Hindawi
Wireless Communications and Mobile Computing
Volume 2020, Article ID 8834450, 14 pages
https://doi.org/10.1155/2020/8834450

Research Article

Leveraging Social Relationship-Based Graph Attention Model for Group Event Recommendation

Guoqiong Liao[1] **and Xiaobin Deng**[1,2]

[1]*School of Information Management, Jiangxi University of Finance and Economics, Nanchang 330013, China*
[2]*Department of Construction Engineering, Jiangxi Water Resources Institute, Nanchang 330013, China*

Correspondence should be addressed to Xiaobin Deng; dengxiaobin83@163.com

Received 9 September 2020; Accepted 13 October 2020; Published 30 October 2020

Academic Editor: Amr Tolba

Recently, event-based social networks(EBSN) such as Meetup, Plancast, and Douban have become popular. As users in the networks usually take groups as an unit to participate in events, it is necessary and meaningful to study effective strategies for recommending events to groups. Existing research on group event recommendation either has the problems of data sparse and cold start due to without considering of social relationships in the networks or makes the assumption that the influence weights between any pair of nodes in the user social graph are equal. In this paper, inspired by the graph neural network and attention mechanism, we propose a novel recommendation model named leveraging social relationship-based graph attention model (SRGAM) for group event recommendation. Specifically, we not only construct a user-event interaction graph and an event-user interaction graph, but also build a user-user social graph and an event-event social graph, to alleviate the problems of data sparse and cold start. In addition, by using a graph attention neural network to learn graph data, we can calculate the influence weight of each node in the graph, thereby generating more reasonable user latent vectors and event latent vectors. Furthermore, we use an attention mechanism to fuse multiple user vectors in a group, so as to generate a high-level group latent vector for rating prediction. Extensive experiments on real-world Meetup datasets demonstrate the effectiveness of the proposed model.

1. Introduction

With the rapid development of the Internet and information technology, social networks are becoming more and more necessary and diversified. As a new type of social networks combining online and offline interactions, event-based social networks (EBSN) have received more and more attentions in recent years. In EBSN, recommending events to groups become an important task since users usually take groups as an unit to participate in events. Recently, a lot of research works on group recommendation have appeared [1–7]. For example, Li et al. [1] proposed a collective matrix factorization and event-user neighbor (CMF-EUN) model to recommend events. Purushotham et al. [2] proposed a Bayesian model based on collaborative filtering for personalized group event recommendation. Yuan et al. [4] introduced the COnsensus Model (COM) probability model to simulate the generation process of group preferences for events. However, these studies do not take into account the social relationship information of users in the group, they have problems of data sparse and cold start. Pham et al. [5] proposed a general graph-based model called HeteRS, in which a random walk method was used to solve different recommendation tasks on EBSN. Yin et al. [6] proposed a general graph-based embedding model (GEM) to solve the event-partner recommendation problem. Although these studies consider the social relationship of users in the group and can alleviate the problems of data sparse and cold start to some extent, when they learn feature, they assume that the influence weights between any pair of users are equal, lessening the accuracy of results they obtain.

In EBSN, there is abundant interactive information and social information, as shown in Figure 1. As it can be seen, EBSN is a heterogeneous social network which includes three kinds of nodes (i.e., groups, users and events), two kinds of interactive information (i.e., group-user interaction and

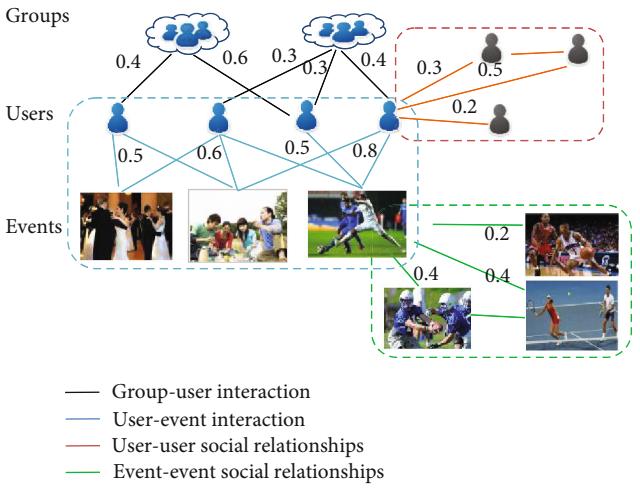


Figure 1: Heterogeneous social network of EBSN.

user-event interaction), and two kinds of social relationship information (i.e., user-user relationship and event-event relationship). The numbers on the interaction line between the users and the events represent the user's preference for scoring events, and the numbers on other line segments in the figure represent the influence weights. For example, as shown in the upper right corner of the figure, a user has three friends, and the influence weights of these three friends on the user are: 0.3, 0.2, 0.5, and the sum of the weights is 1.

In recent years, the deep neural network technology of graph data has made great progress [8]. Now, graph convolution neural networks and graph attention networks have been getting more and more attention, since they can both construct complex heterogeneous graph relational data and also capture the different influence weights of the nodes in the graph. To this end, we propose a novel method leveraging social relationship-based graph attention model (SRGAM) for group event recommendation in EBSN. Specifically, we first build a mathematical model for users in the groups. We construct user latent vectors from two aspects: event aggregation and user social aggregation. We use the attention mechanism to weight and sum the latent vectors of different users to obtain the latent vectors of the group. Then, we build a mathematical model for the events. Graph attention networks are used for event modeling, and event-user interaction graphs and event-event social graphs are used to generate event latent vectors. Finally, we use the group latent vectors and the event latent vectors to perform a dot product to generate a predicted score for recommendation. Our experimental results have demonstrated that the suggested model performs better than state-of-the-art methods on real-world datasets. In summary, the key contributions of the paper are listed as follows:

(i) We propose a novel group event recommendation model SRGAM, which uses a graph neural network and attention mechanism. To the best of our knowledge, this is the first work to recommend events to groups using the heterogeneous graph attention network in EBSN

(ii) We construct a heterogeneous social network for EBSN, which not only contain a user-event interaction graph and an event-user interaction graph but also has a user-user social graph and an event-event social graph, to further alleviate the problems of data sparse and cold start

(iii) We use a graph attention neural network for learning graph data and calculate the different influence weights between the nodes in the heterogeneous network, which facilitate to generate more reasonable latent user vectors and event vectors. The attention mechanism is also used to perform weighted fusion on different user vectors in the group to generate high-level group latent vectors

(iv) We conduct extensive experiments on the real-world datasets to demonstrate the effectiveness of our proposed method

The remainder of this paper is organized as follows. Section 2 introduces our model framework. Section 3 shows the model realization. The comparison experiment on the real-world datasets is presented in Section 4. Related work is reviewed in Section 5. Finally, we conclude the paper and discuss the future work in Section 6.

2. The Proposed Model

In this section, we first introduce the definitions and notations used in this paper and then give an overview about the model framework.

2.1. Definitions and Notations. Let $G = \{g_1, g_2, \cdots, g_{|G|}\}$, $U = \{u_1, u_2, \cdots, u_{|U|}\}$, and $E = \{e_1, e_2, \cdots, e_{|E|}\}$ be the sets of groups, users, and events in EBSN, respectively, where $|G|$, $|U|$, and $|E|$, are the numbers of groups, users, and events, respectively, and g_i is composed of the users from U.

$\mathbf{M} \in \mathbb{R}^{|G|\times|E|}$ is a group-event rating matrix, which is also called a group-event graph. The value in the matrix, denotes as r_{ij}, is the rating score that g_i gives to e_j. We employ 0 to represent an unknown rating from g_i to e_j.

