN learns POIs' representations containing members' POI interaction preferences and members' POI transfer preferences with the Signed Bipartite Graph Neural Networks (SBGNN) model and the Session-based Graph Neural Networks (SRGNN) model, respectively. Finally, PROG-HGNN recommends the potential POIs for the occasional group based on the fitted representation of the occasional group and the learned representations of POIs. We verify our proposed model on three public benchmark datasets (Foursquare, Gowalla and Yelp), which contains 124,933 to 860,888 POI check-in records. The comparison between our proposed model and the twelve baseline models demonstrates the outstanding performance of PROG-HGNN. In terms of Precision@K and Recall@K, our model achieves about 32.92% and 19.67% improvement compared with the best baseline models on the three benchmark datasets averagely. Adequate ablation experiments prove the effectiveness of the members' POI interaction preferences learning module and POI transfer preferences learning module.

1. Introduction

As one of the important services of Location-based Social Networks (LBSN), POI (Point-of-interest) recommendation devotes to provide users with intelligent location services based on users' preferences (Zhao et al., 2022), which has the advantages of alleviating information overload, enhancing users' experience and bringing service providers more profits (Wang et al., 2022; Chen et al., 2022). Traditional POI recommendation methods usually utilizes collaborative filtering (Cai et al., 2022; Liu et al., 2022a) and matrix decomposition (Ren et al., 2017; Davtalab and Alesheikh, 2021) to perform POI recommendation for users. However, these algorithms failed to model deeper interaction between users and POIs. Recently, artificial neural networks (Esen et al., 2008a) and deep learning (Esen et al., 2008c) have shown great power in modeling implicit interaction between different entities, and have been applied in many fields, such as NLP (Goossens et al., 2023), forecast (Esen et al., 2009, 2017), computer vision (Kuru et al., 2023), recommendation techniques (Liu et al., 2023) and others (Esen et al., 2008b). In this end, many effective POI recommendation models have been developed based on deep learning (Li et al., 2022; Liu et al., 2022b) and graph neural networks (Zhang et al., 2022).

Recently, with the popularity of group activities in people's routine work and social communication, POI group recommendation becomes increasingly significant, which aims to pursue a list of POIs for a target group while satisfying each member's preference (Bahari Sojahrood and Taleai, 2022; Liu et al., 2021; Zhao et al., 2020;

Email addresses: 1782356223@qq.com (Lingqiang Meng), lzzmff@126.com (Zhizhong Liu), cdh@hitwh.edu.cn (Dianhui Chu), michael.sheng@mq.edu.au (Qaun Z. Sheng), jian.yu@aut.ac.nz (Jian Yu), sxyytdxjkxy@gmail.com (Xiaoyu Song)
*Corresponding author.
ORCID(s):

Sojahrood and Taleai, 2021). Actually, there are two kinds of groups: fixed groups and occasional groups (Khazaei and Alimohammadi, 2018). A fixed group is made up of several users who explicitly decide to stay in the group for a long time, due to the similar and stable interests. While an occasional group refers to a group of users who gather together temporarily for a specific event, and the group will dissolve automatically when the purpose is achieved.

Currently, most POI group recommendation research focuses on POI recommendation for fixed groups (Bahari Sojahrood and Taleai, 2022; Sojahrood and Taleai, 2021), few studies have been carried out on POI recommendation for occasional groups. Moreover, in our daily lives, more and more occasional groups are appearing in many kinds of social activities (such as meeting, tourism, conferences, etc.). Let's consider a scenario that a group of scholars from the United States are attending an academic conference. In this scenario, all the attendees constitute an occasional group. Usually, to liven up the atmosphere and promote interpersonal communication, conference organizers always selecting some Point-of-interests (POIs) for the occasional group to visit. In this scenario, the conference organizer faces the problem of POI recommendation for the occasional group. Undoubtedly, like the above example, the problem of POI recommendation for occasional groups is common in many fields (such as business, education, tourism, and so on).

Figure 1: An example of POI recommendation for occasional groups.

Compared with fixed groups, occasional groups typically have the following characteristics: (1) Occasional groups are formed randomly and exist for a short period of time, thus, the POI interaction data of occasional groups is sparse. Moreover, in an occasional group, not all members have social relationships with each other. That is, some members may have social relationships with some members, while some members may not have social relationships with other members. (2) In an occasional group, members may have different preferences and different social influences. Actually, when recommending POIs for occasional groups, it is necessary to consider the preferences of the attendees who have greater social influences. Due to the significant differences between occasional groups and fixed groups, existing POI recommendation methods for fixed groups cannot be used for POI recommendation for occasional groups. Therefore, how to effectively realize POI recommendation for occasional groups becomes a critical problem to be solved urgently. To solve this problem, there are some challenges to be tackled:

- **How to overcome the data spare problem**. Since occasional groups usually exist for a short period of time, thus, the POIs interaction data of an occasional group is relatively sparse. This will bring difficulties for model's learning. How to overcome the data spare problem is the first challenge to be tackled.
- **How to generate the fitted representation of an occasional group**. For an occasional group, members usually have different social influences, which has a significant impact on the recommendation of occasional groups. Therefore, it is necessary to consider members' social influences when generating the fitted representation of

an occasional group. However, existing works usually apply predefined aggregation strategies to generate the representation of the groups, and do not consider members' social influences. Thus, how to generate the fitted representation of an occasional group is worth investigating.

- **How to learn members' POI interaction preferences comprehensively**. In an occasional group, members' POI interaction preferences have great impact on POI recommendation for occasional groups. Actually, members' POI interaction preferences includes positive preferences and negative preferences. However, existing methods only learn members' positive POI interaction preferences, and failed to learn members' negative POI interaction preferences (Li et al., 2022; Zhang et al., 2022), which decreases the effect of POI recommendation. Therefore, how to learn members' POI interaction preferences comprehensively is a challenge to be tackled.

- **How to learn members' POI transfer preferences**. For an occasional group, members' POI transfer data implies members' POI visiting preferences, which is valuable for improving the effect of POI recommendation for occasional groups. However, existing research fails to learn group members' POI transfer preferences when conducting POI recommendation.

To address the above issues, we propose a POI recommendation model for occasional groups based on hybrid graph neural networks (PROG-HGNN). PROG-HGNN works as follows. Firstly, PROG-HGNN adopts the Node Influence Indicator (INF) method to calculate members' social influences, applies the Graph Attention Networks (GAT) to learn members' representations, and then generates the fitted representation of the occasional group based on members' social influences and representations. Secondly, with the Signed Bipartite Graph Neural Networks (SBGNN), PROG-HGNN learns POIs' representations from the POIs interaction bipartite graph, which implies members' POIs interaction preferences. Thirdly, with the Session-based Graph Neural Networks (SRGNN), PROG-HGNN learns POIs' representations from the POIs transfer directed graph, which implies members' POI transfer preferences. Finally, PROG-HGNN recommends the optimal POIs to the occasional group based on the fitted representation of the occasional group and the representations of POIs. The main contributions of this work are as follows:

- We propose a POI recommendation model for occasional groups based on hybrid graph neural networks (PROG-HGNN). We first convert the occasional group into a virtual user by generating the fitted representation of the occasional group, and then train the PROG-HGNN model based on members' POI interaction data, which can overcome the data sparse problem. To the best of our knowledge, this is the first work for POI recommendation for occasional groups.

- We propose a fitted representation generation method for occasional groups based on members' social influences and the members' representations. Firstly, we calculate members' social influences with INF (Huang et al., 2020). Then, we apply GAT (Veličković et al., 2018) to learn members' representations based on members' social network. Finally, we generate the fitted representation of the occasional group by combining members' social influences and members' representations.

- We propose a POI interaction preferences learning module to comprehensively learn members' positive and negative POI interaction preferences. Firstly, we construct a signed interactive bipartite graph about members and POIs based on members' POI interaction data. Then, we apply SBGNN (Huang et al., 2021) to learn POIs' representations which contains members' positive and negative POI interaction preferences.

- We propose a POI transfer preferences learning module to learn members' POI transfer preferences. Firstly, we construct a directed POI transfer graph based on the members' POI transfer data. Then, we adopt SRGNN (Wu et al., 2019) to learn POIs' representations which hints members' POI transfer preferences.

The remainder of this paper is organized as follows. Section 2 discusses the related works. Section 3 presents our research objective and the problem statement. Section 4 elaborates our proposed model. Section 5 reports experimental results and analysis. Section 6 concludes this work and discusses our further research.

2. Related Work

POI group recommendation aims to provide POIs to a group of members based on each member's preference. Compared to personalized POI recommendation, POI group recommendation is much more challenging due to the complexity of groups made up of individuals who have diverse interests and preferences. Recently, some effective POI group recommendation methods for fixed group have been developed. Work (Bahari Sojahrood and Taleai, 2022) developed a POI group recommendation method based on the group's LBSN check-in behavior, which enhances group recommendation results by utilizing a 2D kernel density estimate approach. To further improve the precision of POI group recommendation and considering the intragroup divergence, Liu et al. (Liu et al., 2021) developed a POI group recommendation method based on collaborative filtering. Based on the attentive multitask learning, Chen et al. (Chen et al., 2021) proposed a POI group recommendation method, which uses the attention mechanism to get consensus group preferences while dynamically learning the internal relationships between group members. Sojahrood et al. (Sojahrood and Taleai, 2021) developed a POI group recommendation model that simulates user influence, this work also considered the differences in users' personalities and preferences when they are alone or in a group.

Li et al. (Li et al., 2020) investigated the roles of reviews, POI categories, and geographical locations in POI group recommendation, and proposed a POI group recommendation model based on a group-based temporal sentiment-aspect-region recurrent neural network. Work (Zhu et al., 2018) proposed a context aware POI group recommendation model, which incorporates the significance of location in POI group recommendations and uses distance-based pre-filtering and distance-based ranking modification to enhance POI group recommendations. Sojahrood et al.(Sojahrood et al., 2023) designed a hybrid group recommendation model according to the group types, which changes recommendation methods based on the group type and the members' homogeneity. Zhao et al. (Zhao et al., 2020) regarded POI group recommendation as a binary classification issue, and designed an extreme learning machine (ELM) based POI group recommendation model. Zhang et al.(Zhang et al., 2020b) created a POI recommendation algorithm for groups based on the knowledge of machine learning after converting POI group recommendation into a classification issue.

Although existing research have achieved wonderful results on POI group recommendation, these works focus on recommending POIs for fixed groups, and cannot be used for POI recommendation for occasional groups. To address this problem, we propose a POI recommendation model for occasional groups based on hybrid graph neural networks in this work.

3. Preliminaries and research objective

3.1. Preliminaries

Recently, more and more occasional groups are appearing in many kinds of social activities, which are randomly formed by some users. When recommending POIs for occasional groups, it is necessary to consider useful information of the members. In this section, we give some definitions in POI recommendation for occasional groups.

Definition 1: Users set. Let $U = \{u_1, u_2, \dots, u_m\}$ denote the set of users that belonging to a specific field or a company, and m is the size of the users set. Each user in U is identified by a unique user ID.

Definition 2: POIs set. A POI is defined as a geographic location with specific functions that can meet user needs (such as a coffee shop or a supermarket). A POI has three attributes, which are unique id, category c and geographic information d=($long$, lat), where $long$ and lat denotes the longitude and latitude of a POI, respectively. Let $P = \{p_1, p_2, \dots, p_n\}$ represent the set of POIs, where n denotes the number of POIs.

Definition 3: Occasional Group. An occasional group refers to a group of users (belonging to U) that gather together for a certain activity, and the group will dissolve when the activity is finished. In an occasional group, some members may have social relationships with other members, while others may not have social relationships with other members. Moreover, members in an occasional group usually have different social influences due to their backgrounds and power. In this work, an occasional group is defined as $OG = \{u_i, u_j, \dots, u_r\}$, where r represents the size of the OG, and OG is a subset of U.

Definition 4: Social Networks. Let $GS =< V, E >$ denotes the social network of the members of the occasional group OG, where $V = \{u_1, u_2, \dots, u_m\}$ indicates the nodes of GS, E denotes the edges of GS, where $e_{ij} =< u_i, u_j >$ $\left(i \neq j, e_{ij} \in GS\right)$ indicates that member u_i and member u_j exist a social relationship. It should be noted that, there may be some isolated nodes in GS, since there may be some members of the OG having no social relationships with other members due to they join the group for the first time.

Definition 5: Check-in. Let $S = \{S_i, S_j, \dots, S_r\}$ be the set of POI check-in sequences of all members of the OG, where $S_i = \{s_1, s_2, \dots, s_i\}$ means the set of check-in sequences of the member u_i, and $s_i = \{p_1, \dots, p_k\}$ $(k \in [0, n])$ indicates the i^{th} POI check-in sequence of member u_i.

Problem Statement. Given the users set U, POI set P and users' POI check-in sequence S, assume that OG is an occasional group, GS is the social network of OG. Then, the problem studied in this work is how to recommend an optimal POI list for the occasional group OG which can satisfy members' preferences.

3.2. Research Objective

Based on the above-mentioned definitions, we model some important related information between POIs and users as graphs and apply powerful graph neural networks to learn the useful knowledge. Our research goal is to address the problem of POI recommendation for occasional groups. To realize our research goal, the first objective is to generate the fitted representation of the occasional group and convert the occasional group into a virtual user. For this objective, we propose a fitted representation learning method for generating the fitted representation of the occasional group based on members' social influences and representations. The second objective is to learn members' POI interaction preferences. For this objective, we first construct the signed interactive bipartite graph and then apply SBGNN (Huang et al., 2021) to comprehensively learn the representations of POIs which implies members' POI positive and negative POI interaction preferences. The third objective is to learn the members' POI transfer preferences. For this objective, we first construct the directed POI transfer graphs, and then adopt SRGNN (Wu et al., 2019) to learn the representations of POIs which contains members' POI transfer preferences. The last objective is to perform the POI recommendation based on the fitted representation of the occasional group and POIs' representation.

4. Methodology

To tackle the issue of POI recommendation for occasional groups, we propose a POI group recommendation model for occasional groups based on hybrid graph neural networks (named as PROG-HGNN). The methodology of our work is that, we first turn the occasional group into a virtual user; then we construct the signed interactive bipartite graph and the directed POI transfer graphs according members POI check-in data; next, we use different powerful graph neural networks to learning the POIs' representations; and finally perform POI recommendation based on the learning results.The structure of the PROG-HGNN model is illustrated as Fig. 2. PROG-HGNN includes four modules, which are fitted group representation learning module, members' POI interaction preferences learning module, members' POI transfer preferences learning module, and the prediction module. PROG-HGNN works as follows: firstly, it generates the fitted representation of the occasional group based on INF (Huang et al., 2020) and GAT (Veličković et al., 2018). Then, PROG-HGNN learns POIs' representations including members' POI interaction preferences and members' transfer preferences with SBGNN (Huang et al., 2021) and SRGNN (Wu et al., 2019), respectively. Finally, PROG-HGNN recommends the optimal POIs to the occasional group based on the fitted representation of the occasional group and POIs' representations. We will present the technical details of each module of PROG-HGNN in the rest of this section.

4.1. The Fitted Representation Learning Module

Considering that members in an occasional group usually have different social influences, which have important impact on POI recommendation, we propose a fitted representation learning module based on members' social influences and members' representations. Firstly, we construct a social network graph of the OG based on members' social relationships. Next, we compute members' social influences with INF (Huang et al., 2020), and then learn members' representations with GAT (Veličković et al., 2018). Finally, we generate the fitted representation of the OG by aggregating each member's social influence and representation. The details of the fitted representation learning module is presented as follows.

4.1.1. Constructing members' social network graph

Assume that $OG = \{u_i, u_j, \dots, u_r\}$ denotes an occasional group, $GS =< V, E >$ represents the members' social relationships of the OG, and $e_{ij} =< u_i, u_j >$ $\left(i \neq j, e_{ij} \in GS\right)$ denotes a social relation between member u_i and u_j. Based on GS, we construct the social network graph of OG, which is denoted as $G_s = (H, \mathcal{E})$, where

[0]The group members' POI interaction graph is drawn based on the work Cai et al. (2022)

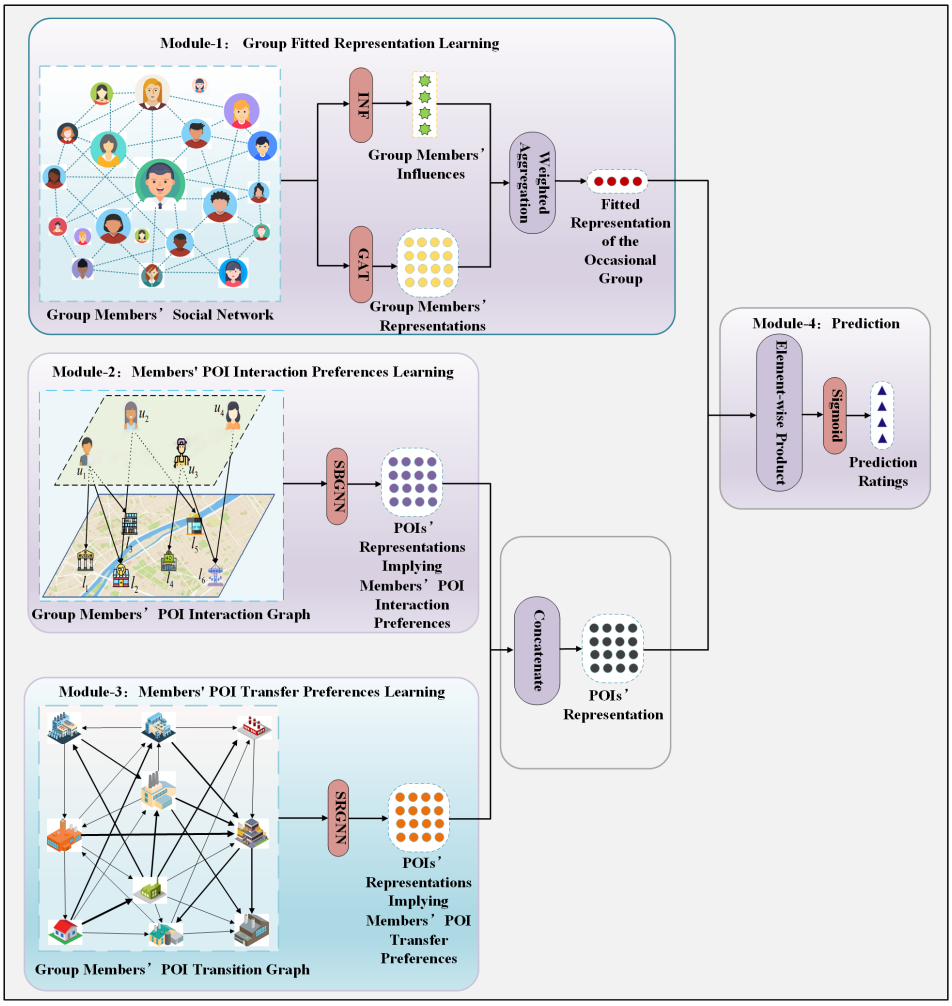


Figure 2: The structure of the PROG-HGNN model.

$H = \left\{\overrightarrow{h_i}, \overrightarrow{h_j}, \ldots, \overrightarrow{h_r}\right\}$ ($\overrightarrow{h_i} \in \mathbb{R}^F$) denotes the set of original representations of members of the OG, F represents the dimension of the representations. $\mathcal{E}$ indicates the set of social edges between members of OG. The social network graph of an OG is illustrated as Fig. 3(a). In Fig. 3(a), the nodes denote the members of the OG, the edges represent social relationships between different members of the OG. It should be noted that, some users in the OG may not have social relationships with other members, thus, there may be some isolated nodes in graph G_s.

Let $A \in \mathbb{R}^{m \times m}$ denote the graph G_s's adjacency matrix (as illustrated in Fig. 3(b)). If there exists a social relationship between member u_i and u_j, then, the value of $a_{i,j}$ is 1 in A. Otherwise, the value of $a_{i,j}$ is 0. Particularly, the element values of the isolated nodes in G_s are set as 0 in the adjacency matrix A. In this work, G_s clearly depicts the social relationships of the members of the OG, which is helpful to learn members' representations and the fitted representation generation of the OG. Besides, G_s is the basis for members' social influences calculation and members' representation learning.

4.1.2. Calculating members' social influences with INF

Currently, existing POI group recommendation research usually takes the number of POI check-in as the influences of members (Sojahrood and Taleai, 2021). However, this kind of methods cannot calculate members' social influences in occasional groups. Node Influence Indicator (INF) (Huang et al., 2020) was proposed to identify the influencers in a social network and always used to detect influential nodes in multi-layer networks. INF considers the local neighbors' information so as to detect the influential nodes in a network quickly and reliably. In this work, we adopt INF to calculate

members' social influences based on the graph G_s. The calculation of INF is defined as Eq.(1):

$$\mathrm{INF}(i) = \sum_{x=1}^{|R|} \sum_{j \in N_x(i)} \frac{d_{ij}^2 \cdot \omega_{ij}}{k_j} \tag{1}$$

where R indicates the social network radius of members. According to work (Huang et al., 2020), we set $R = 1$ for simplicity, which means that only one hoop neighbors are included when calculating members' social influence. $N_x(i)$ is the node i's x_{th} order neighbors, d_{ij} represents the length of the shortest path between node i and j, k_j denotes the degree of node j, and ω_{ij} represents the weight of edge e_{ij}. Since the graph G_s is an undirected graph, so the value of ω_{ij} is set as 1 referring to the parameters setting in work (Huang et al., 2020).

Particularly, for the isolated nodes in graph G_s, since they do not have neighbors and the shortest path. Therefore, INF cannot calculate the social influences of these isolated nodes with Eq.(1). To solve this problem, according to work (Huang et al., 2020), we set the social influences of the isolated nodes in graph G_s as 0.5. Based on the social network graph G_s, according to Eq.(1) and the treating strategy for the isolated nodes, we can calculate each member's social influence in OG, which is recorded as $SI = [SI_i, SI_j, \ldots, SI_r]$.

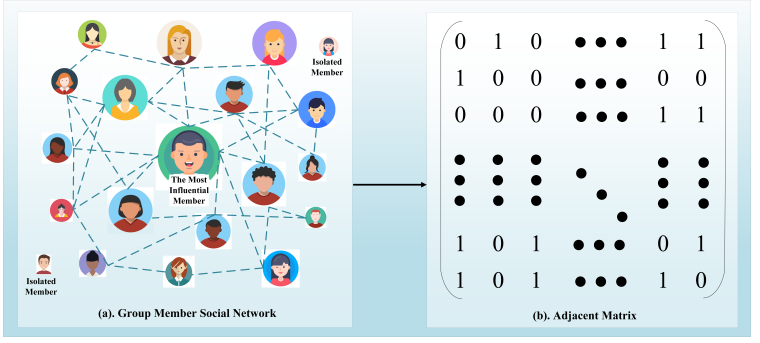


Figure 3: Social network and the adjacency matrix.

4.1.3. Learning members' representations with GAT

Actually, members' representations in a OG are related with their different neighbors in the social network. To learn members' representations better, we adopt a graph neural network GAT (Veličković et al., 2018) to learn members' representations from the G_s of OG. GAT is a variant of GNNs, which combine the attention mechanism with the information propagation, and calculates each node's hidden states according to node's neighbors with a self-attention mechanism. Moreover, GAT utilizes the multi-head attention to make the learning process more stable. Thus, GAT is suitable for learning members' representations. The computing process of members' representation learning is described as follows.