$\mathbf{N_1} \in \mathbb{R}^{|U|\times|E|}$is a user-event rating matrix, which is also called a user-event graph. s_{ij} is the score value of u_i for e_j. If user u_i did not participate in the event e_j, then $s_{ij} = 0$.

$\mathbf{N_2} \in \mathbb{R}^{|E|\times|U|}$is an event-user rating matrix, which is also called an event-user graph. $\mathbf{N_2}$ is the transposed matrix of $\mathbf{N_1}$, so s_{ji} is the score value of u_i for e_j. In addition, users can establish social relationships with each other.

$\mathbf{H} \in \mathbb{R}^{|U|\times|U|}$ is a user-user relationship matrix, which is also called a user-user social graph, where $H_{ij} = 1$ indicates that there is a friend relationship between u_i and u_j, and $H_{ij} = 0$ indicates that there is no friend relationship between them.

$\mathbf{T} \in \mathbb{R}^{|E|\times|E|}$ is an event-event relationship matrix, which is also called an event-event social graph, where $T_{ij} = 1$ indicates that there is an influence relationship between e_i and e_j, and $T_{ij} = 0$ indicates that there is no influence relationship between them.

Let A_i be the set of social friends of user u_i, B_i be the set of events which user u_i participated in, C_j be the set of users who participated in event e_j, D_j be the set of social events of event e_j, and F_i be the set of users who are in the group g_i. Given $\mathbf{M}$, $\mathbf{N_1}$, $\mathbf{N_2}$, $\mathbf{H}$, and $\mathbf{T}$, the goal of the paper is to predict the missing score value in $\mathbf{M}$. We use $\mathbf{p}_i \in \mathbb{R}^d$ to represent the embedding vector of user u_i and $\mathbf{q}_j \in \mathbb{R}^d$ to represent the embedding vector of the event e_j, where d is the dimension of the embedding vector.

The notations used in the paper are listed in Table 1.

2.2. Model Framework. The framework of our proposed leveraging social relationship-based graph attention model (SRGAM) is shown in Figure 2, which consists of three modules, i.e., group modeling, event modeling, and rating prediction.

The first module is group modeling, which is used to learn the latent vector of the groups. This module is divided into two stages: user modeling and user fusion. During user modeling, the user-event graph and the user-user social graph are used to learn user representation from two different graph perspectives. Therefore, we introduce two kinds of aggregations to process these two different graphs separately. The one is event aggregation, which can understand users through the interaction between the users and events in the user-event graph. The other is social aggregation, and the relationship between users in a social graph can model users from a social perspective. Then, we intuitively obtain user latent vectors by combining information from the event space and social space. At last, the attention-weighted summation of different user latent vectors is performed to obtain the latent vector of the group.

The second module is event modeling, used to learn the latent vector of events. Due to the mutual influence between different events, similar to the user-user social graph, we propose the event-event social graph. This part uses the event-user graph and event-event social graph to learn the representation of events from two different graph perspectives.

The last module is the rating prediction, to learn model parameters for prediction by integrating the information both of the group modeling and event modeling.

3. Model Inference and Learning

In this section, we will introduce the implement techniques of the three modules in detail.

3.1. Group Modeling. The goal of group modeling is to generate the latent vector of each group $g_i \in G$, represented as $\mathbf{g}_i \in \mathbb{R}^d$. This component is divided into two stages: user modeling and user fusion. The user modeling stage involves integrating the information of the user-event interactive graphs and the user-user social graphs. As shown in the upper part of Figure 2, there are two kinds of aggregations, i.e., event aggregation and social aggregation, to learn $u_i \in U$'s latent vectors $\mathbf{u}_i^E \in \mathbb{R}^d$ in the event space and $\mathbf{u}_i^S \in \mathbb{R}^d$ in the social space. Then, we combine $\mathbf{u}_i^E$ and $\mathbf{u}_i^S$ to form u_i's final latent vector $\mathbf{u}_i$. In the user fusion stage, the latent vectors $\mathbf{u}_i$ of different users in the group are weighted and summed through an attention mechanism, to generate the group latent vector $\mathbf{g}_i$ finally.

3.1.1. User Modeling Stage. This stage includes three parts: event aggregation, user social aggregation, and user latent vectors learning.

(1) Event Aggregation. The user-event graph contains not only the user's interaction with the event but also the user's opinion on the event (or the user's rating of the event). We provide a way to jointly capture the interactions and opinions in the user-event graph to learn the user latent vectors $\mathbf{u}_i^E$ in the event space. To represent this aggregation mathematically, we use the following function:

$$\boldsymbol{u}_i^E = \sigma\left(\boldsymbol{W} \cdot \mathrm{AggreE}\left(\left\{\boldsymbol{x}_{ij}, \forall j \in B_i\right\}\right) + \boldsymbol{b}\right), \tag{1}$$

where σ represents the nonlinear activation function, AggreE() represents the event aggregation function, and $\mathbf{x}_{ij}$ is a representation vector of interaction and opinions between u_i and e_j. B_i is the set of events participated by the user u_i. $\boldsymbol{W}$ and $\mathbf{b}$ represent the weight and bias in the neural network, respectively.

For an event, the user can express his/her opinion (or rating), denoted as r. These opinions about the event can capture the user's preference for the event, which can help model the latent vector of the event space user. To model opinions, for each opinion r, we introduce an opinion embedding vector $\mathbf{s}_r \in \mathbb{R}^d$, which represents each opinion r as a dense vector representation. For the interaction between user u_i and event e_j with r, we combine the event embedding vector $\mathbf{q}_j$ and the opinion embedding vector $\mathbf{s}_r$ through a multilayer perceptron (MLP) to generate an opinion-aware interaction representation $\mathbf{x}_{ij}$. MLP takes the series of event embedding $\mathbf{q}_j$ and opinion embedding $\mathbf{s}_r$ as input, and the output of MLP is an opinion perception representation $\mathbf{x}_{ij}$ of the interaction between u_i and e_j, as follows:

$$\boldsymbol{x}_{ij} = \mathrm{MLP}\left(\left[\boldsymbol{q}_j \oplus \boldsymbol{s}_r\right]\right), \tag{2}$$

where $\oplus$ represents the connection of two vectors.

For the aggregation function AggreE(), the vector in $\{\mathbf{x}_{ij}, \forall j \in B_i\}$ is generally averaged; then, equation (1) becomes

$$\mathbf{u}_i^E = \sigma\left(\mathbf{W} \cdot \left\{\sum_{j\in B_i} \alpha_i \mathbf{x}_{ij}\right\} + \mathbf{b}\right), \tag{3}$$

where $\alpha_i = 1/|B_i|$, and all events use a fixed equal weight value. This method assumes that all events interacting with

Table 1: Notation.