Firstly, the initial representations of nodes in the G_s are initialized randomly. Then, the set of members representation H and adjacency matrix A are imported into GAT. GAT collect the information of neighbors to update target node's representation. Let u_i denote a node in graph G_s, we firstly calculate the attention coefficients α_{ij} with Eq.(2), which indicates the importance of node u_j to node u_i.

$$\alpha_{ij} = \frac{\exp\left(\mathrm{LeakyReLU}\left(\vec{a}^T \left[W\overrightarrow{h_i} \| W\overrightarrow{h_j}\right]\right)\right)}{\sum_{k \in N_i} \exp\left(\mathrm{LeakyReLU}\left(\vec{a}^T \left[W\overrightarrow{h_i} \| W\overrightarrow{h_k}\right]\right)\right)} \tag{2}$$

where $\overrightarrow{h_i}$ and $\overrightarrow{h_j}$ denotes the representation of node u_i and node u_j. $W \in \mathbb{R}^{F' \times F}$ represents the learnable weight matrix, $\|$ indicates the concatenation operation, $\vec{a} \in \mathbb{R}^{2F}$ denotes a trainable weight vector, $\vec{a}^T$ represents vector transposition of $\vec{a}$, $LeakyReLU(\cdot)$ indicates the activation function, N_i is the neighbors of node u_i, and $exp(\cdot)$ means the softmax function to normalize coefficients.

Next, the transformed representations of neighbors nodes are aggregated by GAT with the normalized attention coefficient α_{ij} as the new representation of node u_i, which is illustrated as Eq.(3):

$$\overrightarrow{h_i'} = \sigma\left(\sum_{j\in N_i} \alpha_{ij}\overrightarrow{h_j}\right) \tag{3}$$

To make the GAT's learning process more stable, we integrate multi-head attention mechanism into GAT. That is, the transformation operation (described as Eq.(3)) is executed by K independent attention mechanisms, then, the representations produced by each attention mechanism are concatenated and obtain each node's final representation. This process is described by Eq.(4):

$$\overrightarrow{h_i'} = \|_{k=1}^{K}\sigma\left(\sum_{j\in N_i} \alpha_{ij}^k W^k\overrightarrow{h_j}\right) \tag{4}$$

where $\|$ represents the concatenation operation, α_{ij}^k denotes the normalized attention coefficients which is calculated based on the k-th attention mechanism, W^k denotes the linear transformation's wight matrix. $\overrightarrow{h_i'}$ indicates the group member u_i's final representation. $\overrightarrow{h_i'}$ can accurately capture the social preferences of member u_i by aggregating the information of neighbor nodes, which is beneficial to generate the fitted representation of the OG. Note that, the final output $\overrightarrow{h_i'}$ consists of KF' representations for each node $\overrightarrow{h_i'}$.

Especially, for the isolated nodes in the G_s, since they are lack of neighbors, GAT cannot update their representations during the process of the forward propagation. Thus, the isolated nodes' representations can only be updated in the back propagation process of GAT. After getting the representations of members $U' = [\overrightarrow{h_i'}, \overrightarrow{h_j'}, \ \ldots, \overrightarrow{h_r'}]$, we take members' social influences $SI = [SI_i, SI_j, \ \ldots, SI_r]$ as the weights, we generate the fitted representation of the OG with a weighted sum operation, which is illustrated as Eq.(5):

$$\vec{g} = \sum_{i=0}^{r} SI_i \cdot \overrightarrow{h_i'} \tag{5}$$

where $\vec{g}$ denotes the fitted representation of the OG, which is generated based on members' social influences computed by INF and members' representations learned by GAT.

4.1.4. Loss function of the members' representation learning module

To train the members' representation learning module based on GAT, we adopt the edge prediction of the social network graph G_s as the driving task to activate the training of GAT. Specifically, we first predict the probability which indicates whether there exist a social relationship between two members u_i and u_j based on their representations, then, we calculate the loss of GAT based on the predicted score and the ground truth value, and finally train the GAT by back propagation algorithm according to the loss. Firstly, we calculate the prediction value y_{pred} according to an MLP model. y_{pred} indicates the probability of a social relationship existed between two group members and is calculated with Eq.(6):

$$y_{\text{pred}} = \text{sigmoid}\left(\text{MLP}\left(\vec{h}_i \| \vec{h}_j\right)\right) \tag{6}$$

where MLP denotes a neural network model which is stacked by two neural layers, $\|$ is used to concatenate vectors together, $\vec{h}_i$ and $\vec{h}_j$ denotes the representations of group members u_i and u_j, respectively. y_{pred} is a probability value in the range of $(0, 1)$ which represents the possibility of having a social relationship between member u_i and u_j. Then, the binary cross entropy is selected as the loss function, which is defined as Eq.(7):

$$\mathcal{L}\left(y_{\text{pred}}\right) = y_{\text{true}} \cdot \log y_{\text{pred}} + (1 - y_{\text{true}}) \cdot \log\left(1 - y_{\text{pred}}\right) \tag{7}$$

where y_{true} denotes the binary ground truth in the range of $\{0, 1\}$. After obtaining the loss $\mathcal{L}\left(y_{\text{pred}}\right)$, the GAT is trained with the back propagation mechanism according to the calculated $\mathcal{L}\left(y_{\text{pred}}\right)$.

4.2. Members' POI Interaction Preferences Learning Module

To learn members' POI interaction preferences comprehensively, in this work, we first establish a signed interactive bipartite graph according to members' POI interaction data, which can clearly depict the members' positive and negative POI interaction information. Then, we adopt the SBGNN (Huang et al., 2021) to comprehensively learn the representations of POIs which contain members' positive and negative POI interaction preferences. In the following sections, we first present the construction of the signed interactive bipartite graph, and then introduce the process of group members' POIs interaction preferences learning.

4.2.1. Constructing the signed interactive bipartite graph

Assume that $OG = \{u_i, u_j, \dots, u_r\}$ is an occasional group, $P_G = \{p_1, p_2, \dots, p_k\}$, $P_G \subseteq P$ denotes the set of POIs that members of the OG has visited, and $I =< u_i, p_j > (u_i \in OG, p_j \in P_G)$ represents POI interaction records of members. The signed interactive bipartite graph is constructed as follows: If u_i has visited p_j, a positive edge is established between u_i and p_j; Otherwise, a negative edge is established between them. If u_i and u_m have links with same sign on p_j, a positive edge is constructed between u_i and u_m; Otherwise, a negative edge is constructed between these two members. Similarly, when u_i has links with same sign on p_k and p_f, a positive edge is constructed between p_k and p_f; Otherwise, a negative edge is constructed between these two POIs. According to the above rules, we can construct the signed interactive bipartite graph which is illustrated as Fig. 4, where black and red edges represents positive and negative edges, respectively. Let $G_I = (OG, P_G, E_G)$ denote the signed interactive bipartite graph, where OG and P_G represents the sets of members and POIs, and E_G refers to the set of connecting edges between nodes in G_I, $E_G = \varepsilon^+ + \varepsilon^-$ and $\varepsilon^+ \cap \varepsilon^- = \varnothing$, where ε^+ and ε^- denotes positive and negative edges, respectively.

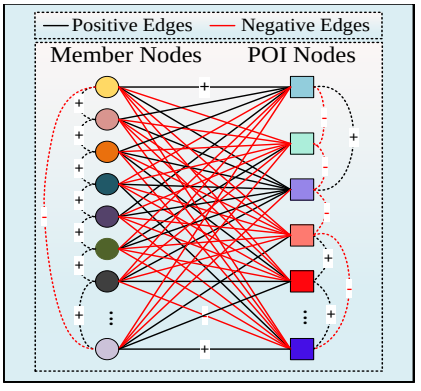


Figure 4: Signed interactive bipartite graph.

4.2.2. Learning members' POI interaction preferences

Specifically, we take the G_I as the input of SBGNN (Huang et al., 2021), and the representations of POIs will be learned by aggregating information from positive and negative neighbors through message propagation, message aggregation and updating mechanisms of SBGNN. The details are described as follows.

(1) ***Message propagation***. Let Set_1 denote the set of nodes including both members and POIs, and Set_2 denote the set of nodes with the same type (such as only members set or POIs set). For the l^{th} layer of SBGNN, $W_p^l \rightarrow +u$ and $W_p^l \rightarrow -u$ are used to propagate the messages from p_j to u_i based on the positive and negative edges. The message propagation process can be illustrated as Eq.(8):

$$\begin{aligned} m^l_{p\rightarrow^+u}\left(p_j, u_i\right) &= \mathrm{MSG}\left(h^l_{u_i}, h^l_{p_j}\right) = W^l_{p\rightarrow^+u} \cdot h^l_{p_j} \\ m^l_{p\rightarrow^-u}\left(p_j, u_i\right) &= \mathrm{MSG}\left(h^l_{u_i}, h^l_{p_j}\right) = W^l_{p\rightarrow^-u} \cdot h^l_{p_j} \end{aligned} \tag{8}$$

where $N_{p\rightarrow+u}\left(u_i\right)$ and $N_{p\rightarrow-u}\left(u_i\right)$ denotes the neighbors based on the positive and negative edges to u_i, $p_j \in N_{p\rightarrow+u}\left(u_i\right)$ and $p_j \in N_{p\rightarrow-u}\left(u_i\right)$. Similarly, we use $W_u^l \rightarrow +p$ and $W_u^l \rightarrow -p$ to propagate the message from u_j

to p_i, and the message propagation process is described as Eq.(9):

$$\begin{aligned} m^l_{u\to^+p}\left(u_j,p_i\right) &= \mathrm{MSG}\left(h^l_{p_i},h^l_{u_j}\right) = W^l_{u\to^+p}\cdot h^l_{u_j} \\ m^l_{u\to^-p}\left(u_j,p_i\right) &= \mathrm{MSG}\left(h^l_{p_i},h^l_{u_j}\right) = W^l_{u\to^-p}\cdot h^l_{u_j} \end{aligned} \tag{9}$$

where $N_{u\to^+p}\left(p_i\right)$ and $N_{u\to^-p}\left(p_i\right)$ represents the neighbors with positive and negative edges to p_i, $u_j \in N_{u\to^+p}\left(p_i\right)$ and $u_j \in N_{p\to^-p}\left(p_i\right)$.

Since here only the representations of POIs are used, so Set_2 refers to the set of POIs. Therefore, we only propagate messages for the positive and negative edges from p_j to p_i with $W^l_p \to +p$ and $W^l_p \to -p$, respectively. The messaging propagation formula for Set_2 is defined as Eq.(10):

$$\begin{aligned} m^l_{p\to^+p}\left(p_j,p_i\right) &= MSG\left(h^l_{p_j},h^l_{p_i}\right) = W^l_{p\to^+p}\cdot h^l_{p_j} \\ m^l_{p\to^-p}\left(p_j,p_i\right) &= MSG\left(h^l_{p_j},h^l_{p_i}\right) = W^l_{p\to^-p}\cdot h^l_{p_j}, \end{aligned} \tag{10}$$

where $N_{p\to^+p}\left(p_i\right)$ and $N_{p\to^-p}\left(p_i\right)$ represents the positive and negative neighbors for p_i, respectively. $p_j \in N_{p\to^+p}\left(p_i\right)$ and $p_j \in N_{p\to^-p}\left(p_i\right)$.

(2) ***Message aggregation***. The message aggregation is performed after obtaining the message of the neighbor nodes. SBGNN uses a graph attention function to aggregate information from the neighbor nodes. The weight coefficient between the two nodes is calculated, which reflects their correlation. The details can be described by Eq.(11):

$$\alpha_{ij} = \frac{\exp\left(\text{Leaky ReLU}\left(\vec{a}^T\left[W\vec{h}_i \| W\vec{h}_j\right]\right)\right)}{\sum_{k\in N_i}\exp\left(\text{Leaky Re}\,LU\left(\vec{a}^T\left[W\vec{h}_i \| W\vec{h}_k\right]\right)\right)} \tag{11}$$

where $\vec{h}_i$ represents the representation of node i, $\|$ denotes the concatenation operation, W means the weight matrix, $\vec{a}$ indicates the learnable parameter vector. N_i refers to the set of neighbors of node i. $LeakyReLU(\cdot)$ is the activation function. $\exp(\cdot)$ represents the softmax function, which is used to normalize the weight coefficients. Then, the messages from neighbors are aggregated based on the weight coefficient α_{ij}, which is described as Eq.(12):

$$\begin{aligned} m^l\left(u_i\right) &= \sum_{j\in N_i}\alpha^{ij}m^l\left(u_j\right) \\ m^l\left(p_i\right) &= \sum_{j\in N_i}\alpha^{ij}m^l\left(p_j\right) \end{aligned} \tag{12}$$

where N_i represents the set of neighbors of node i. We can convert the attention aggregation into a function with learnable weighted average mechanism.

(3) ***Updating representations***. After message propagation and aggregation, each u_i obtains four groups of message sets from neighborhoods, namely $m^l_p \to +u$, $m^l_p \to -u$, $m^l_u \to +u$ and $m^l_u \to -u$. Each p_i also has four sets of messages from neighborhoods, that is $m^l_u \to +p$, $m^l_u \to -p$, $m^l_p \to +p$ and $m^l_p \to -p$. Then, SBGNN aggregates the information of the four groups of neighbor nodes onto node i and obtains the final representation of node i through an MLP. The above process can be detailed as Eq.(13):

$$\begin{aligned} h^{l+1}_u &= MLP\left(h^l_u \left\| m^l_{p\to^+u}\right\| m^l_{p\to-}\left\| m^l_{u\to^+u}\right\| m^l_{u\to-u}\right) \\ h^{l+1}_p &= MLP\left(h^l_p \left\| m^l_{u\to^+p}\right\| m^l_{u\to^-p}\left\| m^l_{p\to^+p}\right\| m^l_{p\to-p}\right) \end{aligned} \tag{13}$$

where h^l_u denotes the representation of members, h^l_p denotes the representation of POIs, $\|$ represents the concatenation operation and MLP denotes a neural network model stacked by two fully connected neural layers with a dropout ratio and activation mechanism. The MLP is defined as Eq.(14):

$$\mathrm{MLP}(x) = W_2\left(\sigma\left(\text{dropout}\left(W_1x + b_1\right)\right)\right) + b_2 \tag{14}$$

where W_1, W_2 and b_1, b_2 denotes the learnable parameters of the MLP, σ is an activation function, $dropout$ is used to avoid over-fitting.

The output sequences of SBGNN are denoted as $Z = [\vec{z}_1^p, \vec{z}_2^p, \ldots, \vec{z}_k^p]$, where $\vec{z}_i^p$ represents the representation of the POI p_i, which imply members' positive and negative POI interaction preferences for p_i.

4.2.3. Loss function of members' interaction preferences learning module

To train the members' POI interaction preferences learning module based on SBGNN, we take the edge prediction of G_I as the driving task to activate the training of SBGNN. Specifically, we first predict the probability which indicates whether there exist an interaction edge between member u_i and POI p_j based on their representations. Then, we calculate the loss of SBGNN based on the predicted score and the ground truth value. Finally, train the SBGNN with the back propagation algorithm according to the loss. Firstly, we calculate the prediction value for $u_i \rightarrow p_j$ according to an MLP method, which is defined as Eq.(15):

$$y_{\text{pred}} = \text{sigmoid}\left(\text{MLP}\left(\vec{z}_i^p \| \vec{z}_j^u\right)\right) \tag{15}$$

where MLP denotes a neural network model stacked by two fully connected neural layers, $\|$ is the concatenation operation, $\vec{z}_i^p$ denotes the representation of POI, $\vec{z}_j^u$ is the members' representation, y_{pred} is a probability value in the range of $(0, 1)$ which indicates the possibility of having a connecting edge between group member u_i and POI p_j.

Then, the binary cross entropy is selected as the loss function, which is described as Eq.(16):

$$\mathcal{L} = -w\left[y_{\text{true}} \cdot \log y_{\text{pred}} + (1 - y_{\text{true}}) \cdot \log\left(1 - y_{\text{pred}}\right)\right] \tag{16}$$

where w is a parameter used to overcome the problem of unbalanced number of positive and negative samples, which can make the training process of SBGNN more stable, y_{true} denotes the ground truth which is mapped $\{-1, 1\}$ to $\{0, 1\}$.

4.3. Learning members' POI transfer preferences

Since members' POI transfer preferences have great impact on POI recommendation for occasional groups. To learn members' POI transfer preferences, we first establish the directed POI transfer graphs according to all the check-in sequences of each member (that is, a directed graph is constructed based on each POI check-in sequence). Then, based on the POI transfer graphs, we apply SRGNN (Wu et al., 2019) to learn the representations of POIs which imply members' POI transfer preferences. In the following sections, we first introduce how to construct the directed POI transfer graphs, then, we elaborate members' POI transfer preferences learning process.

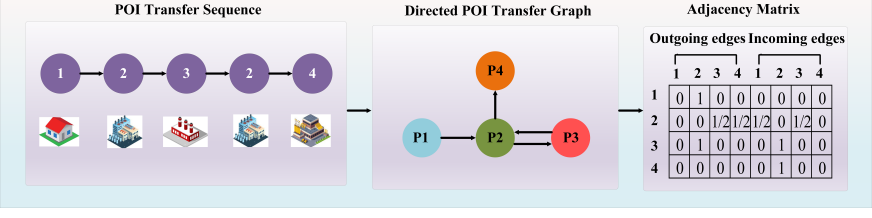


Figure 5: Directed POI transfer graph.

4.3.1. Constructing the directed POI transfer graphs

Assume that $S = \{S_i, S_j, ..., S_r\}$ denotes the set of check-in sequences of all members in the OG, where $S_i = \{s_1, s_2, \ldots, s_i\}$ represents the set of check-in sequences of the member u_i, and $s_i = \{p_1, \ldots, p_k\}$ $(k \in [0, n])$ indicates the i^{th} POI check-in sequence of member u_i. Based on the check-in sequence s_i, we can construct the directed POI transfer graph $G_T^i = (P_i, E_i)$, where P_i denotes the set of POIs, E_i refers to the set of connecting edges in graph G_T^i, edge $e = (p_k, p_j)$ $(e \in E_i)$ indicates that member u_i visited POI p_j after visiting p_k in the sequence s_i.

Assume that $A_i \in \mathbb{R}^{n \times 2k}$ denotes the adjacency matrix of graph G_T^i. A_i is defined as the concatenation of two adjacency matrices $A_i^{(in)}$ and $A_i^{(out)}$, which displays the weighted connections between the G_T^i's edges, including outgoing and incoming edges. In the graph G_T^i, there exists incoming edge e_{ji}^{in} and outgoing edge e_{ij}^{out} for each node.

Considering several POIs may show in a sequence more than once, so we need to allocate normalized weights to these edges (Zhang et al., 2020a). Let A_{ji}^{in} denote the weight of incoming edge e_{ji}^{in}, which can be calculated with Eq.(17), and A_{ij}^{out} denote the weight of outgoing edge e_{ij}^{out}, which can be calculated with Eq.(18):

$$A_{ji}^{in} = \frac{\text{count}\left(v_j, v_i\right)}{\sum_{k \in N_{in}(i)} \text{count}\left(v_k, v_i\right)} \tag{17}$$

$$A_{ij}^{out} = \frac{\text{count}\left(v_i, v_j\right)}{\sum_{k \in N_{out}(i)} \text{count}\left(v_i, v_k\right)} \tag{18}$$

where $count(x, y) \in \{0, 1\}$ represents whether there exist an edge between node v_i and v_j, N_{in}^i denotes the predecessor nodes set of node v_i, N_{out}^i is the successor nodes set of node v_i.

Take a POI check-in sequence $s_i = \left\{p_1, p_2, p_3, p_2, p_4\right\}$ for example, the directed POI transfer graph and the corresponding adjacency matrix A_k can be illustrated as Fig. 5. According to the directed graph construction method, we can construct many directed POI transfer graphs according to all the POI check-in sequences, and then we learn members' POI transfer preferences from these directed POI transfer graphs.

4.3.2. Learning members' POI transfer preferences

Here we take the graph G_T^i as an example to illustrate the learning process of SRGNN. Specifically, assuming that $Q = \left\{\vec{q}_1, \vec{q}_2, \ldots, \vec{q}_k\right\}$ ($\vec{q}_k \in \mathbb{R}^F$) represent the set of POIs' representations, where F indicates the dimension. SRGNN input the Q_i and A_i of the graph G_T^i into SRGNN, and the representations of POIs will be learned by the gating mechanism of SRGNN. For the POI check-in sequence s_i, the representations of neighbors are aggregated based on the adjacency matrix A_i. The calculation is defined as Eq.(19):

$$a_i^t = A_i \left[q_1^{t-1}, q_2^{t-1}, \ldots, q_n^{t-1}\right]^T H + b \tag{19}$$

where A_i denotes the adjacency matrix, which can be used to present the weighted connections of the graph G_T^i, including outgoing and incoming edges, $\vec{q}_k$ denotes the representation of POI p_k, H refers to the weight matrix, b denotes the bias vector.

Secondly, we calculate the values of the update and the reset gate according to Eq.(20). The update gate control what information should be kept, while the reset gate control what information should be ignored.

$$\begin{aligned} z_i^t &= \sigma\left(W_z a_i^t + U_z q_i^{t-1}\right) \\ r_i^t &= \sigma\left(W_r a_i^t + U_r q_i^{t-1}\right) \end{aligned} \tag{20}$$

where z_i^t represents the update gate, r_i^t indicates the reset gate, W and U refers to the model parameters and $\sigma(\bullet)$ denotes the activation function.

Further, the candidate state $\widetilde{q_i^t}$ of node p_i is calculated according to its previous state q_i^{t-1}, current state a_i^t and reset gate state r_i^t, which is defined as Eq.(21):

$$\widetilde{q_i^t} = \tanh\left(W_o a_i^t + U_o\left(r_i^t \odot q_i^{t-1}\right)\right) \tag{21}$$

Finally, the final state of node p_i is calculated based on the previous hidden state q_i^{t-1} and candidate state $\widetilde{q_i^t}$, under the control of the update gate state z_i^t. The calculation process is detailed as Eq.(22):

$$q_i^t = \left(1 - z_i^t\right) \odot q_{i-1}^t + z_i^t \odot \widetilde{q_i^t} \tag{22}$$

The output of SRGNN is denoted as $S = \left[\vec{q}_1^p, \vec{q}_2^p \ldots, \vec{q}_n^p\right]$, where $\vec{q}_i^p$ indicates the representation of POI p_i, which imply members' POI transfer preferences for POI p_i.

4.3.3. Loss function of the members' POI transfer preferences learning module

To train the SRGNN model, we take the sequence prediction as the driving task to activate the training of the SRGNN model. Specifically, we predict the probability for each candidate POI which indicates the possibility of a POI as the last interacted POI in a sequence. In other words, if the predicted probability of a POI is larger, it means this POI is more likely to be the last interacted POI in a sequence. Then, we calculate the loss of SRGNN based on the predicted score and the ground truth value, and finally train the SRGNN by back propagation algorithm according to the calculated loss.

Firstly, according to the calculation process in (Wu et al., 2019) and considering the local embedding $\vec{s}_l$ of POI transfer sequence s_i, for POIs transfer sequence $s_i = \{p_1, p_2, p_3, ..., p_k\}$, the local embedding $\vec{s}_l$ can be defined as $\vec{q}_n$ of last-checked POI p_n ($\vec{s}_l = \vec{q}_n$). Then, we consider the global embedding $\vec{s}_g$ of the POI transfer sequence s_i by aggregating all POIs' representations. Considering information involved in these POIs' representations may have varying degrees of priority, thus, we use the soft-attention mechanism to better depict the global embedding $\vec{s}_g$, which is illustrated as Eq.(23):

$$\begin{aligned} \alpha_i &= \vec{k}^{\top} \cdot \sigma\left(W_1 \vec{q}_n + W_2 \vec{q}_i + c\right) \\ \vec{s}_{\mathrm{g}} &= \sum_{i=1}^{n} \alpha_i \cdot \vec{q}_i \end{aligned} \tag{23}$$

where parameters $\vec{k}$, W_1 and W_2 control the weights of POIs' representations, c is the trainable parameter.