Symbols	Descriptions
G	The set of group g_i $(i = 1, 2, \cdots \mid G \mid)$
U	The set of user u_i $(i = 1, 2, \cdots \mid U \mid)$
E	The set of event e_j $(j = 1, 2, \cdots \mid E \mid)$
M	The group-event graph
N_1	The user-event graph
N_2	The event-user graph
H	The user-user graph
T	The event-event graph
A_i	The set of social friends of user u_i
B_i	The set of events interacted by user u_i
C_j	The set of users who participated in event e_j
D_j	The set of social events of event e_j
F_i	The set of users who are in the group g_i
p_i	The embedding of user u_i
q_j	The embedding of event e_j
s_r	The embedding of the rating value r, $r \in [0, 1]$
$\mathrm{u_i}^{\mathrm{E}}$	The event space user latent factor from event set B_i of user u_i
u_i^S	The social space user latent factor from social friends set A_i of user u_i
u_i	The user latent factor of user u_i, combining from event space $\mathbf{u}_i^E$ and social space $\mathbf{u}_i^S$
x_{ij}	The interaction and opinions representation of event e_j for user u_i
e_j^U	The user space event latent factor from user set C_j of event e_j
e_j^S	The social space event latent factor from social events set D_j of event e_j
e_j	The event latent factor of event e_j, combining from user space $\mathbf{e}_j^U$ and social space $\mathbf{e}_j^S$
y_{ji}	The interaction and opinions representation of user u_i for event e_j
g_i	The group latent vector of g_i
α_{ij}	The attention weight of event e_j in contributing to $\mathbf{u_i}^{\mathrm{E}}$
β_{ij}	The social attention weight of user u_j in contributing to $\boldsymbol{u}_i^S$
θ_{ji}	The attention weight of user u_i in contributing to $\boldsymbol{e}_j^U$
γ_{ji}	The social attention weight of event e_i in contributing to e_j^S
μ_{ij}	The attention weight of user u_j in contributing to g_i
r_{ij}	The rating value of event e_j by group g_i
r_{ij}'	The predicted rating value of event e_j by group g_i
$\oplus$	The concatenation operator of two vectors
W	The weight in neural network
b	The bias in neural network

a user have the same effect on the user. However, since the impact of the interaction on the user may vary greatly, this may not be reasonable. Therefore, we need to allow interactions to make different contributions to users' latent factors by assigning a weight to each interaction.

In order to alleviate the shortcomings of the mean-based aggregation method, we use an attention mechanism to learn how different events affect users.

$$\mathbf{u}_i^E = \sigma\left(\mathbf{W} \cdot \left\{\sum_{j \in B_i} \alpha_{ij} \mathbf{x}_{ij}\right\} + \mathbf{b}\right), \tag{4}$$

where α_{ij} is the weight of event e_j to user u_i's event space

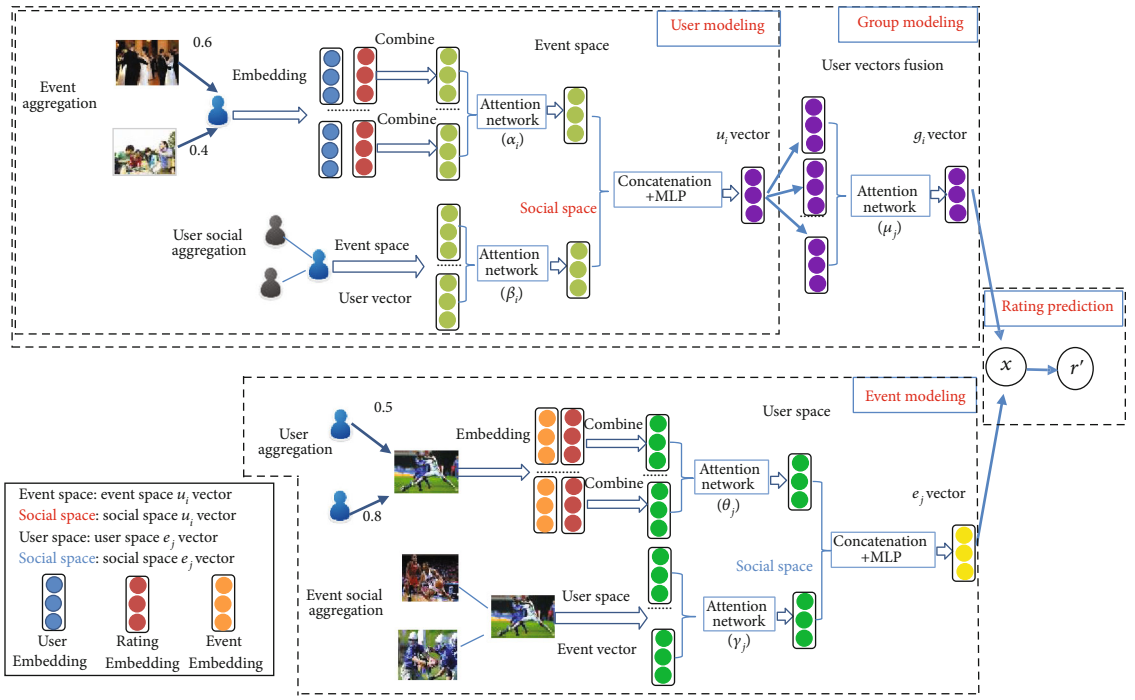


FIGURE 2: The overall architecture of the proposed model. It contains three major components: group modeling, event modeling, and rating prediction. α_i, β_i, θ_j, γ_j, and μ_i are the attention weights. Note that the numbers on the edges of the graph denote the rating score.

latent vector. We use an attention neural network to learn α_{ij}. The input of the attention network is the opinion perception representation $\mathbf{x}_{ij}$ of the user interaction and the embedding vector $\mathbf{p}_i$ of the target user. The attention network is defined as follows:

$$\alpha'_{ij} = \sigma\left(\mathbf{W} \cdot \left[\mathbf{x}_{ij} \oplus \mathbf{p}_i\right] + \mathbf{b}\right) \tag{5}$$

The final attention weight is obtained using the Softmax function to normalize the above attention score, which can be interpreted as the contribution of the event interaction to the user u_i's event space user latent vector $\mathbf{u}_i^E$, specifically:

$$\alpha_{ij} = \frac{\exp\left(\alpha'_{ij}\right)}{\sum_{j \in B_i} \exp\left(\alpha'_{ij}\right)}. \tag{6}$$

(2) User Social Aggregation. The user's preferences are similar to or affected by his/her social friends. Therefore, we need to incorporate social information to more accurately simulate user latent factors. At the same time, the strength of the connection between users can further influence the user's behavior from the social graph. In other words, the learning of user latent factors in social space should consider the heterogeneity of social relations. Therefore, we introduce an attention mechanism to select representative social friends to characterize the user's social information and then summarize the information. In order to represent user latent vectors from social perspective, a user latent vector in social space, i.e., $\mathbf{u}_i^S \in \mathbb{R}^d$, is introduced for u_i. Specifically, $\boldsymbol{u}_i^S$ is a vector which aggregates the latent vectors in event space of the users in friend set A_i of u_i, as follows:

$$\boldsymbol{u}_i^S = \sigma\left(\boldsymbol{W} \cdot \mathrm{AggreUS}\left(\left\{\boldsymbol{u}_j^E, \forall j \in A_i\right\}\right) + \boldsymbol{b}\right), \tag{7}$$

where AggreUS() represents the aggregation function of the user's social friends.

As with event aggregation, for the aggregation function AggreUS(), the vector in $\{\mathbf{u}_j^E, \forall j \in A_i\}$ is generally averaged. Then, equation (7) becomes

$$\mathbf{u}_i^S = \sigma\left(\mathbf{W} \cdot \left\{\sum_{j \in A_i} \beta_i \mathbf{u}_j^E\right\} + \mathbf{b}\right), \tag{8}$$

where $\beta_i = 1/|A_i|$, and a fixed equal weight value is used for all social friends. This method assumes that all friends who have social interactions with the user have the same impact on the user. This is obviously unreasonable, so we also use the attention mechanism to learn the different influence weights of different friends on the user, as follows:

$$\mathbf{u}_i^S = \sigma\left(\mathbf{W} \cdot \left\{\sum_{j \in A_i} \beta_{ij} \mathbf{u}_j^E\right\} + \mathbf{b}\right), \tag{9}$$

$$\beta'_{ij} = \sigma\left(\mathbf{W} \cdot \left[\mathbf{u}_j^E \oplus \mathbf{p}_i\right] + \mathbf{b}\right), \tag{10}$$

$$\beta_{ij} = \frac{\exp\left(\beta'_{ij}\right)}{\sum_{j \in A_i} \exp\left(\beta'_{ij}\right)}, \tag{11}$$

where $\beta_{ij}{}'$ is the attention weight, and β_{ij} is the normalized attention weight.