Thirdly, we compute the hybrid embedding $\vec{s}_h$ by taking linear transformation over the concatenation of the local and global embedding representations, which is illustrated as Eq.(24):

$$\vec{s}_{\mathrm{h}} = W_3 \left[\vec{s}_{\mathrm{l}} \| \vec{s}_{\mathrm{g}}\right] \tag{24}$$

where matrix W_3 is responsible for compressing the $\vec{s}_{\mathrm{g}}$ and $\vec{s}_{\mathrm{l}}$ into the latent space.

Then, after obtaining the embedding representation of each POI transfer sequence $\vec{s}_h$, we compute the score $\hat{z}_i$ for each candidate POI p_i by multiplying its representation $\vec{q}_i$ with hybrid embedding $\vec{s}_h$, which can be defined as Eq.(25):

$$\hat{z}_i = \vec{s}_h^{\,\mathrm{T}} \cdot \vec{q}_i \tag{25}$$

Finally, we construct a prediction vector $\hat{\vec{z}}$ over the concatenation of all the prediction scores $\hat{z}_i$, and then use the softmax function to obtain the output vector $\hat{\vec{y}}$, which can be defined as Eq. (26):

$$\begin{aligned} \hat{\vec{z}} &= \left[\hat{z}_1, \hat{z}_2, ..., \hat{z}_n\right] \\ \hat{\vec{y}} &= \sigma\left(\hat{\vec{z}}\right) \end{aligned} \tag{26}$$

where σ is the softmax function, $\hat{\vec{z}}$ represents the prediction score for all candidate POIs, $\hat{\vec{y}}$ denotes the possibility of POIs showing to the last interaction in POI transfer sequence S_i.

For each POI transfer sequence, the loss function is defined based on the prediction and the ground truth according to the cross entropy function, which is presented as Eq.(27):

$$\mathcal{L}\left(\hat{\vec{y}}\right) = -\sum_{i=1}^{n} \vec{y}_i \log\left(\hat{y}_i\right) + \left(1 - \vec{y}_i\right) \log\left(1 - \hat{y}_i\right), \hat{y}_i \in \hat{\vec{y}} \tag{27}$$

where $\vec{y}$ denotes the one-hot encoding vector of the ground truth POI. After obtaining the loss $\mathcal{L}\left(\hat{\vec{y}}\right)$, SRGNN is trained using the back propagation mechanism according to the calculated $\mathcal{L}\left(\hat{\vec{y}}\right)$. It should be noted that most POI transfer sequences length are relatively short. Therefore, it is suggested to choose a relatively small number of training epochs to prevent over-fitting.

4.4. Prediction module

From the above three sections, we obtain the fitted representation of the OG ($\vec{g}$) and two sets of POI representations (Z and S). Based on the above learned results, we calculate the rating of the OG on each POI, and then recommend the TOP-K POIs to the OG. The details are presented as follows. Firstly, the final representations of POIs ($\vec{h}_{\mathrm{p}}$) are generated via summation on the two sets of POIs representations (Z and S), which is illustrated as Eq.(28):

$$\vec{h}_{p_i} = \vec{z}_i^p + \vec{q}_i^p, \vec{z}_i^p \in Z, \vec{q}_i^p \in S \tag{28}$$

Then, the rating of the OG on each POI (denoted as $\hat{y}_i$) is calculated via vector product based on the OG's fitted representation ($\vec{g}$) and the final POI representation ($\vec{h}_{p_i}$). For comparison purposes, the rating $\hat{y}_i$ is compressed in the range (0,1) with the $Sigmoid$ function. The calculation process is illustrated as Eq.(29). Finally, the ratings of the OG on all the POIs are ranked in descending order, and the TOP-K POIs are recommended to the OG.

$$\hat{y}_i = \mathrm{Sigmoid}\left(\vec{g}^T \cdot \vec{h}_{p_i}\right), \hat{y}_i \in (0,1) \tag{29}$$

4.5. The training strategy of PROG-HGNN

Our proposed model PROG-HGNN consists of four modules: group fitted representation learning module (Module I), members' POI interaction preferences learning module (Module II), members' POI transfer preferences learning module (Module III) and POI recommendation module (Module IV). Actually, Module I learns the members' representation based on members' social network graph, Module II learns POIs' representations based on members' POI interactive graph, Module III learns POIs presentations based on directed POI transfer graphs. On the one hand, the first three modules conduct different presentations learning process based on three completely different graphs, different input of the three modules makes the training process of the first three modules different, and these three modules cannot be trained jointly. On the other hand, it is difficult to adjust the parameters of the first three modules at the same time, so as to ensure the whole model to obtain the best performance.

Table 1
Parameters and loss functions of the three modules.

Parameters	**GAT**	**SBGNN**	**SRGNN**
num_layers	2	2	1
epochs	512	32	64
num_heads	2	-	-
batch_size	32	500	100
hidden_dim	32	9	9
dropout	0.5	0.5	0.5
learning_rate	0.01	0.005	0.001
optimizer	Adam	Adam	Adam
weight_decay	5e-4	1e-5	1e-5

Therefore, we propose to conduct independent training for each module. Moreover, we argue that the performance of the whole PROG-HGNN model would be best when the performance of each module reaches the optimal performance. To get the superior parameters of the first three modules, so as to ensure the PROG-HGNN model to obtain optimal performance, we train the three modules separately based on different kinds of graphs, we retain the trained parameters of each module for the POIs recommendation for OG. The parameters for the three modules are presented in Table 1. POI recommendation for OG based on PROG-HGNN model is described as Algorithm 1.

5. Experiments

To verify the performance of our proposed model, we conduct extensive experiments based on three public datasets. In the following sections, we first introduce the three public datasets and experimental environment, then we introduce the twelve newest baseline models, and finally present and analyze the experimental results.

Algorithm 1 : POI Recommendation for Occasional Group Based on PROG-HGNN

Input: OG: Occasional group, S: members' POI check-in set, GS: social relationships of members of the OG
Output: POI recommendation results for the occasional group OG.

Construct social network graph G_s
for each member u_i in G_s **do**
    Compute u_i's social influence I_i according to Eq.(1)
    Compute u_i's attention coefficients e_ij according to Eq.(2)
    Update u_i's representation $\vec{h'}$ according to Eq. (4)
end for
Generate OG's representation $\vec{g}$ according to Eq. (5)
Construct signed interactive bipartite graph G_I
for each POI p_j in G_I **do**
    Get positive and negative neighbors
    Collect messages from positive and negative neighbors according to Eq. (8) and Eq. (10)
    Aggregate messages from positive and negative neighbors according to Eq. (12)
    Update p_j's representation $\vec{z}_j^p$ according to Eq. (13) and Eq. (14)
end for
Construct directed POI transfer graph G_T
for each p_j in G_T **do**
    Collect messages from neighbors according to Eq. (19)
    Compute values of update and reset gates according to Eq. (20)
    Compute candidate state according to Eq. (21)
    Compute output state $\vec{q}_j^p$ according to Eq. (22)
end for
Generate final POI representation according Eq. (28)
Compute prediction rating $\hat{y}_i$ according to Eq. (29)
Sort predict ratings in descending order
Select TOP-K POIs as recommendation result for the OG
end

Table 2
Statistics of the three datasets.

Dataset	Number of users	Number of POIs	Number of check-ins
Foursquare	2,551	13,474	124,933
Gowalla	5,628	31,803	620,683
Yelp	30,887	18,995	860,888

5.1. Datasets and Experimental Environment

Currently, to the best of our knowledge, there is no public datasets about POI check-in records of the OGs. Moreover, the main idea of our work is first to turn the OG into a virtual user, and then transform the problem of POI recommendation for OG to the problem of personalized POI recommendation. Therefore, we adopt the three public LBSN datasets about personalized POI recommendation to verify the performance of our proposed model. The three public LBSN datasets are Yelp[1], Gowalla[2] and Foursquare[3] (described as Table 2). Massive social relations and check-in data are contained in these three datasets, and each check-in record includes a unique user ID and POI ID. We process these three datasets to filtered out inactive users and unpopular POIs. Then, 60%, 20% and 20% of each dataset are selected as the training, validation and testing datasets.

In our experiments, the OGs are generated by randomly selecting certain users from the three processed public datasets. It should be noted that the POI interaction preferences learning module and POI transfer preferences learning module are trained based on the group members' POIs interaction data and POIs transfer data. The experimental platform is as follows, Operating system: Windows 10 Professional 64-bit. CPU:12th Gen Intel(R) Core(TM)

[1] https://www.kaggle.com/datasets/yelp-dataset/yelp-dataset
[2] https://snap.stanford.edu/data/loc-gowalla.html
[3] https://sites.google.com/site/yangdingqi/home/foursquare-dataset

i9-12900K 3.19 GHz. GPU: NVIDIA GeForce RTX 3090. RAM: 32GB. Model developing platform: Pycharm (Community Edition)[4]. Programming language: Python 3.8[5]. Deep learning framework: Pytorch[6].

5.2. Evaluation Metrics

To evaluate the performance of the PROG-HGNN model, three widely-used evaluation metrics are adopted in our experiments, which are Precision ($P@K$), Recall ($R@K$) and Mean Average Precision (MAP, $M@K$). The Precision metric is used to measure how many POIs in the recommended POI list are included in the testing data. The Recall metric is used to measure how many POIs in the testing set are returned to the recommended POI list. The metric MAP is used to measure the quality of recommendation from the position of POIs in the recommended list. The formulas for the three metrics are defined as Eq.(30), Eq.(31) and Eq.(32):

$$\text{Precision@K}=\frac{|R(\text{ g})\cap T(g)|}{|R(g)|} \tag{30}$$

$$\text{Recall }@K=\frac{|R(g)\cap T(g)|}{|T(g)|} \tag{31}$$

$$MAP@K=\frac{\frac{1}{K}\sum_{k=1}^{K}\text{Precision }@k}{|T(g)|} \tag{32}$$

where $R(g)$ denotes the recommended POI list for the OG, and $T(g)$ denotes the actual check-in list of the OG. In our experiments, we use N to denote the length of the POI recommendation list, and N is set as 5, 10, 15, 20, respectively.

5.3. Baseline Models

Currently, to the best of our knowledge, none research has been conducted on POI recommendation for occasional groups. Therefore, there are no suitable models for performance comparison with our proposed model. To solve this problem, considering we transform the POI recommendation problem for occasional groups into the personalized POI recommendation problem. Consequently, we select twelve state-of-the-art personalized POI recommendation models as the baseline models to evaluate and validate the effectiveness of our proposed model. The introductions and parameter settings about these twelve baseline models are presented as follows.

TransMKR (Hu et al., 2022): which is a multi-task learning model for POI recommendation based on knowledge graph translation. This model used POI attributes (such as access times and locations of POIs) to construct knowledge graph. Based on different attribute values, the representations of POIs can be described in a more detailed and accurate manner. The dimension of user and POI representation are 8, the number of low layers and the high layers are both 1, the epochs is 10, the batch size is set as 32, the weight of L_2 regularization is $1e^{-6}$, the learning rate of recommendation module is $2e^{-4}$, the learning rate of KGE module is $2e^{-5}$, the training interval of KGE module is 2.

NPGR (Yu et al., 2022): NPGR constructs a heterogeneous graph based on users' POI interaction data, which includes users, POI categories, and check-in time windows. Using the Node2Vec method to extract node representations, NPGR also takes into account the user check-in frequency, locations and popularity of POIs, and other factors for POI recommendations. The γ denotes the relative importance of geographical influences and is set as to 0.1, α and β denotes he relative importance of popularity and category influences are set as to 0.4 and 0.2 for the Foursquare, respectively, the best threshold δ is 1, the optimal $MinPts$ is set to 7, other parameters $d, n\omega, \omega l, \omega, p$ are set as $d = 16, n\omega = 14, \omega l = 25, p = 1$, respectively.