(3) User Latent Vector Learning. In order to learn better user latent factors, since user social graphs and user-event graphs provide information about users from different perspectives, and event space user latent vectors and social space user latent vectors need to be considered together. We propose to connect them and learn through the multilayer perceptron (MLP) and finally obtain the user latent vector $\boldsymbol{u}_i$.

$$\boldsymbol{u}_i = \mathrm{MLP}\left(\boldsymbol{u}_i^E \oplus \boldsymbol{u}_i^S\right). \tag{12}$$

3.1.2. User Fusion Stage. A group contains several users, and the latent vectors of multiple users are fused to form the latent vector of the group. Because different users have different positions and weights in the group, we use the attention mechanism to learn the different weights of users and then use the weights and user latent vectors to perform weighted summation to obtain the group latent vector. The corresponding formula is as follows:

$$\mathbf{g}_i = \sum_{j \in F_i} \mu_{ij} \mathbf{u}_j, \tag{13}$$

$$\mu'_{ij} = \sigma\left(\mathbf{W} \cdot \mathbf{u}_j + \mathbf{b}\right), \tag{14}$$

$$\mu_{ij} = \frac{\exp\left(\mu'_{ij}\right)}{\sum_{j \in F_i} \exp\left(\mu'_{ij}\right)}, \tag{15}$$

where $\mu_{ij}{}'$ is the attention weight, and μ_{ij} is the normalized attention weight.

3.2. Event Modeling. As shown in the lower part of Figure 2, event modeling is used to learn the event latent vector $\boldsymbol{e}_j$. Event modeling also includes three parts: user aggregation, event social aggregation, and event latent vectors learning.

3.2.1. User Aggregation. We adopt the similar method to learn event latent vectors in user space through user aggregation. The information to be aggregated here comes from the user set C_j that interacts with the event e_j. Even for the same event, different users may express different opinions during the user-event interaction. These opinions from different users can capture the characteristics of the same event, which can help model the event latent vector. We connect the user vector interacting with the event and the user's opinion vector on the event and then learn through MLP to get the interactive user representation $\mathbf{y}_{ji}$ of opinion perception. The formula is as follows:

$$\boldsymbol{y}_{ji} = \mathrm{MLP}([\boldsymbol{p}_i \oplus \boldsymbol{s}_r]). \tag{16}$$

Then, in order to learn the event latent vector $\boldsymbol{e}_j{}^U$, we also aggregate different users interacting with the event e_j. The aggregation function is *AggreU()*. It mainly aggregates the user's opinion-aware interaction representation in $\{\boldsymbol{y}_{ji}, \forall i \in C_j\}$, as follows:

$$\boldsymbol{e}_j^U = \sigma\left(\boldsymbol{W} \cdot \mathrm{AggreU}\left(\left\{\boldsymbol{y}_{ji}, \forall i \in C_j\right\}\right) + \boldsymbol{b}\right). \tag{17}$$

In addition, we also use the attention mechanism to learn the different influence weights of different users on the same event.

$$\mathbf{e}_j^U = \sigma\left(\mathbf{W} \cdot \left\{\sum_{i \in C_j} \theta_{ji} \mathbf{y}_{ji}\right\} + \mathbf{b}\right), \tag{18}$$

$$\theta'_{ji} = \sigma\left(\mathbf{W} \cdot \left[\mathbf{y}_{ji} \oplus \mathbf{q}_j\right] + \mathbf{b}\right), \tag{19}$$

$$\theta_{ji} = \frac{\exp\left(\theta'_{ji}\right)}{\sum_{i \in C_j} \exp\left(\theta'_{ji}\right)}, \tag{20}$$

where $\theta_{ji}{}'$ is the attention weight, and θ_{ji} is the normalized attention weight.

3.2.2. Event Social Aggregation. Just as a social graph can be constructed between users, a social graph of events can also be constructed between events. We assume that if the similarity between a pair of events exceeds a certain threshold, it is considered that the pair have an event social relationship. Specifically, the social space event latent vector $\mathbf{e}_j{}^S$ of the event e_j is a user space event latent vector that aggregates the social event set D_j of the event e_j as follows:

$$\boldsymbol{e}_j^S = \sigma\left(\boldsymbol{W} \cdot \mathrm{AggreES}\left(\left\{\boldsymbol{e}_i^U, \forall i \in D_j\right\}\right) + \boldsymbol{b}\right), \tag{21}$$

where AggreES() represents the aggregation function of the social relationship of the event.

Events that have friendships with one event have different influence weights on this one event. Therefore, we also use an attention mechanism to learn the influence weights for event social aggregation and perform weighted summation for the purpose of aggregation.

$$\mathbf{e}_j^S = \sigma\left(\mathbf{W} \cdot \left\{\sum_{i \in D_j} \gamma_{ji} \mathbf{e}_j^U\right\} + \mathbf{b}\right), \tag{22}$$

$$\gamma'_{ji} = \sigma\left(\mathbf{W} \cdot \left[\mathbf{e}_j^U \oplus \mathbf{q}_j\right] + \mathbf{b}\right), \tag{23}$$

$$\gamma_{ji} = \frac{\exp\left(\gamma'_{ji}\right)}{\sum_{i \in D_j} \exp\left(\gamma'_{ji}\right)}, \tag{24}$$

where $\gamma_{ji}{}'$ is the attention weight, and γ_{ji} is the normalized attention weight.

3.2.3. Event Latent Vectors Learning. In order to learn better event latent factors, since event social graphs and event-user graphs provide information about events from different perspectives, we connect the user space event latent vectors and social space event latent vectors and learn through MLP to obtain the event latent vector $\boldsymbol{e}_j$.

$$\mathbf{e}_j = MLP\left(\mathbf{e}_j^U \oplus \mathbf{e}_j^S\right). \tag{25}$$

3.3. Rating Prediction and Model Training. In the work of this paper, we use the score prediction as a recommendation task. Specifically, we use the latent vector $\mathbf{g}_i$ of the group and the latent vector $\mathbf{e}_j$ of the event to perform a dot product to generate a predicted score:

$$r'_{ij} = \boldsymbol{g}_i \cdot \boldsymbol{e}_j. \tag{26}$$

In order to estimate the parameters of the model, we need to specify an objective function for optimization. Since the task we focus on in this work is the rating prediction, the objective function is expressed as

$$\text{Loss} = \frac{1}{|N|}\sum_{i,j\in N}\left(r'_{ij} - r_{ij}\right)^2, \tag{27}$$

where $|N|$ is the number of observed scores, r_{ij} is the actual score of group g_i on event e_j, and $r_{ij}{}'$ is the predicted score of group i on event j.

To optimize the objective function, AdamOptimizer as the optimizer is used to optimize the mean square error function. Our model contains three embeddings: event embedding, $\mathbf{q}_j$, user embedding, $\mathbf{p}_i$, and opinion embedding, $\mathbf{s}_r$. During the training phase, they are randomly initialized and learned together. Because the original features are very large and sparse, we do not use one-hot encoding vectors to represent each user and each event, but we embed high-dimensional sparse features into low-dimensional latent space so that the model can be more easily trained. To alleviate the problem of overfitting in optimizing deep neural network models, we adopt the dropout strategy to randomly drop some neurons during the training process.

4. Experiments

In this section, we compare the experimental results of SRGAM with five baseline methods and three variant methods on two real-world datasets. Generally speaking, our experimental goal is to answer the following research questions (RQ):

(i) RQ1: how does SRGAM perform as compared to existing advanced methods?

(ii) RQ2: what are the effects of each component in our method on performance?

(iii) RQ3: how do the hyper-parameters affect our model's performance?

4.1. Experimental Settings

4.1.1. Datasets. We conducted experiments on the real-world datasets from two different cities in the event social network Meetup (https://www.meetup.com/): Chicago and San Diego. The actual score data of a group on an event is a floating point number between 0 and 5; data with a score of 0 was discarded since the 0 score most likely occurred as a result of a user not participating. For users in the Meetup dataset do not explicitly rate events, we calculate the similarity based on the theme of each user and the theme of each participating event and normalize it to obtain a value in the range of [0,1], which is used as the user's score data for the event. There is also no explicit friend relationship in the Meetup dataset, so we consider two users who have participated in three or more events together as friend relationship. We assume that if the degree of acquaintance between two events exceeds 0.4, then it is considered that there is an influence relationship between them. The statistics of the two datasets are shown in Table 2.

4.1.2. Evaluation Metrics. We use two indicators to evaluate the model performance: root mean square error (RMSE) and mean absolute error (MAE). The smaller the RMSE and MAE indicators, the higher the accuracy of the model.

$$\text{RMSE} = \sqrt{\frac{1}{|T|}\sum_{(i,j)\in T}\left(r'_{ij} - r_{ij}\right)^2}, \tag{28}$$

$$\text{MAE} = \frac{1}{|T|}\sum_{(i,j)\in T}\left|r'_{ij} - r_{ij}\right|, \tag{29}$$

where (i, j) represents group i and event j, $|T|$ is the number of scores in the test set, $r_{ij}{}'$ is the score predicted by the model, and r_{ij} is the actual score of the test set.

4.1.3. Baselines. In order to evaluate the performance, our model is compared with the other five methods and three variant methods. These methods are described in detail below.

(i) GARec [9]: the model uses genetic algorithms to learn the interactions between users in a group and can capture the different importance of users in the group

(ii) HeteRS [5]: this method proposes to construct a heterogeneous graph model, that makes full use of the interaction between groups, users, events, and tags and complete three different recommendation tasks: recommend groups to users, recommend tags to groups, and recommend events to users

(iii) RRWR-S [10]: this method constructs a heterogeneous graph to represent the relationships between various entities and social networks and then uses restart to perform a reverse random walk to obtain user-event proximity values from the constructed graph

Table 2: Statistics of the datasets.

Dataset	Chicago	San Diego
Total groups	5675	8462
Total events	2365	3529
Total users	18164	35543
Ratings of groups to events	41427	47387
Ratings of users to events	148944	231029
Users social connections	58124	81748
Events social connections	13244	21526

(iv) AGREE [11]: for the first time, the attention mechanism in the neural network is used for group recommendation. The fusion strategy is dynamically learned based on the input data. At the same time, user-item interaction information is introduced to improve the effect of group recommendation

(v) MoSAN [12]: this model introduces the attention mechanism into group recommendation, and it proposed to use an attention mechanism to obtain the influence of each user in the group. The model can learn the influence weight of each user in the group, and consider different contexts, and then recommend items to groups based on their members' weighted preferences. This method can model complex group decision-making processes

(vi) SRGAM1: this is a variant of the model in this paper, which deletes the user's social relationships and retains the event's social relationships in the model. It recommends events to groups without considering the user's social relationships

(vii) SRGAM2: this is a variant of the model in this paper, which deletes the event's social relationships and retains the user's social relationships in the model. It is that recommends events to groups without considering the event's social relationships

(viii) SRGAM3: this is a variant of the model in this paper, which removes the attention mechanism in the model. It assumes that the impact of different events on users, the impact of different users on events, the impact of social interaction, and the impact of different users on groups are equal

4.1.4. Parameter Settings. For each dataset, we use x% as the training set to learn the parameters, (1-x%)/2 as the verification set to adjust the hyper parameters, and (1-x%)/2 as the test set, where x% takes the values {50%, 70%, 90%}. For the embedding size d, we test the value of [8, 16, 32, 64, 128]. The batch size and learning rate were searched in [32, 64, 128, 512] and [0.0001, 0.001, 0.01, 0.1], respectively. When training with neural networks, we uniformly used three hidden layers and ReLu as activation functions, the training period Epoch was 20, and the group size tested was the number of members of [1–20].

4.2. Overall Performance Comparison (RQ1). First, we compare the recommendation performance of our model with the other five models. Figure 3 shows the RMSE and MAE on the two datasets of Chicago and San Diego. We have the following main findings:

(i) The performance of the GARec model is the worst. Although this method considers the different weights of users in the group, it only uses the score information and ignores the user-user social relationship information, so it leads to poor performance

(ii) The HeteRS and RRWR-S models are better than the GARec model, because these two methods use a graph structure to model social relationships. The GARec model only uses rating information and ignores user-user social relationship information. These results indicate that social relationship information is necessary in group recommendation

(iii) The AGREE and MoSAN models are better than the HeteRS and RRWR-S models. Although the first two do not consider the user's social relationship information, they use the attention mechanism and neural network for deep learning, which can learn better groups and events. This shows that the attention mechanism and neural network play an important role in group recommendation

(iv) Our model SRGAM is superior to the other five baseline methods. Our model combines user-user social relationships and event-event social relationships. At the same time, our model also uses attention mechanisms and neural networks for learning to get a better representation of groups and events

Overall, the comparison results of various model methods show that (1) social information is helpful for group recommendation, (2) attention mechanism and neural network can improve recommendation performance, and (3) our model is better than other baseline methods.

4.3. Ablation Study(RQ2). In this section, we conduct ablation research from two aspects: social relationship ablation and attention mechanism ablation.

4.3.1. Social Relationship Ablation. SRGAM1 and SRGAM2 are two variants of the model in this paper, they delete user social relationship and event social relationship, respectively. Figure 4 shows the comparison of the RMSE and MAE indicators of the SRGAM1, SRGAM2, and SRGAM models on the two datasets. We have the following findings:

(i) The performance of the model SRGAM in this paper is better than that of the two models of the variant, indicating that both the user's social relationship information and the event social relationship information play a positive role in group event recommendation

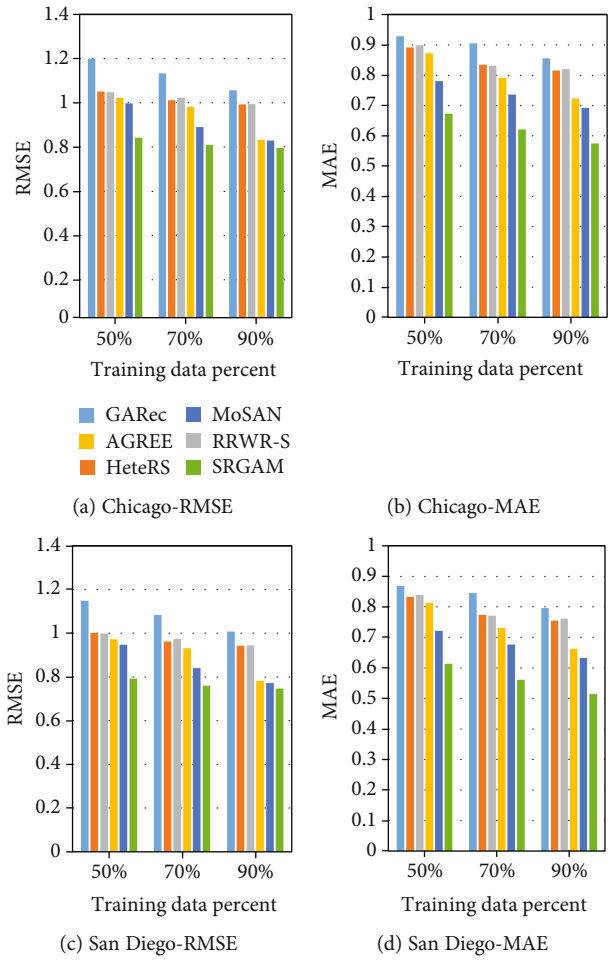


Figure 3: Performance comparison of different methods on two datasets.