GSTN (Wang et al., 2022): GSTN is a graph-enhanced spatio-temporal network model. Relying on the GNN technology, GSTN can effectively capture the relationship between time and space of POI. The grid search strategy is leveraged to perform hyper-parameter tuning for the GSTN, the learning rate is set as in the range of $\{0.01, 0.005, 0.001, 0.0005, 0.0001\}$, the coefficients λ of L_2 regularization is searched in the range of $\{10^{-3}, 10^{-4}, 10^{-5}\}$, the dropout is examined in the range of $\{0, 0.1, 0.2, 0.3, 0.4, 0.5\}$, the sequence length is fixed to 20, the distance threshold is 1 km, the dimension of latent representation is 64, K means the negative samples and is set as 5, the cell

[4]https://www.jetbrains.com/pycharm/download/
[5]https://www.python.org/downloads/release/python-380/
[6]https://pytorch.org/

and hidden state are both initialized as 0, the batch size is 128, the Adam optimizer is used to optimize the model and the learning rate is set as 0.001, the coefficients λ is 10^{-4}.

STORE (Liu et al., 2022c): STORE studies impact of spatio-temporal factors (for the recommendation of POIs) on user check-in behaviors. Compared with the traditional methods of studying time and space separately, STORE checks the factors of time and space simultaneously. The parameters of STORE are initialized randomly with a gaussian distribution, the learning rate is set as 0.001, the Adam is applied as optimizer, the batch size is 256, the regularization parameter is 0.0001, the dimension of user and POI embedding is set as 128, the dimension of spatio-temporal effects and meta-path-based context is 32, the encoding length of Geohash is set as 5, 10% training data is selected as the validation set to optimize the ItemKNN's parameters.

MBR (Lang et al., 2022): MBR is a POI recommendation model designed for individual based on multivariate GNNs, which reduces the computational costs of traditional bipartite graph models through clustering operations on the graph model. To improve the performance, MBR deeply analyzes the influence of the social network of users, the locations of POIs, and the user check-in time on the users' check-in behaviors. We cannot present the parameters of MBR in detail, because the authors did not display the parameters related to the MBR in the paper (Lang et al., 2022).

STGN (Zhao et al., 2022): STGN applies the gating mechanism to the study of spatio-temporal factors. The users' check-in behaviors are serialized, and the gating mechanism is one of the most efficient tools for processing serialized data. Therefore, the recommendation accuracy is improved by introducing the gating mechanism in modeling the user's check-in behavior pattern. STGN leverages the mini-batch learning method, and train the model on users' existing histories until convergence, and a gradient step is used to optimize the loss based on the model output, Adam is selected to train the model.

OMPR (Lv et al., 2022): OMPR resolves the difficulty to obtain users' long-term preferences in online POI recommendations. The accuracy and speed of online POI recommendation are improved through the "offline learning, online recommendation" strategy. The latent dimension is 50, the dimension of the importance vector is 20, the geographical influence is set 60, the parameters α and η of the weighting scheme are set to 2 and $1e^{-5}$, respectively, The gradient descent learning rate and regularization λ are all set to 0.001, the network architecture is set in the range of $[N, 200, 50, 200, N]$. The number of convolutional layer and filter is 1 and 10 in the online transfer network, respectively.

FG-CF (Cai et al., 2022): Combining collaborative filtering with graph convolutional network, FG-CF alleviates the problem of user check-in data sparsity. In addition, the user's social information is added to the user-POI bipartite graph, so that the user's representations can be described more accurately. The batch size is 1024, The size of user and POI representation is fixed to 64, the number of connectivity layers is set to 3, the dimension of each layer is 64, the learning rate is determined to be between $\{0.0001, 0.001, 0.01, 0.05, 0.1\}$, the coefficient of L_2 normalization is in the range of $\left\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\right\}$, the dropout ratio is in the range of $\{0.1, ..., 0.9\}$, FG-CF use the Xavier initializer to initialize model's parameters and select the Adam to train the model.

FedPOIRec (Perifanis et al., 2023): FedPOIRec focuses on the privacy protection when perform POI recommendation. It combines federated learning techniques with preferences from users social friends to generate recommendation results. The presentation dimension is set to 128, learning rate L_2 is set to 10^{-6}. The Adam optimizer is used to optimize the model. As for the federated setting, FedPOIRec use the SGD technique with learning rate $\eta = 10^{-1}$.

DCLR (Long et al., 2023): DCLR is a decentralized model which trains personalized recommendation model locally. DCLR also includes two self-supervised signals to improve POI representations, which can be used to solve the data sparsity problem. The batch size is 16, latent embedding dimension d is set to 32, learning rate is 0.002, dropout is 0.2 for all layers, the maximum training epoch is 50.

DSMR (Wang et al., 2023): DSMR is a deep semantic POI recommendation model, which adopts a prompt engineering to conduct semantic modeling and uses a pre-trained language model to learning deep semantic information. The presentation dimension d is set to 128, dropout rate is 0.3, Adam optimizer is used to train the model, learning rate is 10^{-4} with batch size 40 with 0 epochs for the first training process, and for the second training process, the batch size is set to 20 with 200 epochs.

TGSTAN (Cao et al., 2023): TGSTAN is an improved spatial–temporal attention network, which includes a dynamic graph convolution network module to capture the local spatial correlations and update the graph weights in real time and learn user's personalized local spatial relationships. The attention head is 2, the dropout is 1.0 and 0.7 for two different datasets, learning rate is set to 10^{-3}, weight decay is $5e-4$, the training epochs is 35, and Adam optimizer is selected to train the model.

5.4. Performance Comparison

To verify the performance of our proposed model PROG-HGNN, we randomly generate different *OGs* with size of 50 based on the three datasets, and set the actual check-in size for the occasional groups as 20. We compare the performance of PROG-HGNN with other baseline models (selected from top academic journals and top conferences) in terms of Precision@K and Recall@K. Since the value of K set by the baseline models is different with the PROG-HGNN model, and some important model's parameters are not presented in the original papers, so we cannot reproduce these baseline models perfectly as the original papers. Therefore, to ensure the fairness in performance comparison, we take the results in the original papers as the baseline models' results, and get the experimental results of our proposed model through executing PROG-HGNN. Experimental results are shown in Table 3, Table 4 and Table 5. We mark the best performance value in each table with boldface. Moreover, since some baseline models were not tested with all the K values on the three datasets, that is the reason why there are some missing values in Table 3, Table 4 and Table 5, and we mark these missing data with symbol "-". In the following sections, we will analyze the experimental results on the three datasets, respectively.

On the dataset Foursquare, we select OMPR, STORE, GSTN, NPGR and TransMKR as the baseline models, and the experimental results are presented in Table 3. From Table 3, for the metric Precision@K, we can find that our proposed model PROG-HGNN outperforms other baseline models when the K is set from 5 to 20. Specifically, it first decreases from 0.6 to 0.4 as K increases from 5 to 10, and then keeps the same value with the change of K. Moreover, for the metric Recall@K, with K increases from 5 to 20, Recall@K presents a gradual upward trend, and reaches the highest value 0.4 when the K is set to 20. However, when the K is set to 5 and 10, the Recall@K value of the PROG-HGNN are 8% and 4% lower than that of the GSTN, respectively. Generally, based on the results in Table 3 we can calculate that the PROG-HGNN improves Precision@K and Recall@K by 24% and 6% evenly. Therefore, we can conclude that the PROG-HGNN is better than that of the other baseline models on the Foursquare dataset.

Table 3
Performance comparison on the Foursquare dataset.

Models	Precision@K				Recall@K			
	5	10	15	20	5	10	15	20
OMPR	-	0.02	-	-	-	0.12	-	-
STORE	-	-	-	-	0.15	0.23	-	-
GSTN	-	-	-	-	**0.27**	**0.34**	-	-
NPGR	0.05	0.04	0.04	0.04	0.05	0.07	0.09	0.11
TransMKR	-	0.40	0.35	-	0.20	0.30	-	-
FedPOIRec	0.21	0.15	-	-	0.13	0.16	-	-
DSMR	0.51	**0.59**	-	-	-	-	-	-
TGSTAN	0.47	0.53	-	-	-	-	-	-
PROG-HGNN	**0.6**	0.4	**0.4**	**0.4**	0.15	0.2	**0.3**	**0.4**

On dataset Gowalla, we take FG-CF, STGN, MBR, GSTN and TransMKR as the baseline models, and the experimental results are presented in Table 4. From Table 4, for the metric Precision@K, it is obvious that our proposed model PROG-HGNN is generally superior to the other baseline models. As K changes from 5 to 20, the value of Precision@K shows a fluctuating tendency. Specifically, it first drops from 0.8 to 0.4 when the K increases from 5 to 15, and then rises to 0.45 when the K increases to 20. As for Recall@K, it performs an upward trend when the K increases from 5 to 20, and gets the highest value 0.45 when the K is set to 20. However, when the K is set to 5 and 10, the Recall@K value of the PROG-HGNN are 5% and 12% lower than that of the GSTN, respectively. Based on the results in Table 4 we can calculate that the PROG-HGNN improves Precision@K and Recall@K by 30% and 7% evenly. Therefore, we can conclude that the PROG-HGNN is better than that of the other baseline models on the Gowalla dataset.

On the dataset Yelp, since only FG-CF, DCLR and OMPR verified their model performance on this dataset, so we choose these three models as baselines models to test the performance of our proposed model PROG-HGNN, and the experimental results are presented in Table 5. From Table 5 we can find that the PROG-HGNN outperforms other baseline models in term of all evaluation metrics. Generally, based on the results in Table 5, we can calculate that the PROG-HGNN improves Precision@K and Recall@K by 61% and 48% evenly. Therefore, we can conclude that the PROG-HGNN is better than that of the other baseline models on the Yelp dataset.

Table 4
Performance comparison results on the Gowalla dataset.

Models	Precision@K				Recall@K			
	5	10	15	20	5	10	15	20
FG-CF	0.09	0.07	0.06	0.06	0.07	0.10	0.13	0.15
STGN	0.16	0.20	-	-	-	-	-	-
MBR	0.03	0.02	0.02	0.02	0.12	0.18	0.20	0.21
GSTN	-	-	-	-	**0.21**	**0.28**	-	-
TransMKR	0.43	0.42	-	-	-	-	0.13	0.23
TGSTAN	0.30	0.39	-	-	-	-	-	-
PROG-HGNN	**0.8**	**0.5**	**0.4**	**0.45**	0.2	0.25	**0.3**	**0.45**

Table 5
Performance comparison results on the Yelp dataset.

Models	Precision@K				Recall@K			
	5	10	15	20	5	10	15	20
FG-CF	0.28	0.02	0.02	0.02	0.03	0.05	0.06	0.08
OMPR	-	0.02	-	-	-	0.10	-	-
DCLR	0.44	0.55	-	-	-	-	-	-
PROG-HGNN	**0.8**	**0.9**	**0.87**	**0.9**	**0.2**	**0.45**	**0.65**	**0.9**

Based on the above experimental results, we can get the following conclusions: (1) PROG-HGNN outperforms twelve baseline models in term of all evaluation metrics, which demonstrates that our proposed model PROG-HGNN is reasonable and effective; (2) Experimental results prove that GNN-based or graph-based models can obtain better performance than other kinds of models, which also can prove that our idea of learning from the signed interactive bipartite graph and the directed POI transfer graphs is effective.