(ii) The performance of the SRGAM2 model is better than that of the SRGAM1 model, which shows that the user's social relationship information is more important than the event social relationship information to a certain extent

4.3.2. Attention Mechanism Ablation. As shown in the previous chapter, SRGAM3 is another variant of the model in this paper, one which removes the attention mechanism in the model. Since there are five places in the model that use the attention mechanism, five variants were produced: SRGAM3-α, SRGAM3-β, SRGAM3-θ, SRGAM3-γ, and SRGAM3-μ. They are defined as follows:

(i) SRGAM3-α: when the user's events are aggregated, the event attention α of the model SRGAM is removed. This variant method uses the average aggregation method for event aggregation; that is, the impact of each event on the user is considered equal

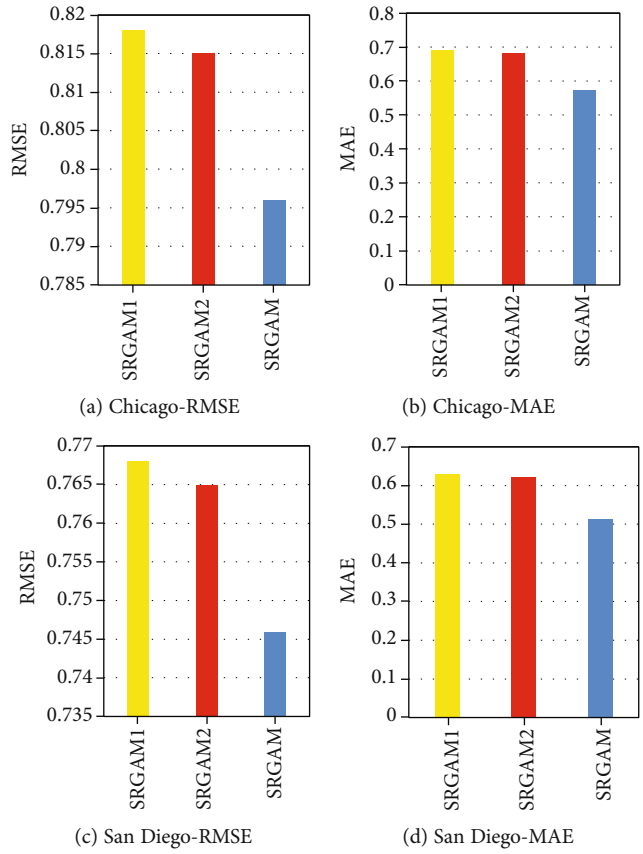
(ii) SRGAM3-β: when the user's social relationships are aggregated, the social attention β of the model SRGAM is removed. This variant method uses the average aggregation method for user social aggregation; that is, the impact of each friend on the user is considered equal

(iii) SRGAM3-θ: when the event's users are aggregated, the user attention θ of the model SRGAM is removed. This variant method uses the average aggregation method for user aggregation; that is, the impact of each user on the same event is considered equal

(iv) SRGAM3-γ: when the event's socials are aggregated, the social attention γ of the model SRGAM is removed. This variant method uses the average aggregation method for event social aggregation. That is, the impact of each event on the same event is considered equal

(v) SRGAM3-μ: when the users of the group are aggregated, the user attention μ of the model SRGAM is removed. This variant method uses the average aggregation method for user vector aggregation; that is, the impact of each user on the group is considered equal

Figure 5 shows the performance comparison of this model and the five attention variant models. From the results, we have the following findings:



(a) Chicago-RMSE (b) Chicago-MAE

(c) San Diego-RMSE (d) San Diego-MAE

Figure 4: Effect of social network on two datasets.

(i) The impact of multiple events involving a user on the latent vector of the user in the event space is not equal. Similarly, the impact of multiple users participating in an event on the latent vector of the event in the user space is not equal. Based on this assumption, our model was implemented by using two different attention mechanisms (α and θ). From the experimental results, SRGAM3-α and SRGAM3-θ did not perform as well as SRGAM. The results show that the attention mechanism plays a role in event aggregation and user aggregation

(ii) The influence of the user's friends on the user is not equal. Similarly, the influence of the event's friends on the event is not equal. Based on this assumption, our model was implemented by using two different attention mechanisms (β and γ). From the experimental results, SRGAM3-β and SRGAM3-γ did not perform as well as SRGAM. The results show that the attention mechanism plays a role in user social aggregation and event social aggregation

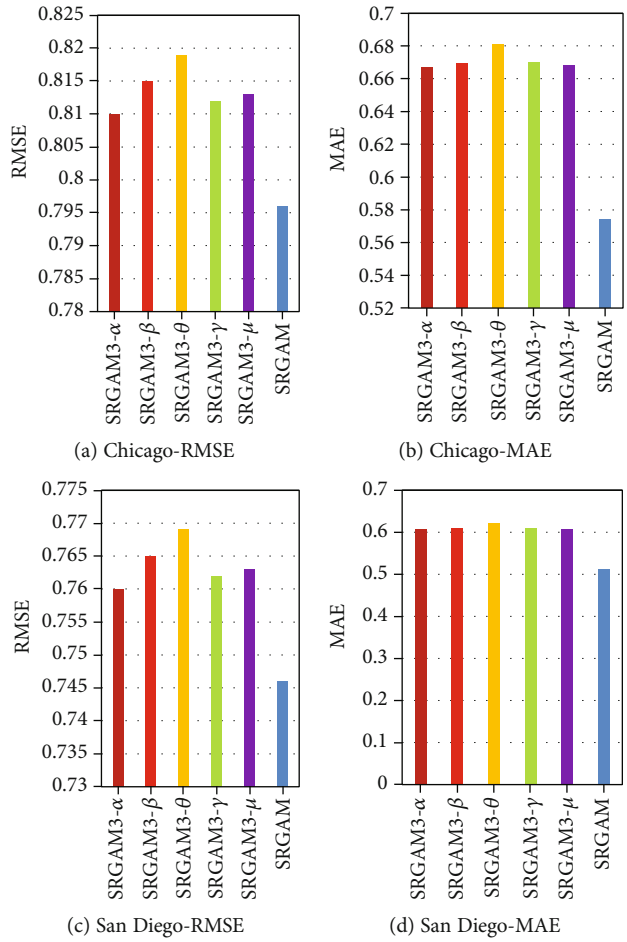
(iii) Different users in a group have different influences on group decisions. Based on this assumption, our model was implemented by using the attention mechanism (μ). From the experimental results, SRGAM3-μ is not perform as well as SRGAM. The results show that the attention mechanism plays a role in the aggregation of users in the group

To sum up, SRGAM can capture various influence weights in users, groups, and events through application of an attention mechanism, which improves the recommendation performance.

4.4. Effect of Hyper Parameters on the Model Performance (RQ3). There are two main hyper parameters in the SRGAM model: embedding size and group size.

4.4.1. Embedding Size. We select five values, {8, 16, 32, 64, 128}, for testing. As shown in Figure 6, for the two datasets, the small embedding size cannot fully express the characteristics of users, events, and ratings, while an excessively large embedding size will lead to overfitting of the model and a decrease in learning efficiency. Therefore, we set the embedding size to 32.

4.4.2. Group Size. When we train the model, we try four different group sizes (the number of members in the group): {1-5, 6-10, 11-15, 16-20} and found that the model achieve best performance when the group size was 6-10 users on two datasets. As shown in Figure 7, a large group size causes the model complexity to increase and reduces recommendation performance, while a small group size reduces the effect of



(a) Chicago-RMSE (b) Chicago-MAE

(c) San Diego-RMSE (d) San Diego-MAE

Figure 5: Effect of attention mechanism on two datasets.

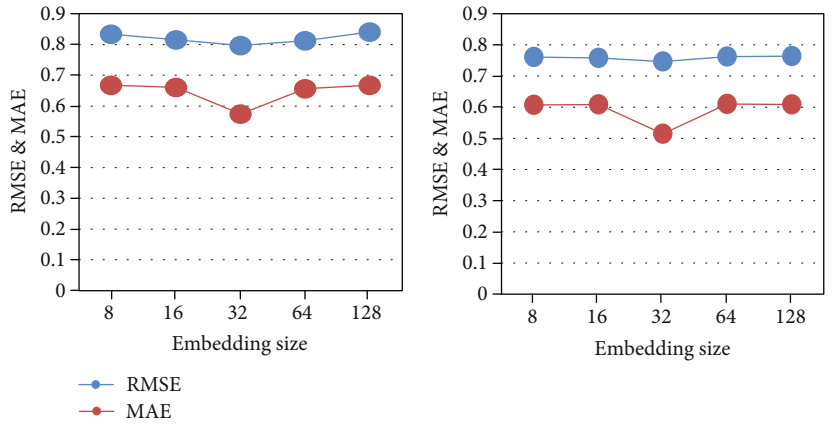
the attention mechanism and reduces the recommendation effect.

5. Related Work

In this section, we discuss some related work including general group recommendation, group recommendation based on the graph model, graph neural network, and graph attention network.

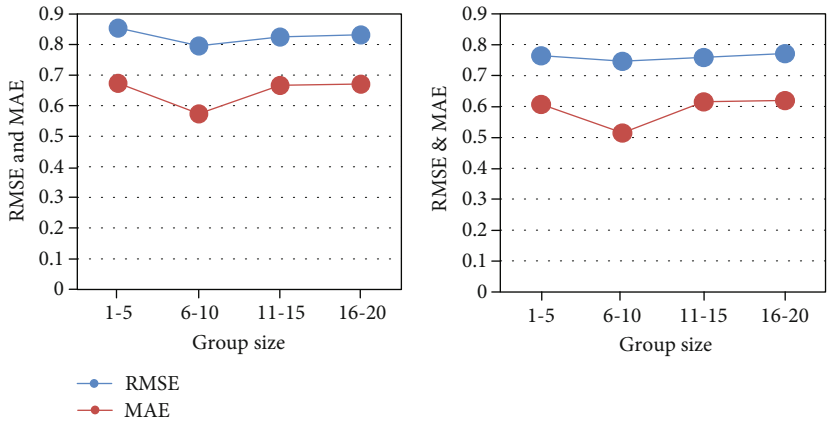
5.1. General Group Recommendation. The key issue of the general group recommendation is preference fusion. The preference fusion methods can be divided into two categories: model fusion and recommendation fusion [17]. Yu et al. [13] proposed a model fusion method based on item feature scoring. Yuan et al. [4] proposed a probabilistic model named COM to simulate the generation process of group preferences for events. Kagita et al. [14] proposed a fusion method based on priority sequence mining and a virtual user model. O'Connor et al. [16] used the least painful strategy to fuse a user recommendation list and obtain a group recommendation list. Chen et al. [9] used a genetic algorithm to optimize the weight of group members and proposed a group recommendation method combining collaborative filtering and genetic algorithm. Naamani-Dery et al. [15] show that it is possible to find a balance between the size of the recommendation list and the cost of group preference extraction and thereby, reduce the size of the group recommendation list using an iterative preference extraction method. Cao et al. [11] proposed the AGREE model. Where for the first time, the attention mechanism in the neural network was used for group recommendation. Lucas et al. [12] proposed the MoSAN model, which also introduced the attention mechanism to group recommendation. It proposes using an attention mechanism to identify the influence level of each user in the group. However, as these methods seldom consider social relationship information, they have to be confronted with the problems of data sparse and cold start.

5.2. Group Recommendation Based on the Graph Model. The recommendation method based on a graph model is widely used in the recommendation field. Meng et al. [20] divided graph-based social recommendation methods into graph-based recommendation methods and link prediction methods. The graph is the most natural and direct representation of social network relationships in EBSN. There have been many studies on group recommendation in EBSN based on graph models, including [5–7, 10, 18, 19, 21]. Li et al. [10,



(a) Chicago (b) San Diego

Figure 6: Effect of embedding size on two datasets.



(a) Chicago (b) San Diego

Figure 7: Effect of group size on two datasets.

18] and Liu et al. [19] constructed a heterogeneous graph to express the interaction between different types of entities in EBSN. The perform random walks on the graph to obtain candidate events with high convergence probability. Pham et al. [5] proposed a general graph-based model HeteRS, in which a random walk method was used to solve the different recommendation tasks on EBSN. Yin et al. [6] proposed a general graph-based embedding model (GEM) to solve the event-partner recommendation problem. Liu et al. [21] constructed a primary graph (PG) based on the entities and their relationships in the EBSN and calculated the user-event similarity score by applying the random walk restart algorithm on the PG. However, these methods assume that the social impact of each friend on users is equal in importance.

5.3. Graph Neural Network and Graph Attention Network. Graph neural network (GNN) and graph attention network (GAN) have become a hotspot in the field of deep learning in recent years. Recently, the related work has focused on using CNN to model more general graph structure data [8, 22–25]. Specifically, Thomas et al. [8] proposed a graph convolution network for semisupervised graph classification. The model learns the node representation by using node attributes and graph structure. It consists of multiple graph convolutional layers, and each layer updates the node representation using the representation of the current node and its neighbors. Through this process, it can capture the dependencies between nodes. However, in the original formula, when updating the node representation, all neighbors are given a static weight. Existing studies use GAN to solve social impact analysis [26], graph node classification [27], dialogue generation [28], and association matching [29]. In addition, some research work such as [28, 30, 31] not only uses GAN technology to build user-friend social networks and user-project network graphs but also can calculate different influence weights of friends on users.

Zhang et al. [32] proposed the HetGNN model, which can jointly consider heterogeneous graph information as well

as heterogeneous contents information of each node effectively. Wang et al. [33] thought that a graph has strong relationship expression ability and proposed a user identity linkage method based on a heterogeneous graph attention network mechanism (UIL-HGAN). Wang et al. [34] proposed a novel heterogeneous graph neural network based on the hierarchical attention, including node-level and semantic-level attentions.

However, the goals of the above research are not about group event recommendations. Therefore, they are different from the problems studied in this paper.

6. Conclusion and Future Work

The paper present a leveraging Social Relationship based Graph Attention Model (SRGAM) for group event recommendation. The SRGAM model uses the social relationship information of users and events to alleviate the data sparse and cold start problems inherent in group event recommendation. We adopt an attention mechanism inside users, events and groups, and learn the different influence weights of various factors, and finally generate high-level comprehensive feature vectors of groups and events, which makes the prediction score of groups participation events more accurate. Experimenting on two real-world datasets, our SRGAM model performs best.

SRGAM has considered the social relationship between users and events, but does not consider the social relationship between groups. Thus, in future, we intend to integrate the group social relationship into the model, to further improve recommendation performance.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflict of interest.