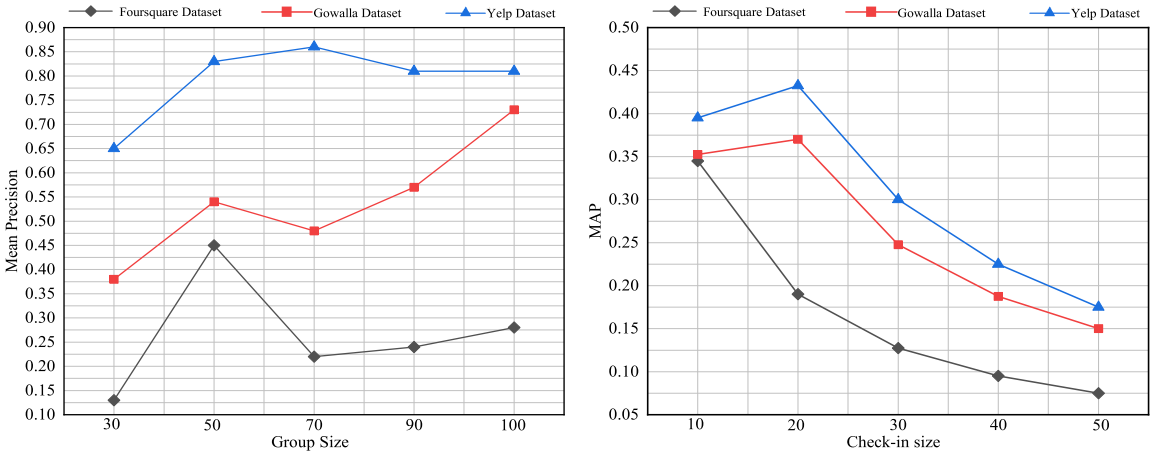
5.5. Influence of Group Sizes on Performance of PROG-HGNN

For group recommendation model, the group size have great influence on the model's performance. In this experiment, we aim to investigate the effects of various group sizes on PROG-HGNN's performance, and then determine the best group size for the PROG-HGNN model. Specifically, we first set the group size from 30 to 100, and then we test the PROG-HGNN's performance with mean Precision under different group sizes on the three datasets. The experimental results are presented in Fig. 6(a). From Fig. 6(a) we can find that the mean Precision of the PROG-HGNN model is fluctuating with the increase of the group size on the three datasets. Besides, the performance of the PROG-HGNN on the Gowalla and Foursquare datasets is weaker than that on the Yelp dataset. Specifically, on the dataset Foursquare, when the group size increases from 30 to 70, the Mean Precision value first increases from 0.13 to 0.45, then decreases to 0.22. After that, the Mean Precision value gradually raises to 0.28 when the group size increases from 70 to 100. On the Gowalla dataset, the Mean Precision value first raises from 0.38 to 0.54 and then drops to 0.48 when the group size increases from 30 to 70. Then, the value of the Mean Precision steadily increases from 0.48 to 0.73, as the group size raises from 70 to 100. On the Yelp dataset, the Mean Precision value first raises from 0.65 to 0.86 when the group size increases from 30 to 70, and then drops to 0.81 as the group size grows from 70 to 100.

Based on the above experimental results, we can find that when the group size is small, the training data for PROG-HGNN is insufficient, which influences PROG-HGNN's performance. Otherwise, when the group size is too large, the performance differences of group members will also impact the performance of the model PROG-HGNN. Therefore, group size is an important factor in group recommendation, and selecting an appropriate group size is crucial for improving the model performance.

5.6. Influence of Check-in Sizes on PROG-HGNN

For personalized POI recommendation, the POI list recommended by the model is usually compared with the user's historical POI interaction records, thus to calculate the model's recommendation accuracy. Actually, different numbers of historical check-in records will have a certain impact on the performance of PROG-HGNN model, we need to determine an optimal number of historical check-in records to enable PROG-HGNN to obtain better performance.



(a) Impact of the group size for PROG-HGNN on three datasets (b) Impact of the check-in size for PROG-HGNN on three datasets

Figure 6: Influence of group sizes and check-in sizes on PROG-HGNN.

However, since there is no real POI check-in data about occasional groups, which makes it impossible to calculate the recommendation accuracy of the PROG-HGNN model. To address this issue, in our experiments, we take the historical check-in records of users with similar features to the occasional group as the historical check-in records of the occasional group, the similarity between the occasional group and users are computed by the cosine similarity. In this experiment, we set the check-in size from 10 to 50, and compare the performance of PROG-HGNN in terms of MAP with different check-in sizes on the three datasets. The experimental results are presented in Fig. 6(b).

From the Fig. 6(b) we can find that, the MAP of PROG-HGNN on the datasets Gowalla and Yelp are higher than that on the dataset Foursquare. This is because users in the Gowalla and Yelp datasets have more check-in records compared to the Foursquare dataset. When check-in records becomes richer, the PROG-HGNN model can be trained adequately. Therefore, PROG-HGNN can make full use of the richer check-in records and get better performance. When the check-in size increases from 10 to 20, the MAP value on the datasets Gowalla and Yelp raises from 0.35 and 0.39 to 0.37 and 0.43, respectively. Then, as the check-in size is in the range of [20, 50], the MAP value on these two datasets keeps decreasing and falls to 0.15 and 0.17, respectively. On the dataset Foursquare, the MAP value first reaches its maximum of 0.35 when the check-in size is set to 10, and then gradually falls when the check-in size increases. When the check-in size is set to 50, the MAP value gets the minimum value of 0.08. The above experimental results demonstrate that check-in size is an significant factor in group recommendation, and choosing an appropriate check-in size is important for the model performance improvement.

5.7. Ablation Experiments

To verify the effectiveness of different modules of the PROG-HGNN, some module ablation experiments are conducted based on the three datasets. Specifically, we compare PROG-HGNN with its two variants, namely SBGNN-PROG and SRGNN-PROG. The generation of these two variants are described in the following paragraphs. It should be noted that the group fitted representation learning module in the PROG-HGNN is used for generating the fitted representation of the occasional group. Without this module, PROG-HGNN cannot work. Therefore, we do not conduct ablation study of this particular module.

- **SBGNN-PROG:** which is generated by removing the members' POI transfer preferences learning module from PROG-HGNN. This variant only uses group fitted representation learning module and members' POI interaction preferences learning module, while ignoring members' POI transfer information.
- **SRGNN-PROG:** which is generated by removing the members' POI interaction preferences learning module from PROG-HGNN. This variant only uses group fitted representation learning module and members' POI transfer preferences learning module, while ignoring members' POI interaction information.

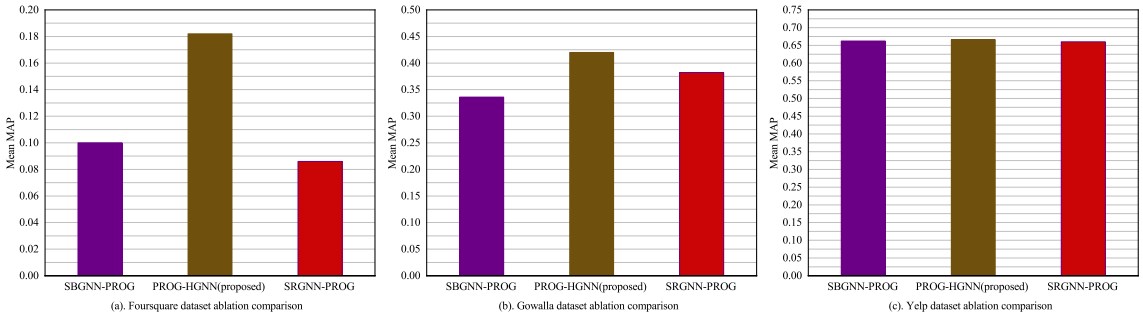


Figure 7: Results of the ablation experiment on the three datasets.

We compare the performance of the three models (PROG-HGNN, SBGNN-PROG, SRGNN-PROG) with mean MAP on the three datasets. The experimental results are presented in Fig. 7. From Fig. 7 we can find that PROG-HGNN outperforms the other two variants on the three datasets. Specifically, on the Foursquare dataset, SBGNN-PROG and SRGNN-PROG decreases by 0.08 and 0.09 in mean MAP compared with PROG-HGNN, respectively. On the Gowalla dataset, the mean MAP score of SBGNN-PROG and SRGNN-PROG are 0.08 and 0.04 lower than that of the PROG-HGNN. Although the mean MAP of the SBGNN-PROG is higher than that of the SRGNN-PROG on the Gowalla dataset, the difference is only 0.05, which indicates that members' POI transfer preferences learning module has weaker influence on PROG-HGNN than that of the POI interaction preferences learning module. On the Yelp dataset, the performance difference between these three models is not significant. This is because Yelp has the largest amount of check-in data in the three datasets, implying that sufficient data is beneficial for improving the performance of the model. The above experimental results demonstrate that exploiting members' POI interaction preferences and POI transfer preferences are effective in enhancing the performance of POI recommendation for occasional groups.

5.8. Time Complexity Analysis

In this section, we conduct the time complexity analysis on our proposed model PROG-HGNN and four representative baseline models. For model PROG-HGNN, it mainly consists of three GNN models, which are GAT, SBGNN and SRGNN. For the GAT, there are two main time-consuming operations: one is to map the feature representations of all nodes and the time complexity is $O(|V|\cdot F\cdot F^{'})$. The other operation is to calculate the attention coefficient α for each edge in the graph and the time complexity is $O(|E|\cdot F^{'})$, where F and $F^{'}$ are the dimension of input and output features, E and V are the number of the edges and nodes of the graph. After applying the multi-head attention mechanism, the complete time complexity of GAT is $O(K\cdot(|V|\cdot F\cdot F^{'}+|E|\cdot F^{'}))$, where K is the number of the attention heads. For SBGNN, it contains three main operations: message propagation, message aggregation and representation updating. The function of the message propagation is to collect information from different neighbor nodes on the different hidden layers according to the interaction edges. Therefore, the number of the interaction edges and the hidden layers determine the time complexity of the message propagation, which can be presented as $O(|E|\cdot l)$, where l and $|E|$ are the number of hidden layers and interaction edges. The message aggregation is conducted based on the multi-head attention mechanism, so the time complexity is $O(K\cdot(|V|\cdot F\cdot F^{'}+|E|\cdot F^{'}))$. Representation updating operation aims to update nodes' representations using a MLP, the time complexity of the MLP is $O(N\cdot d\cdot H^{l}\cdot O\cdot I)$, where N is the size of training samples, d is the feature dimension, H is the number of neurons for each hidden layer, O represents the number of output neurons, and I is the number of iterations. SRGNN is developed based on the gated graph neural networks (GGNN), so, we can use the time complexity of the GGNN to represent the time complexity of the SRGNN, which is $O(|E|)$, where $|E|$ is the number of the edges in the graph. Based on the above analysis, we can find that the SBGNN has the highest time complexity among the three models, thus we take SBGNN's time complexity as the time complexity of PROG-HGNN, that is $O(|E|\cdot l+K\cdot(|V|\cdot F\cdot F^{'}+|E|\cdot F^{'})+N\cdot d\cdot H^{l}\cdot O\cdot I)$.

Furthermore, we select four representative baseline models with better performance as the comparison objects, which are DCLR, TransMKR, TGSTAN and FedPOIRec. For DCLR, the time consumption mainly relies in two parts: local model optimization and model enhancement. The former's time complexity is $O(M)$, while the latter's time complexity is $O(2Q+|P|)$. Therefore, the total time complexity of DCLR is $O(M+2Q+|P|)$, where $|P|$ is the number of the POIs, M is the size of one check-in sequence, and Q represents the sum of the semantic

and geographical neighbors. For TransMKR, since it is a knowledge graph (KG) enhanced POI recommendation model, and the time complexity of the KG is determined by the knowledge graph embedding (KGE). Thus, we use the KGE's time complexity to denote the TransMKR's time complexity, which is $O(d)$, where d is the is the dimension of nodes presentation. For TGSTAN, its time complexity manly includes $O(l^2)$ for interpolation presentation, $O(|E|\cdot d^{'}+l^2\cdot d^{'}+(l^2+l)\cdot d^{'})$ for the spatial attention mechanism and graph convolution operations, $O(l)$ for positional encoding operation, and $O(l\cdot d^{'2})$ for the forward propagation. FedPOIRec is developed based on the federated learning, the calculation consumption lies in the evaluation keys generation and the weighted mean vector computation. The time complexity of FedPOIRec is $O(epoch\cdot N\cdot M)$, where $epoch$ means the training times, N refers to the number of samples in the dataset, and M denotes the population size in the federated learning mechanism. The time complexity of the five contrastive models are presented in Table 6.

Table 6
Time complexity analysis of five models.

Models	Time Complexity
PROG-HGNN	$O(\|E\|\cdot l+K\cdot(\|V\|\cdot F\cdot F^{'}+\|E\|\cdot F^{'})+N\cdot d\cdot H^{l}\cdot O\cdot I)$
DCLR	$O(M+2Q+\|P\|)$
TransMKR	$O(d)$
TGSTAN	$O(l^2+\|E\|\cdot d^{'}+l^2\cdot d^{'}+(l^2+l)\cdot d^{'}+l+l\cdot d^{'2})$
FedPOIRec	$O(epoch\cdot N\cdot M)$

From Table 6 we can find that TransMKR has the lowest time complexity, which implies that knowledge graph is a type of graph learning model with relatively lower time complexity. Besides, HGNN-PROG has a higher time complexity compared to the other baseline models. This is because that SBGNN module of PROG-HGNN needs to process two types of nodes (users and POIs) and two types of interaction edges (positive and negative), which is much more complex compared to the traditional homogeneous GNNs. Last but not least, PROG-HGNN has the best learning ability and outperforms the other baseline models significantly.

To verify the efficiency of our proposed model PROG-HGNN, we record the average training time and average convergence time of the three modules of PROG-HGNN on the three benchmark datasets. Specifically, for the SBGNN based module, the training time on each epoch data is 13 seconds, and the convergence time of this module is 512 seconds. For the GAT based module, the training time on each epoch data is 8 milliseconds, and the convergence time of this module is 4096 milliseconds. For the SRGNN based module, the training time on each epoch data and the convergence time is 6 milliseconds and 384 milliseconds, respectively. Moreover, for the prediction process, PROG-HGNN takes 10 seconds to produce the POI recommendation results for a occasional group. Based on the above experimental results we can conclude that although the time complexity of PROG-HGNN is a little high, the time consumption during the training and the prediction process is acceptable. Therefore, our proposed model PROG-HGNN is effective in POI recommendation for occasional groups.

6. Conclusion

Recently, although some research has been conducted on POI recommendation for fixed groups, few studies consider POI recommendation for occasional groups. To tackle this issue, this work proposes a POI recommendation model for occasional groups based on hybrid graph neural networks (named as PROG-HGNN). Specifically, PROG-HGNN first generates the fitted representation of the occasional group based on members' representations and their social influences. Then, based on members' POI signed interactive bipartite graph and members' POI transfer graphs, PROG-HGNN learns POIs' representations which imply members' POI interaction preferences and POI transfer preferences. Finally, based on the learned POIs' representations and the fitted representation of the occasional group, PROG-HGNN recommends the optimal POIs to the occasional group. All the experiments and further analyses are conducted on the three real-world datasets (Foursquare, Gowalla and Yelp), and the experimental results demonstrate the effectiveness of the proposed model. In our future works, we will explore context-aware POI recommendation for individuals and occasional groups. Moreover, we also plan to investigate how to recommend POIs to random groups.

Acknowledgment

This work is supported by the National Natural Science Foundation of China (Grant nos. 62273290, 61872126, 61832004, 61772155), Key projects of Shandong Natural Science Foundation (Grant no. ZR2020KF019), Special Funding Program of Shandong Taishan Scholars Project, Australian Research Council (ARC) Future Fellowship FT140101247 and Discovery Project DP200102298.

References

Bahari Sojahrood, Z. and Taleai, M. (2022). Behavior-based POI recommendation for small groups in location-based social networks. *Transactions in GIS*, 26(1):259–277.

Cai, Z., Yuan, G., Qiao, S., Qu, S., Zhang, Y., and Bing, R. (2022). FG-CF: Friends-aware graph collaborative filtering for POI recommendation. *Neurocomputing*, 488:107–119.

Cao, G., Cui, S., and Joe, I. (2023). Improving the spatial–temporal aware attention network with dynamic trajectory graph learning for next point-of-interest recommendation. *Information Processing & Management*, 60(3):1–19.

Chen, L., Cao, J., Chen, H., Liang, W., Tao, H., and Zhu, G. (2021). Attentive multi-task learning for group itinerary recommendation. *Knowledge and Information Systems*, 63(7):1687–1716.

Chen, W., Wan, H., Guo, S., Huang, H., Zheng, S., Li, J., Lin, S., and Lin, Y. (2022). Building and exploiting spatial–temporal knowledge graph for next POI recommendation. *Knowledge-Based Systems*, 258:109951.

Davtalab, M. and Alesheikh, A. A. (2021). A POI recommendation approach integrating social spatio-temporal information into probabilistic matrix factorization. *Knowledge and Information Systems*, 63(1):65–85.

Esen, H., Esen, M., and Ozsolak, O. (2017). Modelling and experimental performance analysis of solar-assisted ground source heat pump system. *Journal of Experimental & Theoretical Artificial Intelligence*, 29(1):1–17.

Esen, H., Inalli, M., Sengur, A., and Esen, M. (2008a). Artificial neural networks and adaptive neuro-fuzzy assessments for ground-coupled heat pump system. *Energy and Buildings*, 40(6):1074–1083.

Esen, H., Inalli, M., Sengur, A., and Esen, M. (2008b). Forecasting of a ground-coupled heat pump performance using neural networks with statistical data weighting pre-processing. *International Journal of Thermal Sciences*, 47(4):431–441.

Esen, H., Inalli, M., Sengur, A., and Esen, M. (2008c). Performance prediction of a ground-coupled heat pump system using artificial neural networks. *Expert Systems with Applications*, 35(4):1940–1948.

Esen, H., Ozgen, F., Esen, M., and Sengur, A. (2009). Artificial neural network and wavelet neural network approaches for modelling of a solar air heater. *Expert systems with applications*, 36(8):11240–11248.

Goossens, A., De Smedt, J., and Vanthienen, J. (2023). Extracting decision model and notation models from text using deep learning techniques. *Expert Systems with Applications*, 211:118667.

Hu, B., Ye, Y., Zhong, Y., Pan, J., and Hu, M. (2022). TransMKR: Translation-based knowledge graph enhanced multi-task point-of-interest recommendation. *Neurocomputing*, 474:107–114.

Huang, J., Shen, H., Cao, Q., Tao, S., and Cheng, X. (2021). Signed Bipartite Graph Neural Networks. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 740–749.

Huang, X., Chen, D., Wang, D., and Ren, T. (2020). Identifying influencers in social networks. *Entropy*, 22(4):450.

Khazaei, E. and Alimohammadi, A. (2018). An automatic user grouping model for a group recommender system in location-based social networks. *ISPRS international journal of geo-information*, 7(2):67.

Kuru, K., Clough, S., Ansell, D., McCarthy, J., and McGovern, S. (2023). Wildetect: An intelligent platform to perform airborne wildlife census automatically in the marine ecosystem using an ensemble of learning techniques and computer vision. *Expert Systems with Applications*, page 120574.

Lang, C., Wang, Z., He, K., and Sun, S. (2022). POI recommendation based on a multiple bipartite graph network model. *The Journal of Supercomputing*, 78(7):9782–9816.

Li, G., Chen, Q., Zheng, B., Yin, H., Nguyen, Q. V. H., and Zhou, X. (2020). Group-based recurrent neural networks for POI recommendation. *ACM Transactions on Data Science*, 1(1):1–18.

Li, R., Guo, J., Liu, C., Li, Z., and Zhang, S. (2022). Using Attributes Explicitly Reflecting User Preference in a Self-Attention Network for Next POI Recommendation. *ISPRS International Journal of Geo-Information*, 11(8):440.

Liu, J., Chen, Y., Huang, X., Li, J., and Min, G. (2023). Gnn-based long and short term preference modeling for next-location prediction. *Information Sciences*.

Liu, K., Zheng, W., Xiao, Y., and Zhai, X. (2022a). POI Recommendation Algorithm based on Region Transfer Collaborative Filtering. In *2022 IEEE 25th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pages 903–907. IEEE.

Liu, X., Yang, Y., Xu, Y., Yang, F., Huang, Q., and Wang, H. (2022b). Real-time POI recommendation via modeling long-and short-term user preferences. *Neurocomputing*, 467:454–464.

Liu, Y., Yang, Z., Li, T., and Wu, D. (2022c). A novel POI recommendation model based on joint spatiotemporal effects and four-way interaction. *Applied Intelligence*, 52(5):5310–5324.

Liu, Y., Yin, M., and Zhou, X. (2021). A Collaborative Filtering Algorithm with Intragroup Divergence for POI Group Recommendation. *Applied Sciences*, 11(12):5416.

Long, J., Chen, T., Nguyen, Q. V. H., and Yin, H. (2023). Decentralized collaborative learning framework for next poi recommendation. *ACM Transactions on Information Systems*, 41(3):1–25.

Lv, Y., Sang, Y., Tai, C., Cheng, W., Shang, J. S., Qu, J., Chu, X., and Zhang, R. (2022). Online meta-learning for POI recommendation. *GeoInformatica*, 27:61–76.

Perifanis, V., Drosatos, G., Stamatelatos, G., and Efraimidis, P. S. (2023). Fedpoirec: Privacy-preserving federated poi recommendation with social influence. *Information Sciences*, 623:767–790.

Ren, X., Song, M., Haihong, E., and Song, J. (2017). Context-aware probabilistic matrix factorization modeling for point-of-interest recommendation. *Neurocomputing*, 241:38–55.

Sojahrood, Z. B. and Taleai, M. (2021). A POI group recommendation method in location-based social networks based on user influence. *Expert Systems with Applications*, 171:114593.

Sojahrood, Z. B., Taleai, M., and Cheng, H. (2023). Hybrid poi group recommender system based on group type in lbsn. *Expert Systems with Applications*, 219:119681.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., and Bengio, Y. (2018). Graph attention networks. *arXiv preprint arXiv:1710.10903*, page 12.

Wang, Z., Zeng, J., Wen, J., Gao, M., and Zhou, W. (2023). Point-of-interest recommendation using deep semantic model. *Expert Systems with Applications*, pages 1–13.

Wang, Z., Zhu, Y., Zhang, Q., Liu, H., Wang, C., and Liu, T. (2022). Graph-Enhanced Spatial-Temporal Network for Next POI Recommendation. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 16(6):1–21.

Wu, S., Tang, Y., Zhu, Y., Wang, L., Xie, X., and Tan, T. (2019). Session-based recommendation with graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 346–353.

Yu, D., Yu, T., Wang, D., and Shen, Y. (2022). NGPR: A comprehensive personalized point-of-interest recommendation method based on heterogeneous graphs. *Multimedia Tools and Applications*, page 39207–39228.

Zhang, Y., Liu, G., Liu, A., Zhang, Y., Li, Z., Zhang, X., and Li, Q. (2020a). Personalized geographical influence modeling for poi recommendation. *IEEE Intelligent Systems*, 35(5):18–27.

Zhang, Z., Wang, G., and Zhao, X. (2020b). Point-of-interest group recommendation with an extreme learning machine. In *Proceedings of ELM 2018 9*, pages 125–133. Springer.

Zhang, Z., Zhu, J., and Yue, C. (2022). Session-Based Graph Attention POI Recommendation Network. *Wireless Communications and Mobile Computing*, 2022.

Zhao, P., Luo, A., Liu, Y., Zhuang, F., Xu, J., Li, Z., Sheng, V. S., and Zhou, X. (2022). Where to go next: A spatio-temporal gated network for next POI recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 34(5):2512 – 2524.

Zhao, X., Zhang, Z., Bi, X., and Sun, Y. (2020). A new point-of-interest group recommendation method in location-based social networks. *Neural Computing and Applications*, pages 1–12.

Zhu, Q., Wang, S., Cheng, B., Sun, Q., Yang, F., and Chang, R. N. (2018). Context-aware group recommendation for point-of-interests. *IEEE Access*, 6:12129–12144.