Acknowledgments

This research was funded by the National Natural Science Foundation of China (No. 61772245), the Jiangxi Provincial Graduate Innovation Fund (No. YC2019-B093), and Science and Technology Project of Jiangxi Provincial Department of Education (No. GJJ181349).

References

[1] M. Li, D. Huang, B. Wei, and C. D. Wang, "Event recommendation via collective matrix factorization with event-user neighborhood," in *Intelligence Science and Big Data Engineering. IScIDE 2017. Lecture Notes in Computer Science, vol 10559*, Y. Sun, H. Lu, L. Zhang, J. Yang, and H. Huang, Eds., Springer, Cham, 2017.

[2] S. Purushotham and C. C. J. Kuo, "Modeling group dynamics for personalized group-event recommendation," in *Social Computing, Behavioral-Cultural Modeling, and Prediction. SBP 2015. Lecture Notes in Computer Science, vol 9021*, N. Agarwal, K. Xu, and N. Osgood, Eds., pp. 405–411, Springer, Cham, 2015.

[3] Y. Gu, J. Song, W. Liu, L. Zou, and Y. Yao, "CAMF: context aware matrix factorization for social recommendation," *Web Intelligence*, vol. 16, no. 1, pp. 53–71, 2018.

[4] Q. Yuan, G. Cong, and C. Lin, "COM: a generative model for group recommendation," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 163–172, New York, NY, USA, August 2014.

[5] T. Pham, X. Li, G. Cong, and Z. Zhang, "A general graph-based model for recommendation in event-based social networks," in *2015 IEEE 31st International Conference on Data Engineering*, pp. 567–578, Seoul, South Korea, April 2015.

[6] H. Yin, L. Zou, Q. V. H. Nguyen, Z. Huang, and X. Zhou, "Joint event-partner recommendation in event-based social networks," in *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, pp. 929–940, Paris, France, April 2018.

[7] M. M. Gonzalez, "A general recommendation model for heterogeneous networks," *Computing Reviews*, vol. 58, no. 7, pp. 418–418, 2017.

[8] T. N. Kipf and W. Max, "Semi-supervised classification with graph convolutional networks," in *Proceedings of the 5th International Conference on Learning Representations*, pp. 1–10, Toulon, France, 2017.

[9] Y.-L. Chen, L.-C. Cheng, and C.-N. Chuang, "A group recommendation system with consideration of interactions among group members," *Expert Systems with Applications*, vol. 34, no. 3, pp. 2082–2090, 2008.

[10] Y. Mo, B. Li, B. Wang, L. T. Yang, and M. Xu, "Event recommendation in social networks based on reverse random walk and participant scale control," *Future Generation Computer Systems*, vol. 79, pp. 383–395, 2018.

[11] D. Cao, X. He, L. Miao, Y. An, C. Yang, and R. Hong, "Attentive group recommendation," in *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 645–654, Ann Arbor, MI, USA, June 2018.

[12] V. Lucas, N. Tuan-Anh, T. Yi, Y. Liu, G. Cong, and X. Li, "Interact and decide: medley of sub-attention networks for effective group recommendation," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 255–264, Paris, France, July 2019.

[13] Z. Yu, X. Zhou, Y. Hao, and J. Gu, "TV program recommendation for multiple viewers based on user profile merging," *User Modeling and User-Adapted Interaction*, vol. 16, no. 1, pp. 63–82, 2006.

[14] V. R. Kagita, A. K. Pujari, and V. Padmanabhan, "Virtual user approach for group recommender systems using precedence relations," *Information Sciences*, vol. 294, no. 3, pp. 15–30, 2015.

[15] L. Naamani-Dery, M. Kalech, L. Rokach, and B. Shapira, "Preference elicitation for narrowing the recommended list for groups," in *Proceeding of the 8th ACM Conference on Recommender Systems*, pp. 333–336, Silicon Valley, CA, USA, October 2014.

[16] M. O'Connor, D. Cosley, J. A. Konstan, and J. Riedl, "PolyLens: a recommender system for groups of users," in *ECSCW*

2001, W. Prinz, M. Jarke, Y. Rogers, K. Schmidt, and V. Wulf, Eds., Springer, Dordrecht, 2002.

[17] Y. Zhang, Y. Du, and X. Meng, "Research on group recommender systems and their applications," *Chinese Journal of Computers*, vol. 39, no. 4, pp. 745–764, 2016.

[18] B. Li, B. Wang, Y. Mo et al., "A novel random walk and scale control method for event recommendation," in *2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*, pp. 228–235, Toulouse, France, July 2016.

[19] S. Liu, B. Wang, and M. Xu, "Event recommendation based on graph random walking and history preference reranking," in *Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval*, pp. 861–864, New York, NY, USA, August 2017.

[20] X.-W. Meng, S.-D. Liu, Y.-J Zhang, and X. Hu, "Research on social recommender systems," *Journal of Software*, vol. 26, no. 6, pp. 1356–1372, 2015.

[21] S. Liu, B. Wang, and M. Xu, "SERGE: successive event recommendation based on graph entropy for event-based social networks," *IEEE Access*, vol. 6, pp. 3020–3030, 2018.

[22] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," in *Proceedings of the 2th International Conference on Learning Representations*, pp. 1–14, Banff, Canada, 2014.

[23] D. Michael, B. Xavier, and V. Pierre, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proceedings of the 30th Neural Information Processing Systems Conference*, pp. 3844–3852, Barcelona, SPAIN, 2016.

[24] H. Mikael, B. Joan, and L. Yann, "Deep convolutional networks on graph-structured data," 2015, http://arxiv.org/abs/1506.05163.

[25] K. Alex, S. Ilya, and E. Geoffrey, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[26] J. Qiu, J. Tang, H. Ma, Y. Dong, K. Wang, and J. Tang, "DeepInf: socail influence prediction with deep learning," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 2110–2119, New York, NY, USA, July 2018.

[27] L. Gong and Q. Cheng, "Adaptive edge features guided graph attention networks," 2018, http://arxiv.org/abs/1809.02709.

[28] Q. Wu, H. Zhang, X. Gao et al., "Dual graph attention networks for deep latent representation of multifaceted social effects in recommender systems," in *Proceedings of the 2019 World Wide Web Conference*, pp. 2091–2102, New York, NY, USA, May 2019.

[29] T. Zhang, B. Liu, D. Niu, K. Lai, and Y. Xu, "Multiresolution graph attention networks for relevance matching," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pp. 933–942, New York, NY, USA, October 2018.

[30] W. Fan, Y. Ma, Q. Li et al., "Graph neural networks for social recommendation," in *Proceedings of the 2019 World Wide Web Conference*, pp. 417–426, New York, NY, USA, May 2019.

[31] W. Song, Z. Xiao, Y. Wang, L. Charlin, M. Zhang, and J. Tang, "Session-based social recommendation via dynamic graph attention networks," in *Proceedings of the 12th ACM International Conference on Web Search and Data Mining*, pp. 555–563, New York, NY, USA, January 2019.

[32] C. Zhang, D. Song, C. Huang, A. Swami, and N. V. Chawla, "Heterogeneous graph neural network," in *Proceedings of the 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 793–803, New York, NY, USA, July 2019.

[33] R. Wang, H. Zhu, L. Wang, Z. Chen, M. Gao, and Y. Xin, "User identity linkage across social networks by heterogeneous graph attention network modeling," *Applied Sciences*, vol. 10, article 5478, 16 pages, 2020.

[34] X. Wang, H. Ji, C. Shi et al., "Heterogeneous graph attention network," in *Proceedings of the 2019 World Wide Web Conference*, pp. 2022–2032, New York, NY, USA, May 2019.